BMC Runtime Component System
Configuration and Administration Guide

Supporting

Version 6.0 of MainView Infrastructure
MainView products
Version 10.1 of APPTUNE for DB2®
Version 10.1 of BMC System Performance for DB2
Version 10.1 of Pool Advisor for DB2
Version 10.1 of SQL/Explorer for DB2
Version 10.1 of SQL Performance for DB2

April 2011
**Contacting BMC Software**

You can access the BMC Software website at [http://www.bmc.com](http://www.bmc.com). From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

**United States and Canada**

<table>
<thead>
<tr>
<th>Address</th>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC SOFTWARE INC</td>
<td>713 918 8800 or</td>
<td>713 918 8000</td>
</tr>
<tr>
<td>2101 CITYWEST BLVD</td>
<td>800 841 2031</td>
<td></td>
</tr>
<tr>
<td>HOUSTON TX 77042-2827</td>
<td>USA</td>
<td></td>
</tr>
</tbody>
</table>

**Outside United States and Canada**

<table>
<thead>
<tr>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>(01) 713 918 8800</td>
<td>(01) 713 918 8000</td>
</tr>
</tbody>
</table>


BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners.

CICS, DB2, DFSMS, IBM, IMS, MQSeries, MVS, OS/390, RACF, REXX, VTAM, WebSphere, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Linux is the registered trademark of Linus Torvalds.

UNIX is the registered trademark of The Open Group in the US and other countries.

The information included in this documentation is the proprietary and confidential information of BMC Software, Inc., its affiliates, or licensors. Your use of this information is subject to the terms and conditions of the applicable End User License agreement for the product and to the proprietary and restricted rights notices included in the product documentation.

**Restricted rights legend**

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED -- RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2101 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.
Customer support

You can obtain technical support by using the BMC Software Customer Support website or by contacting Customer Support by telephone or e-mail. To expedite your inquiry, see “Before contacting BMC.”

Support website

You can obtain technical support from BMC 24 hours a day, 7 days a week at http://www.bmc.com/support. From this website, you can

- read overviews about support services and programs that BMC offers
- find the most current information about BMC products
- search a database for issues similar to yours and possible solutions
- order or download product documentation
- download products and maintenance
- report an issue or ask a question
- subscribe to receive proactive e-mail alerts when new product notices are released
- find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by telephone or e-mail

In the United States and Canada, if you need technical support and do not have access to the web, call 800 537 1813 or send an e-mail message to customer_support@bmc.com. (In the subject line, enter SupID:<yourSupportContractID>, such as SupID:12345). Outside the United States and Canada, contact your local support center for assistance.

Before contacting BMC

Have the following information available so that Customer Support can begin working on your issue immediately:

- product information
  - product name
  - product version (release number)
  - license number and password (trial or permanent)
- operating system and environment information
  - machine type
  - operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - system hardware configuration
  - serial numbers
  - related software (database, application, and communication) including type, version, and service pack or maintenance level
- sequence of events leading to the issue
- commands and options that you used
- messages received (and the time and date that you received them)
  - product error messages
  - messages from the operating system, such as file system full
  - messages from related software
License key and password information

If you have questions about your license key or password, use one of the following methods to get assistance:

- Send an e-mail message to customer_support@bmc.com.
Contents

About this book 13
Related publications ................................................................. 14
Conventions .............................................................................. 15
Syntax statements ................................................................. 15

Chapter 1  Understanding RTCS 17
Introduction .............................................................................. 18
Continuous system operation .................................................... 18
RTCS components .................................................................... 19
  RTCS kernel ................................................................. 20
  RTCS subsystem address space ............................................ 20
  RTCS system registry ....................................................... 21
  RTCS memory registry ..................................................... 21
  RTCS product registry ..................................................... 22
  RTCS Generalized Server address space ............................... 22
Understanding the RTCS system registry .............................. 23
  Examples of configuring the RTCS system registry .............. 23
  About the registry VLDS owner ......................................... 25
  When to unallocate the registry VLDS ................................. 25
  Registry instance state transition .................................... 26

Chapter 2  Using RTCS product libraries 29
Library categories ................................................................. 30
  Distribution libraries ........................................................ 31
  Target libraries .................................................................. 31
  Runtime production libraries .......................................... 32
RTCS target libraries ........................................................... 33
  RTCS subsystem program library (TOSZRTCS) .................. 33
  RTCS product program library (TOSZLINK) ....................... 34
  RTCS hypertext document library (TOSZHTML) ................. 34
  RTCS XML document library (TOSZRXML) ...................... 35
  Sample JCL and parameter library (TOSZCNTL) ................. 35

Chapter 3  Configuring RTCS 39
Introduction ................................................................. 40
Using MainView Customization ............................................ 40
Modifying an RTCS configuration ........................................ 41
IMPORT ................................................................. 97
OPTIONS ................................................................. 98
Return codes and messages ........................................... 99
  Return codes ......................................................... 99
  Messages ........................................................... 100

Appendix C  Sample JCL and z/OS image configuration members 101

  RTCS Initiator started task PROC .................................. 102
  STEPLIB DD statement .............................................. 103
  RTCS subsystem started task PROC ................................. 104
  RTCS Generalized Server started task PROC ..................... 105
  z/OS Program Properties Table additions ........................... 105

Appendix D  RTCS initialization member 107

Index 119
<table>
<thead>
<tr>
<th>Figure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registry instance state transition</td>
</tr>
<tr>
<td>RTCS product libraries</td>
</tr>
<tr>
<td>Sample JCL to create RTCS subsystem program library</td>
</tr>
<tr>
<td>Sample JCL to create RTCS product program library</td>
</tr>
<tr>
<td>Sample JCL to create RTCS hypertext document library</td>
</tr>
<tr>
<td>Sample JCL to create the RTCS system registry</td>
</tr>
<tr>
<td>Sample JCL to create RTCS product authorization table library</td>
</tr>
<tr>
<td>RACF example</td>
</tr>
<tr>
<td>CA ACF2 example</td>
</tr>
<tr>
<td>CA Top Secret example</td>
</tr>
<tr>
<td>RTCS subsystem messages</td>
</tr>
<tr>
<td>Generalized Server messages</td>
</tr>
<tr>
<td>RTCS registry Import Utility job</td>
</tr>
<tr>
<td>RTCS initiator started task PROC</td>
</tr>
<tr>
<td>RTCS subsystem started task PROC</td>
</tr>
<tr>
<td>RTCS Generalized Server started task PROC</td>
</tr>
<tr>
<td>z/OS Program Properties Table additions</td>
</tr>
<tr>
<td>RTCS initialization member</td>
</tr>
</tbody>
</table>
Tables

Tasks to manually configure RTCS ................................................................. 41
SET statements ......................................................................................... 78
Subcommands ............................................................................................ 79
Operands ....................................................................................................... 82
Registry Import Utility return codes ......................................................... 99
RTCS Initiator started task PROC customizations .................................. 102
About this book

This book contains information about configuring and administering the Runtime Component System (RTCS).

This book is written for IBM z/OS® technical professionals and managers who are involved in the installation, deployment, and management of RTCS, as well as the configuration and management of z/OS systems to support those products.

To use this book, you should be familiar with the following items:

- configuration of your IBM z/OS images, DASD volumes, ICF catalog structure, data set naming conventions, and TCP/IP communication environment
- MVS job control language (JCL), MVS operator commands, and the installed external security manager (ESM)

Like most BMC documentation, this book is available in printed and online formats. To request printed books or to view online books and notices (such as release notes and technical bulletins), see the Customer Support website at http://www.bmc.com/support. Most product shipments also include the books on a documentation CD.

**NOTE**

Online books are formatted as PDF or HTML files. To view, print, or copy PDF books, use the free Adobe Reader from Adobe Systems. If your product installation does not install the reader, you can obtain the reader at http://www.adobe.com.

The software also offers online Help. To access Help, press F1 within any product or click the Help button in graphical user interfaces (GUIs).
The following related publications supplement this book and the online Help:

<table>
<thead>
<tr>
<th>Category</th>
<th>Document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>installation</td>
<td><strong>MainView Installation Guide</strong></td>
<td>provides information about the installation of MainView products on OS/390 and z/OS systems</td>
</tr>
<tr>
<td>documents</td>
<td><strong>Installation Checklist Generator (ICG)</strong></td>
<td>creates a customized checklist of installation steps and information about installation requirements such as software, storage, and system requirements for your combination of products</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Best Practices: Installation, Implementation, and Maintenance</strong></td>
<td>provides recommendations for the installation, implementation, and maintenance of MainView products</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Common Customization Guide</strong></td>
<td>provides instructions for manually customizing the MainView environment for your products</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Security Guide</strong></td>
<td>describes the basics of how to define security for MainView products with an external security manager (ESM)</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Security Reference Manual</strong></td>
<td>provides complete information about the MainView security interfaces for windows mode, full-screen mode, and MainView Alternate Access</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Alternate Access Implementation and User Guide</strong></td>
<td>explains how to configure, start, and stop IBM VTAM® and EXCP AutoLogon sessions to access MainView products without TSO communication support</td>
</tr>
<tr>
<td>core documents</td>
<td><strong>MainView Administration Guide</strong></td>
<td>provides information about MainView operations, targets, single system image contexts, data sets, and diagnostic facilities</td>
</tr>
<tr>
<td></td>
<td><strong>MainView User Guide</strong></td>
<td>provides information about working with MainView products in windows mode, in full-screen mode, and from MainView Explorer</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Alarm Management Guide</strong></td>
<td>explains how to create and install alarm definitions that indicate when exceptions occur in a sysplex</td>
</tr>
<tr>
<td></td>
<td><strong>MainView Reference Summary</strong></td>
<td>provides a summary of commands that can be used to navigate and manage the MainView environment</td>
</tr>
<tr>
<td>notices</td>
<td>release notes, flashes, and technical bulletins</td>
<td>provide important product information and last-minute information</td>
</tr>
</tbody>
</table>
This book uses the following special conventions:

- All syntax, operating system terms, and literal examples are presented in this typeface.

- Variable text in path names, system messages, or syntax is displayed in italic text:

  ```
  testsys/instance/file\Name
  ```

- The symbol `=>` connects items in a menu sequence. For example, `Actions => Create Test` instructs you to choose the `Create Test` command from the `Actions` menu.

## Syntax statements

The following example shows a sample syntax statement:

```command
COMMAND KEYWORD1 [KEYWORD2 | KEYWORD3] KEYWORD4={YES | NO} fileName...
```

The following table explains conventions for syntax statements and provides examples:

<table>
<thead>
<tr>
<th>Item</th>
<th>Example</th>
</tr>
</thead>
</table>
| Items in italic type represent variables that you must replace with a name or value. If a variable is represented by two or more words, initial capitals distinguish the second and subsequent words. | alias  
databaseDirectory  
serverHostName |
| Brackets indicate a group of optional items. Do not type the brackets when you enter the option. A comma means that you can choose one or more of the listed options. You must use a comma to separate the options if you choose more than one option. | `[tableName, columnName, field]`  
`[-full, -incremental, -level]` (UNIX) |
### Syntax statements

<table>
<thead>
<tr>
<th>Item</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Braces indicate that at least one of the enclosed items is required. Do not type the braces when you enter the item.</td>
<td>{DBDName</td>
</tr>
<tr>
<td></td>
<td>UNLOAD device={disk</td>
</tr>
<tr>
<td></td>
<td>{-a</td>
</tr>
<tr>
<td>A vertical bar means that you can choose only one of the listed items. In the example, you would choose either commit or cancel.</td>
<td>{commit</td>
</tr>
<tr>
<td></td>
<td>{-commit</td>
</tr>
<tr>
<td>An ellipsis indicates that you can repeat the previous item or items as many times as necessary.</td>
<td>columnName . . .</td>
</tr>
</tbody>
</table>
Understanding RTCS

This chapter provides conceptual information about the RTCS subsystem, the RTCS kernel, the RTCS system registry, the RTCS Generalized Server, and their support for products.

Introduction .................................................. 18
Continuous system operation .................................... 18
RTCS components .............................................. 19
    RTCS kernel .............................................. 20
    RTCS subsystem address space ............................. 20
    RTCS system registry ..................................... 21
    RTCS memory registry ..................................... 21
    RTCS product registry ..................................... 22
    RTCS Generalized Server address space ...................... 22
Understanding the RTCS system registry ...................... 23
Introduction

The Runtime Component System (RTCS) is an infrastructure that uses the latest z/OS system facilities to take advantage of modern enterprise servers. The RTCS infrastructure forms the basis for products that are designed to simplify product installation and configuration.

The RTCS infrastructure is established in a target z/OS image by a dedicated z/OS subsystem control address space. After being initialized, the RTCS subsystem control address space only executes code that is necessary to respond to operator commands and RTCS system registry operations.

All RTCS administrative functions and all products execute in either a dedicated product address space (PAS) or in an RTCS Generalized Server address space that is started by RTCS for the exclusive use by one or more products.

Continuous system operation

Continuous system operation refers to software and system availability without interruption or downtime.

After the RTCS subsystem has been started, products can be started and run. If any product address spaces (PASs) exist, any attempt to stop or cancel the RTCS subsystem address space while these PASs are still executing will be suppressed. As a consequence, RTCS subsystem and kernel software updates cannot easily be performed by shutting down and restarting RTCS.

The RTCS subsystem provides a facility for updating software and configuration information that enables the RTCS kernel and any loaded dynamic link library (DLL) packages to be refreshed dynamically without disruption to ongoing operation. In addition, changes that are made to the RTCS system registry are immediately effective, and the RTCS Registry Import Utility is provided to effect the changes.

SMP/E-based maintenance of program object elements in the RTCS subsystem program library cannot interfere with ongoing RTCS kernel or subsystem functions because the RTCS subsystem address space does not retain an allocation for the target or production library after the RTCS kernel is initialized. RTCS dynamically deallocates the data set, which permits the data set to be updated or reallocated. When the library is needed again (for example, to refresh the RTCS kernel after application of maintenance), the library is dynamically allocated.
The RTCS kernel and DLL refresh facility dynamically allocate the program libraries with the original DSNAMEs. If these libraries are not the actual SMP/E target libraries, updated elements (or the entire library) must be copied to the original library that is being used in production. These DSNAMEs can be changed dynamically by an RTCS operator `SET` command (for more information, see “SET field=value” on page 78). Because these libraries are either dynamically deallocated or can be freed by stopping any product Generalized Server address spaces, updating the production libraries for this purpose is proper, and conventional program library utilities can safely be used to perform this function. In general, it is not necessary to shut down Generalized Server address spaces prior to performing maintenance using SMP/E or copying SMP/E target libraries to z/OS image production libraries.

The RTCS infrastructure comprises the following components:

- RTCS kernel
- RTCS subsystem address space
- RTCS system registry
- RTCS memory registry
- RTCS product registry
- RTCS Generalized Server address spaces
RTCS runs as a z/OS subsystem. During initialization, the RTCS subsystem loads the RTCS kernel into above-the-line common storage. All code that is directly used by products is physically located in the kernel; references to code in the RTCS address space are not allowed. After the RTCS subsystem has been successfully initialized and the RTCS kernel has been loaded, RTCS services are available to applications and products that have been written to use the services.

Only one instance of RTCS can execute on each z/OS image. After the RTCS kernel is initialized, the RTCS subsystem control address space remains in the system as the owner of certain z/OS resources. Other than responding to operator commands and remote requests for local system registry data, plus hardening any updates made to the system registry (which are infrequent), it is essentially idle, executing very little code.

Products interact only with the kernel and not directly with the RTCS subsystem address space. Only operator commands that are directed specifically to RTCS (such as MODIFY) and RTCS system registry activity will cause any instructions to be executed in the RTCS subsystem control address space. Operator commands are normally only required when you want to start RTCS Generalized Server (product) address spaces or to control RTCS system registry operations.

The main responsibilities of the RTCS subsystem address space are to initialize the kernel and to provide the point of ownership for certain z/OS resources. z/OS exits and cross-memory resources that are established by the RTCS subsystem are used by functions and code in other permanent z/OS system address spaces, which never terminate. Because these RTCS functions, z/OS resources, and the RTCS kernel code are normally expected to be available for the life of the IPL, the RTCS subsystem address space should not be shut down.

Because the RTCS subsystem address space should not be stopped, starting it under any JES as a started task is not recommended. Instead, it should be started under the Master Subsystem by using SUB=MSTR on the START command.

Because the RTCS subsystem address space should not be shut down after it is started, only a limited number of data sets are allocated to it. The RTCS system registry VSAM linear data set (LDS) is the only data set that normally remains allocated.
The RTCS subsystem program library is a separate, dedicated SMP/E target library that is not shared by any other product. It is dynamically allocated by the RTCS subsystem address space and is deallocated after the RTCS kernel is initialized. The RTCS subsystem program library is required to be a partitioned data set extended (PDSE) program library.

**RTCS system registry**

The RTCS system registry is a repository for configuration parameters, product definitions, and other typical PARMLIB-type data. The RTCS system registry can be seen as read/write auxiliary storage for the RTCS subsystem and all products that use the registry. Data in the RTCS system registry is accessed as if it were in virtual storage, but it is actually stored on DASD in a VSAM LDS. The RTCS system registry is managed and accessed much like an z/OS page data set, except that it is organized in a tree-like structure, similar to the Microsoft Windows registry. This structure speeds up the process of locating data in the registry because complex, self-defining structures can be stored as easily as a single byte or a string of characters.

The RTCS system registry can optionally be shared with additional RTCS subsystems on other z/OS images in the same sysplex. In the shared mode, one of the RTCS address spaces becomes the local registry owner and manages all access to the RTCS system registry. The other RTCS address spaces become remote registry accessors and make requests to put data into the registry and to retrieve data from the registry to the local owner through an XCF communications pipeline.

For more information about the RTCS system registry, see “Understanding the RTCS system registry” on page 23.

**RTCS memory registry**

The RTCS memory registry is not backed by a VSAM VLDS and is intended to retain information temporarily as long as the RTCS subsystem is available. All content in the RTCS memory registry is lost when RTCS is shut down or the z/OS system image fails. The RTCS memory registry cannot be shared. There is one for each RTCS subsystem on each z/OS system image.
RTCS product registry

The RTCS product registry is a repository for product-specific information. It is defined by the products and is optionally backed by a VSAM LDS cluster.

**NOTE**

This VSAM LDS cluster must be allocated for use by the RTCS subsystem to back the primary RTCS registry partition using the z/OS Data-In-Virtual (DIV) service. Because it is a VSAM cluster and will be dynamically allocated by the RTCS subsystem address space using only its DSNAME, this library **must** be cataloged.

All processing activity uses the same access techniques as the RTCS system registry, and like the RTCS system registry is organized in a tree-like structure.

The RTCS product registry can optionally be shared with another copy of the product address space either on the same system or on a different z/OS image in the same sysplex. In the shared mode, one of the product address spaces becomes the local registry owner and manages all access to the RTCS product registry. The other product address spaces become remote registry accessors and make requests to put data into the registry and to retrieve data from the registry to the local owner through an XCF communications pipeline.

RTCS Generalized Server address space

Code for products does not execute in the RTCS subsystem address space. Instead, products execute either in their own address space or are started by RTCS in an RTCS Generalized Server address space.

A single RTCS Generalized Server address space can run multiple products, but all of the products in the address space must execute in the same protection key (keys 0 through 9 are supported). Multiple General Server address spaces might be started if products that execute in different protection keys need to be running at the same time.
Understanding the RTCS system registry

The RTCS system registry is backed by a VSAM LDS, whose DSNAME is specified by the SREGVLDS option in the RTCS initialization member of Logical PARMLIB. The z/OS Data-In-Virtual (DIV) service writes all changes that are made to the contents of the registry to the LDS.

Each z/OS image requires some form of access to the data in the registry VSAM LDS. A single VSAM LDS can be shared among all members of the sysplex or a dedicated VSAM LDS can be used for each individual z/OS system image. BMC recommends that you share a single VSAM LDS among all members of a sysplex. The MainView CAS places certain configuration information in the registry in order to make it available to all CASs in the sysplex. If you do not share a single VSAM LDS for the registry, you will not be able to take advantage of this CAS function.

A dedicated VSAM LDS can be configured for each RTCS subsystem or a single VSAM LDS can be shared by all members of the sysplex. If sharing a VSAM LDS, only one member actually allocates the registry VLDS with DISP=OLD, but the data in the registry is made available to other (remote) members of the sysplex using the XCF communications facility of z/OS.

Examples of configuring the RTCS system registry

The following examples illustrate different configurations for the RTCS system registry. The examples contain only the statements in the RTCS initialization member (OSZINIxx) that configure the sharing of the RTCS system registry. An example of an entire RTCS initialization member is in Appendix D, “RTCS initialization member.”

Example of separating production and test systems

The following statements would be used in OSZINIxx for the production systems to indicate that the production systems will

- communicate only with other production systems using XCF group RTCSPROD
- share registry hilevel.PRODUCTION.REGISTRY

| SREGVLDS=hilevel.PRODUCTION.REGISTRY |
| REGISTRY-XCF-GROUP=RTCSPROD |
| ELIGIBLE-OWNER |
| NOPRIVATE-REGISTRY |
| SHARED-REGISTRY |
The following statements would be used in OSZINIxx for the test systems to indicate that the test systems will

- communicate only with other test systems using XCF group RTCSTEST
- share registry hilevel.TEST.REGISTRY

```
SREGVLDs=hilevel.TEST.REGISTRY
REGISTRY-XCF-GROUP=RTCSTEST
ELIGIBLE-OWNER
NOPRIVATE-REGISTRY
SHARED-REGISTRY
```

**Example of adding an additional XCF group**

The following statements would be used in OSZINIxx to define an additional XCF group that will

- communicate only with other systems using XCF group RTCSGRPn
- share registry hilevel.GRPn.REGISTRY

```
SREGVLDs=hilevel.GRPn.REGISTRY
REGISTRY-XCF-GROUP=RTCSUSER
ELIGIBLE-OWNER
NOPRIVATE-REGISTRY
SHARED-REGISTRY
```

**Example of a nonshared RTCS system registry**

The following statements would be used in OSZINIxx to define an RTCS system registry that will

- co-exist with any other RTCS in a sysplex
- communicate with no other RTCS
- allocate and open but not expose, a private registry unique to the image where this RTCS is started

```
SREGVLDs=hilevel.&SYSID..BMCMQ.REGISTRY
ELIGIBLE-OWNER
PRIVATE-REGISTRY
NOSHARED-REGISTRY
```
About the registry VLDS owner

An XCF group member which has dynamically allocated the registry VLDS in a shared registry configuration is said to be the Local Owner. If the Local Owner of the registry has exposed the registry, the member is said to have acquired the registry VLDS. Exposed means that the other members in the XCF group are aware of the fact that the Local Owner member has dynamically allocated the registry VLDS and is allowing other, remote (non-Local Owner XCF group members) to access data in the registry.

The registry VLDS can be unexposed on the Local Owner member, in which case remote XCF group members will not be able to access data in the registry, but address spaces and applications on the Local Owner's z/OS system image will be able to. After being unexposed, the registry can be exposed again, which allows remote XCF group members to again access data in the registry.

When the Local Owner needs to relinquish ownership if the registry VLDS, the registry can be released, in which case any other member of the XCF group that is eligible to acquire the registry will attempt to do so. The first one that acquires the registry becomes the new Local Owner. If a specific new Local Owner is desired, that member of the XCF group can be designated in a REGISTRY TRANSFER command and it alone will then attempt to reacquire the registry VLDS. If the registry has been drained and unallocated, a REGISTRY ACQUIRE command may be used to have the XCF member on which it was entered attempt to dynamically allocate the registry VLDS, expose the registry, and reestablish remote instance access to the data in the registry for all members of the registry’s XCF group.

When to unallocate the registry VLDS

The purpose of being able to unallocate the registry VLDS is so that its contents can be backed up using normal VSAM utility and DASD backup programs, or so that DASD maintenance activities can take place, which might even include renaming or moving the location of the VSAM LDS. If the DSNAME of the registry VLDS is changed, the new DSNAME can be specified by the REGISTRY ALTER DSNAME command.

NOTE

Changing the DSNAME by using the REGISTRY ALTER DSNAME command does not alter the RTCS subsystem initialization parameters in Logical PARMLIB, nor in any parameter specification for a product that has its own registry. If you want to permanently change the DSNAME of the registry VLDS, you must appropriately update the location where this DSNAME is originally and uniquely specified.
To change the RTCS system registry VLDS to a different RTCS system registry VLDS while RTCS is running

1. Allocate a new RTCS system registry of the proper size.

2. Issue the following commands:
   
   \[ F \text{ RTCS,REGISTRY UNEXPOSE} \]
   \[ F \text{ RTCS,REGISTRY UNALLOC} \]
   \[ F \text{ RTCS,REGISTRY ALTER DSNAMEnewSregvldsName} \]
   \[ F \text{ RTCS,REGISTRY ACQUIRE} \]

3. Change the OSZINI\text{xx SREGVLDS} parameter on all sharing systems to the new registry data set name.

Registry instance state transition

Figure 1 on page 27 illustrates the transition state of the registry VLDS.

In the figure

- RELEASE instructs the current Local Owner to unallocate the registry VLDS and to inform Remote XCF group members that it has done so, at which point all remote members that are eligible to be a Local Owner attempt to acquire the registry.

- TRANSFER instructs the current Local Owner to unallocate the registry VLDS and specifically indicates which Remote XCF group member is to attempt to acquire the registry in preference to all others.
Figure 1  Registry instance state transition

- VLDS Allocated
- Exposed
- Available on Local Owner and from Remotes

- VLDS Allocated
- Not exposed
- Available on Local Owner only (not from Remotes)

- UNALLOCATE

- REALLOCATE

- RELEASE (note 1)
- TRANSFER (note 2)

- ACQUIRE
Chapter 2 Using RTCS product libraries

This chapter describes the product libraries that are used by the RTCS product family. This chapter explains how the product libraries are created and describes their intended uses.

Library categories .......................................................... 30
   Distribution libraries ......................................................... 31
   Target libraries ............................................................... 31
   Runtime production libraries ............................................ 32
RTCS target libraries ......................................................... 33
   RTCS subsystem program library (TOSZRTCS) ..................... 33
   RTCS product program library (TOSZLINK) ......................... 34
   RTCS hypertext document library (TOSZHTML) ..................... 34
   RTCS XML document library (TOSZRXML) .......................... 35
   Sample JCL and parameter library (TOSZCNTL) .................... 35
Library categories

RTCS is installed by using the BMC Installation System, which is SMP/E based. The BMC Installation System dictates the use of sets of distribution and target libraries for installation and maintenance of product function SYSMODs and PTFs. Most SMP/E-installed products can normally be run directly out of the SMP/E target libraries. For RTCS, BMC Software recommends using copies of the target libraries that are needed at runtime. These copies are called production libraries and might need to be z/OS image-specific.

Figure 2  RTCS product libraries
Distribution libraries

Distribution libraries are constructed by SMP/E as part of function SYSMOD ACCEPT processing. They contain the individually-distributed and replaceable elements that are used to construct program objects and members in the target libraries. Distribution libraries are read-only and should not be modified or customized.

The RTCS distribution libraries are as follows:

- AOSZRTCS: RTCS Initiator, RTCS subsystem, and RTCS kernel
- AOSZLINK: RTCS utilities and HTTP server
- AOSC70D: only used by certain RTCS-based products
- AOSZHTML: hypertext documents
- AOSZXML: XML documents
- AOSZCNTL: PROCs, sample JCL, and PARMLIB members

Target libraries

Target libraries are constructed by SMP/E as part of function SYSMOD or PTF APPLY processing. They contain program objects, hypertext documents, parameters, procedures, and samples. The target libraries should not normally be made available for production use on a target z/OS image.

The RTCS target libraries are as follows:

- TOSZRTCS: RTCS Initiator, RTCS subsystem, and RTCS kernel
- TOSZLINK: RTCS utilities and HTTP Server
- TOSZHTML: hypertext documents
- TOSZXML: XML documents
- TOSZCNTL: PROCs, sample JCL, and PARMLIB members

BMC Software recommends making copies of the following RTCS target libraries for use as production libraries on each z/OS image. These libraries (or, preferably, their production copies) are required on an z/OS image in production to execute RTCS. These libraries are referred to collectively in this book as the runtime libraries.

- TOSZRTCS
- TOSZLINK
- TOSZHTML

For more information about each RTCS target library, see “RTCS target libraries” on page 33.
Runtime production libraries

Because the runtime libraries are PDSE data sets, consider making a dedicated copy of the runtime libraries for each target z/OS image on which you want to run RTCS in production. These copies are referred to as "production libraries." There is no reason for the production libraries to be allocated as SMS-managed data sets. For example, they might be allocated on one of the target z/OS image’s SYSRES volumes. But they can be SMS-managed if you desire.

When you use production libraries, maintenance can be applied to the original SMP/E target libraries by using an allocation that specifies DISP=OLD. Subsequently, maintenance can be copied to the production libraries for use on one or more specific z/OS images.

Applying maintenance to a product using SMP/E while the SMP/E target libraries are in use on an z/OS image might produce incorrect results or ABENDs when library members in the target library are accessed that are out of sync with other members that have already been accessed. Because of this possibility, BMC recommends that you do not configure the RTCS started task PROCs and the RTCS system registry to use your actual SMP/E target libraries on any production machine. However, you can configure a test z/OS image or LPAR to use the SMP/E-maintained target libraries directly if the consequences of an unexpected failure as a result of applying maintenance to libraries that are being used concurrently are deemed acceptable.

**NOTE**

You do not need to make production copies of the TOSZRXML and TOSZCNTL libraries because they contain samples and are used only during RTCS system registry initialization.
Sharing PDSEs in a sysplex

Because of PDSE sharing requirements, you might want to consider making a unique production copy of each runtime library for exclusive use on each target z/OS image. If your systems, however, are configured properly for PDSE sharing across all members of the sysplex, you can share the runtime libraries across multiple z/OS images if you adhere to the IBM z/OS requirements for sharing PDSEs in a sysplex. These requirements include the following:

- The libraries can only be shared among members of a sysplex (never outside).
- The GRSplex must match the sysplex.
- No additional systems can access the PDSEs from outside of the sysplex or outside of the GRSplex. CA-MIM cannot be used to mediate shared access to PDSE libraries, even in read-only mode.

**NOTE**

If you do not strictly adhere to the IBM z/OS requirements, the PDSE might become corrupted, and products might subsequently fail in unpredictable ways. A corrupt PDSE library might cause system outages on one or more images on your sysplex.

RTCS target libraries

The RTCS product family uses a limited number of product libraries. The contents of these target libraries are established by an SMP/E-based installation process.

This section describes the RTCS target libraries. For more information, see “Target libraries” on page 31.

RTCS subsystem program library (TOSZRTCS)

The RTCS subsystem program library (TOSZRTCS) must be a PDSE and can be used as is in the RTCS Initiator PROC, where it must be defined in the STEPLIB DD statement. This library contains only RTCS subsystem PM3 (IBM DFSMS/MVS Release 4 compatible) program objects. This library is used only during RTCS subsystem initialization by the RTCS Initiator address space, and then the data set is dynamically allocated to the RTCS subsystem address space. After the RTCS kernel has been initialized, this library is dynamically deallocated, and it is not referenced again unless the RTCS kernel needs to be refreshed.
RTCS product program library (TOSZLINK)

The RTCS product program library (TOSZLINK) must be a PDSE. It can be used as is by specifying its DSNAME in the POSZLINK option in the RTCS initialization member of Logical PARMLIB. This library contains RTCS component PM3 (DFSM5/MVS release 4 compatible) program objects and contains no RTCS subsystem code.

This library is referenced and validated by the RTCS Initiator, which records its DSNAME. However, no programs in the library are executed during RTCS subsystem initialization. This data set is dynamically allocated by the RTCS Generalized Server program (OSZEXEC[n]) when an RTCS component is executed in a started task or batch job. Thus, this data set will be allocated to any address space that is running an RTCS component program or utility.

RTCS hypertext document library (TOSZHTML)

The members of the RTCS hypertext document library (TOSZHTML) are HTML documents or binary, byte-stream files. The RTCS hypertext document library contains variable-length record files used by RTCS utilities and product address spaces, including batch address spaces. RTCS batch utility programs, such as the Registry Import Utility, reference members of this library.

The RTCS hypertext document library is dynamically allocated by the RTCS Generalized Server, but only when it is known to be needed, whether in a started task or in batch. It is also used in a DD statement in the JCL used to execute certain RTCS utility programs in batch. The data set remains allocated to the Generalized Server or batch address space as long as it continues to execute. This data set is dynamically allocated by the RTCS Initiator address space to verify its existence and some of its contents, but it is not allocated by the RTCS subsystem address space.

Maintenance is more convenient if this data set is allocated as a PDSE. This data set can be used as is by specifying its DSNAME in the POSZHTML option in the RTCS initialization member of Logical PARMLIB.
The RTCS XML document library (TOSZRXML) is used by the RTCS Registry Import Utility (RIU). This data set is never used by the RTCS subsystem control address space, nor by a Generalized Server address space because it only contains XML documents that are used in the initial setup of the RTCS system registry. The DSNAME of this library needs to be specified in a SET statement in the SYSIN input stream each time the RTCS RIU is executed. For more information, see Appendix B, “RTCS Registry Import Utility.”

Sample JCL and parameter library (TOSZCNTL)

This library contains several samples that you can use to configure the RTCS started task PROCs and the RTCS initialization member. The library also contains sample JCL to execute the RTCS RIU and sample parameter input members for RTCS registry definitions.

The following TOSZCNTL library members are of particular interest after RTCS is installed and when a target z/OS image is being configured to run RTCS:

- OSZ$INIT
- OSZ$PARM
- OSZ$RTCS
- OSZ$EXEC
- OSZ$REGI
- OSZSCHED

The following sections provide instructions on how to use these members.

**OSZ$INIT**

OSZ$INIT is sample RTCS Initiator started task procedure (PROC) JCL. Copy this member into SYS1.PROCLIB (or another Master Subsystem started task procedure library that is specified in the z/OS image’s Master JCL). BMC Software recommends naming this copy OSZINIT; however, this is not required.

Customize the STEPLIB DD statement to specify the DSNAME of the RTCS subsystem program library.

- If OSZINIT is to be started under the z/OS Master Subsystem (MSTR), the PROC must be located in SYS1.PROCLIB (or equivalent). You would then use the following operator command to start the RTCS Initiator:
But if OSZINIT is to be started under the Primary JES, the PROC can be located in any JES-managed started task PROCLIB. You would then use the following operator command to start the RTCS Initiator:

```
$ OSZINIT
```

**OSZ$RTCS**

OSZ$RTCS is sample RTCS subsystem started task procedure (PROC) JCL.

If OSZINIT is started under the Master Subsystem, the PROC must be located in SYS1.PROCLIB (or equivalent) because OSZRTCS will also, by default, be started by the RTCS Initiator (internally) under the z/OS Master Subsystem (MSTR).

Because no parameters, data sets, or DD statements exist in this PROC, customization is unnecessary. Copy this member into SYS1.PROCLIB (or another Master Subsystem started task procedure library that is specified in the z/OS image’s Master JCL) if RTCS is to be started under the Master Subsystem, or into a JES-managed PROCLIB if RTCS is to be started under the primary JES. BMC Software recommends naming the copy OSZRTCS. You must specify the RTCS subsystem PROC name in the OSZRTCS-PROC option in the RTCS initialization member of Logical PARMLIB.

**OSZ$EXEC**

OSZ$EXEC is sample RTCS Generalized Server address space started task procedure (PROC) JCL.

Because no parameters, data sets, or DD statements exist in this PROC, customization is unnecessary.

Copy this member to both of the following locations:

- any JES started task procedure library
- SYS1.PROCLIB (or another Master Subsystem started task procedure library that is specified in the z/OS image’s master JCL)

BMC Software recommends naming the copy OSZEXEC. You must specify the RTCS Generalized Server PROC name in the OSZEXEC-PROC option in the RTCS initialization member of Logical PARMLIB.
OSZ$REGI

OSZ$REGI is sample JCL that is used to execute the RTCS RIU. Copy this member to a site-specific data set. In addition to the JOB statement, the JCL has DD statements that need to be customized to specify the DSNAMEs of the SMP/E TOSZRXXML target XML document library as well as any customized XML document library.

OSZSCHED

OSZSCHED is a sample set of required SYS1.PARMLIB PPT additions for RTCS. You can copy this member as is into the appropriate z/OS target image-specific PARMLIB data set. Copy it as a new member named SCHEDxx, or copy its contents into any existing PPT member.
This chapter explains how to configure a target z/OS image to be able to start the RTCS Initiator and initialize the RTCS subsystem.

Introduction ................................................................. 40
Using MainView Customization ........................................ 40
Modifying an RTCS configuration ....................................... 41
Manually configuring RTCS ................................................ 41
  Creating POSZRTCS ..................................................... 42
  Creating POSZLINK .................................................... 45
  Creating POSZHTML .................................................. 46
  Authorizing the RTCS libraries .......................... 48
  Allocating the RTCS system registry ......................... 48
  Allocating the product authorization table library ............ 50
  Ensuring sufficient common area data spaces ................. 51
  Ensuring sufficient system linkage indexes .................... 51
  Adding RTCS programs to the z/OS PPT ......................... 51
Selecting an SSID for the RTCS subsystem control address space 52
Updating the subsystem name table ............................... 53
Creating and customizing the RTCS Initiator started task PROC 53
Creating the RTCS subsystem started task PROC .................. 55
Creating the RTCS Generalized Server started task PROCs .... 55
Establishing ESM user IDs for RTCS address spaces ............. 57
Starting RTCS .............................................................. 66
Adding the RTCS start command to SYS1.PARMLIB ............... 66
Introduction

RTCS is installed by using the BMC Installation System. During the installation, you also configure RTCS by using MainView Customization.

- BMC recommends using MainView Customization to configure RTCS. To use MainView Customization, see “Using MainView Customization.”

- If you need to modify the RTCS configuration after using MainView Customization, use the information in “Modifying an RTCS configuration” on page 41.

- If you need to manually configure RTCS, use the information in “Manually configuring RTCS” on page 41.

The z/OS image configuration process must be performed so that the RTCS Initiator and the RTCS subsystem started task programs can receive control in the proper state and key from the z/OS initiator. This enables the RTCS subsystem control address space to establish RTCS in the z/OS image as a formal z/OS subsystem. Started task PROCs to start these address spaces must be placed in the proper STC procedure libraries (PROCLIBs). In addition, required ESM attributes must be established for the RTCS subsystem and Generalized Server address spaces. The required product data sets (or production copies that were made) must be accessible from the target z/OS image.

Using MainView Customization

When you use MainView Customization, it creates a default environment for a shared RTCS system registry at a sysplex level. You provide the name of the JCL procedures to start the RTCS initiator started task, the RTCS subsystem address space, and the RTCS Generalized Server started task. You also supply the sysplex name and the names of the z/OS images in the sysplex. Use the information in Appendix A, “Planning RTCS configuration,” to prepare before using MainView Customization.

MainView Customization provides the ability to define one or more RTCS system registries all at the same time, you just specify the different sysplex names and z/OS image names on which you want the environments defined. The Online Help explains how to use MainView Customization to define multiple instances of an environment.

After specifying the information to MainView Customization, it

- Creates the jobs you need to run to implement the environment
Modifying an RTCS configuration

At times, you might need to modify the default RTCS environment created by MainView Customization. For example, if you do not want to share an RTCS system registry.

Use the information in Appendix C, “Sample JCL and z/OS image configuration members,” and Appendix D, “RTCS initialization member,” as a reference to modify the RTCS started task procedures, the z/OS program property table, or the RTCS initialization member as needed.

**NOTE**

Before making any modifications, complete MainView Customization, run the created jobs, and complete the tasks on the created checklists. At any time, you can rerun the created jobs to return to the default environment created by MainView Customization.

Manually configuring RTCS

If you do not use MainView Customization to configure RTCS, complete the steps in Table 1 to manually configure a target z/OS image to run RTCS.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>(optional, but recommended)</em> Create the production RTCS subsystem program library (POSZRTCS).</td>
<td>“Creating POSZRTCS” on page 42</td>
</tr>
<tr>
<td>2</td>
<td><em>(optional, but recommended)</em> Create the production RTCS product program library (POSZLINK).</td>
<td>“Creating POSZLINK” on page 45</td>
</tr>
<tr>
<td>3</td>
<td><em>(optional, but recommended)</em> Create the production RTCS hypertext document library (POSZHTML).</td>
<td>“Creating POSZHTML” on page 46</td>
</tr>
<tr>
<td>4</td>
<td>APF-authorize the RTCS subsystem program library and the RTCS product program library</td>
<td>“Authorizing the RTCS libraries” on page 48</td>
</tr>
</tbody>
</table>
Because of PDSE sharing requirements, consider making a unique copy of this library for exclusive production use on each individual target z/OS image on which RTCS is to be started. For more information about production libraries and their advantages, see “Runtime production libraries” on page 32. Alternatively, you can use a single data set, shared among all members of a single sysplex. For information about sharing this library in a sysplex, see “Sharing PDSEs in a sysplex” on page 33.

**Requirements**

The following requirements apply when creating POSZRTCS:
The POSZRTCS library that you create must be a PDSE library. BMC recommends that it be allocated as a non-SMS-managed PDSE (that is, no STORCLAS specified), but it can be SMS-managed if desired.

The name of the data set that you create must be specified by the DSNAME parameter on the STEPLIB DD statement in the RTCS Initiator PROC (see “Creating and customizing the RTCS Initiator started task PROC” on page 53).

This is the only RTCS production library whose DSNAME needs to be specified in a DD statement in a started task PROC. If each of the remaining RTCS production libraries are allocated with a DSNAME prefix that matches this library, along with its recommended low-level index, the RTCS Initiator will be able to derive a matching, default DSNAME for them, which the RTCS subsystem will then use to dynamically allocate these data sets. If not, then the DSNAMEs of the non-conforming production libraries will need to be specified by the appropriate parameters in the RTCS Initialization member (which is usually, OSZINI00) of the z/OS Logical PARMLIB data set.

Because it will be dynamically allocated by the RTCS Allocator (OSZMOSYS, the first program to run in the RTCS Subsystem address space), this library must be cataloged. The catalog must be available when RTCS is STARTed.

The low-level index of the DSNAME selected for this production library should be POSZRTCS or OSZRTCS.

To create POSZRTCS

To allocate and copy the target SMP/E RTCS subsystem program library (TOSZRTCS) into a new POSZRTCS PDSE, you can customize the sample JCL shown in Figure 3 on page 44. The text that should be customized is shown in bold. This sample JCL is contained in member OSZINJ71 in the TOSZCNTL target library.
Creating POSZRTCS

**Figure 3** Sample JCL to create RTCS subsystem program library

```plaintext
//OSZINJ71 JOB (acct), 'CREATE POSZRTCS PDSE',
// MSGLEVEL=(1,1), CLASS=A, TIME=1
//
//* acct    JOB statement accounting information
//* xxxxxx   VOLSER on which to allocate .POSZRTCS PDSE
//* tlibprfx DSN prefix of RTCS SMP/E target libraries
//*
//ALLOC EXEC PGM=IEFBR14
//*
//* Allocate a new non-SMS-managed cataloged PDSE for
//* the RTCS Subsystem Library. This data set may be
//* SMS-managed, but the JCL will have to be changed
//* to allocate an SMS-managed data set. There is no
//* particular need for the library to be SMS-managed.
//* This data set CAN be shared by more than one system
//* if MVS sysplex PDSE sharing requirements are met.
//*
//OSZRTCS DD DSN=SYS2.RTCS.POSZRTCS,
// DISP=(NEW,CATLG),
// UNIT=3390, VOLSER=xxxxxx,
// DSNTYPE=LIBRARY, SPACE=(CYL, (16,5,01)),
// DSORG=PO, RECFM=U, LRECL=0, BLKSIZE=23476
//*
//COPY EXEC PGM=IEBCOPY, TIME=1, REGION=4M
//*
//* Copy SMP/E Target RTCS Subsystem LIBRARY
//* to the newly-allocated "production" PDSE.
//*
//SYSPRINT DD SYSOUT=* 
//TOSZRTCS DD DISP=SHR, DSN=tlibprfx.TOSZRTCS
//OSZRTCS DD DISP=SHR, DSN=*.ALLOC.OSZRTCS
//SYSIN DD *
//COPY INDD=((TOSZRTCS,R)), OUTDD=OSZRTCS
//*
```
Creating POSZLINK

Because of PDSE sharing requirements, consider making a unique copy of this library for exclusive production use on each individual target z/OS image on which RTCS is to be started. For more information about production libraries and their advantages, see “Runtime production libraries” on page 32. Alternatively, you can use a single data set, shared among all members of a single sysplex. For information about sharing this library in a sysplex, see “Sharing PDSEs in a sysplex” on page 33.

Requirements

The following requirements apply when creating POSZLINK:

- The POSZLINK library that you create must be a PDSE library.
- The low-level index of the DSNAME selected for this production library should be POSZLINK or OSZLINK.
- The name of the data set that you create will need to be specified in the RTCS Initialization member in the POSZLINK parameter if its low-level index is not POSZLINK or OSZLINK.
- Because it will be dynamically allocated by the RTCS Allocator (OSZMOSYS) to the RTCS subsystem address space using only its DSNAME, this library must be cataloged.

To create POSZLINK

To allocate and copy the target SMP/E RTCS product program library (TOSZLINK) into a new POSZLINK PDSE, you can customize the sample JCL shown in Figure 4 on page 46. The text that should be customized is shown in bold. This sample JCL is contained in member OSZINJ73 in the TOSZCNTL target library.
Because of PDSE sharing requirements, consider making a unique copy of this library for exclusive production use on each individual target z/OS image on which RTCS is to be started. For more information about production libraries and their advantages, see “Runtime production libraries” on page 32. Alternatively, you can use a single data set, shared among all members of a single sysplex. For information about sharing this library in a sysplex, see “Sharing PDSEs in a sysplex” on page 33.

**Requirements**

The following requirements apply when creating POSZLINK:

- The POSZHTML target library that you create must be a PDSE library.
The low-level index of the DSNAME selected for this production library should be POSZHTML or OSZHTML.

The name of the data set that you create will need to be specified in the RTCS Initialization member in the POSZHTML option if its low-level index is not POSZHTML or OSZHTML.

Because it will be dynamically allocated using only its DSNAME, this library must be cataloged.

**To create POSZHTML**

To allocate and copy the target SMP/E RTCS hypertext document library (TOSZHTML) into a new PDSE, you can customize the sample JCL shown in Figure 5. The text that should be customized is shown in **bold**. This sample JCL is contained in member OSZINJ75 in the TOSZCNTL target library.

**Figure 5 Sample JCL to create RTCS hypertext document library**

```
//OSZINJ75 JOB (acct),'CREATE POSZHTML PDSE',
   MSGLEVEL=(1,1),CLASS=A,TIME=1
//*
//* acct      JOB statement accounting information
//* zzzzzz    VOLSER on which to allocate .POSZHTML PDSE
//* tlibprfx  DSN prefix of RTCS SMP/E target libraries
//*
//ALLOC   EXEC PGM=IEFBR14
//*
//* Allocate a new non-SMS-managed cataloged PDSE for
//* the RTCS HTML Doc Library. This data set may be
//* SMS-managed, but the JCL will have to be changed
//* to allocate an SMS-managed data set. There is no
//* particular need for the library to be SMS-managed.
//* This data set CAN be shared by more than one system
//* if MVS sysplex PDSE sharing requirements are met.
//*
//OSZHTML   DD DSN=SYS2.RTCS.POSZHTML,
//           DISP=(NEW,CATLG),
//           UNIT=3390,VOL=SER=zzzzzz,
//           DSNTYPE=LIBRARY,SPACE=(CYL,(25,5,1)),
//           DSORG=PO,RECFM=VB,LRECL=260,BLKSIZE=6504
//*
//COPY    EXEC PGM=IEBCOPY,TIME=1,REGION=4M
//*
//* Copy SMP/E Target RTCS HTML Doc LIBRARY
//* to the newly-allocated "production" PDSE.
//*
//SYSPRINT DD SYSOUT=*  
//TOSZHTML DD DISP=SHR,DSN=tlibprfx.TOSZHTML
//OSZHTML  DD DISP=SHR,DSN=*._ALLOC.OSZHTML
//SYSIN    DD *
//COPY INDD=((TOSZHTML,R)),OUTDD=OSZHTML
//*
```
Authorizing the RTCS libraries

Temporarily or permanently APF-authorize the RTCS subsystem program library and the RTCS product program library.

NOTE
You need to complete this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

In the following examples, libraryName should be POSZRTCS for the RTCS subsystem program library or POSZLINK for the RTCS product program library, and zzzzzz is the VOLSER for the library. The following command examples assume that the production libraries were allocated with the DSNAMES indicated in the sample JCL in Figure 3 on page 44 and Figure 4 on page 46. If you used different DSNAMES, change the names in the commands accordingly.

To temporarily APF-authorize these libraries

Issue one of the following commands to add each library to the APF list:

- If the library is not SMS-managed (recommended)
  
  SETPROG APF,ADD,LIBRARY=SYS2.RTCS.libraryName,VOLUME=zzzzzz

- If the library is SMS-managed
  
  SETPROG APF,ADD,LIBRARY=SYS2.RTCS.libraryName,SMS

To permanently APF-authorize these libraries

1. Add the library’s DSNAMe to the PROGxx member of SYS1.PARMLIB (or the z/OS image-specific Logical PARMLIB data set) that is appropriate for your site.

2. IPL the z/OS image or issue the SET PROG=xx z/OS operator command, which causes z/OS to rebuild the list of APF-authorized data sets in storage.

Allocating the RTCS system registry

Allocate a VSAM LDS cluster for the RTCS system registry, either by allocating

- one for each z/OS image if you are not going to share the registry
- one to be shared sysplex-wide by all members of the sysplex
Perform the following steps to allocate the RTCS system registry:

1. Select a VSAM LDS cluster DSNAME for the REGISTRY data set. Its low-level index should be REGISTRY, but this is not required. If its DSNAME does not follow the same naming convention that you have already used for the POSZRTCS, POSZLINK, and POSZHTML data sets, or its low-level index is not REGISTRY, you will have to specify the name of the LDS cluster in the SREGVLDS option of the RTCS Initialization member of Logical PARMLIB.

To allocate a new VSAM LDS to back the RTCS system registry, you can customize the sample JCL shown in Figure 6. The text that should be customized is shown in bold. This sample JCL is contained in member OSZIJ80 in the TOSZCNTL target library.

2. Determine the amount of space, expressed in terms of the number of 4K blocks, to allocate for the VSAM LDS.

NOTE

- BMC recommends that the RTCS system registry data set not be under the control of the Storage Management System (SMS). If the registry data set must be under SMS control, be sure to specify storage class and management class values that allow the data set to be kept indefinitely. It is very important that the register data set not be deleted.

- If you choose to manage the RTCS registry data set using SMS, then you must wait until SMS has fully initialized before starting RTCS.

- This VSAM LDS cluster must be allocated for use by the RTCS subsystem to back the primary RTCS registry partition using the z/OS Data-In-Virtual (DIV) service. Because it is a VSAM cluster and will be dynamically allocated by the RTCS subsystem address space using only its DSNAME, this library must be cataloged.
The minimum amount of space you should allocate for the RTCS system registry VSAM LDS is 17,472 blocks, which is 68.25 MB, and requires 98 cylinders on a 3390.

Although less space may actually work, BMC does not recommend allocating any amount less than 17,472 blocks (or 98 cylinders). The RTCS system registry is not likely to require more than 260.25 MB of space, even in very large configurations, so an allocation greater than 371 cylinders is not needed. For most sites, 98 cylinders is adequate for future use.

Allocate the library using the information you have gathered.

### Allocating the product authorization table library

Allocate a new, empty, one-track PDS or PDSE for the BMC product authorization table library (POSZPSWD) data set following your selected RTCS production library data set naming convention.

**NOTE**

You need to completed this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

To allocate a new product authorization table library, you can customize the sample JCL shown in Figure 7. The text that should be customized is shown in bold. This sample JCL is contained in member OSZINJ84 in the TOSZCNTL target library.

**Figure 7  Sample JCL to create RTCS product authorization table library**

```plaintext
//OSZINJ84 JOB (acct),'ALLOC PASSWORD AUTH',
//     MSGLEVEL=(1,1),CLASS=A,TIME=1
//*
//*    acct     JOB statement accounting information
//*    yyyyyy   VOLSER on which to allocate .POSZPSWD
//**
//PSWDUV EXEC PGM=IEFBR14
//*
//*   Allocate a New BMC Product Authorization Table
//*    Library
//**
//SYSLIB DD DSN=SYS2.RTCS.POSZPSWD,
//     DISP=(NEW,CATLG,DELETE),
//     SPACE=(CYL,(1,1,1)),DSORG=PO,
//     BLKSIZE=23476,RECFM=U,
//     UNIT=3390,VOL=SER=yyyyyy,
//     DSNTYPE=LIBRARY
//*
```
Ensuring sufficient common area data spaces

The RTCS subsystem requires two SCOPE=COMMON data spaces.

**NOTE**

Complete this task for each target z/OS image.

The MAXCAD parameter in member IEASYSxx of SYS1.PARMLIB or a z/OS system image Logical PARMLIB data set specifies the maximum number of SCOPE=COMMON data spaces that will be available for the life of the IPL.

If you have difficulty starting RTCS because no SCOPE=COMMON data spaces are available, increase the value of the MAXCAD parameter, and IPL the z/OS image.

Ensuring sufficient system linkage indexes

The RTCS subsystem requires only one system LX.

**NOTE**

Complete this task for each target z/OS image.

The NSYSLX parameter in member IEASYSxx of SYS1.PARMLIB or a z/OS system image Logical PARMLIB data set specifies the number of additional system LXs that are to be reserved for use by authorized applications and products on this system image.

If you have difficulty starting RTCS because no system LXs are available, increase the value of the NSYSLX parameter, and reIPL the z/OS image.

Adding RTCS programs to the z/OS PPT

The RTCS Initiator, the RTCS subsystem, and the RTCS Generalized Server require that they be executed in a started task address space with certain z/OS-assigned special attributes. Definitions for the job step program used in these PROCs in the z/OS Program properties table (PPT) cause the required attributes to be assigned by z/OS. Before RTCS can be started, the z/OS PPT must be updated to include the entries required for RTCS.
Selecting an SSID for the RTCS subsystem control address space

**NOTE**

You need to completed this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

Use one of the following methods to add RTCS programs to the PPT by using member OSZSCHED of the SMP/E TOSZCNTL target library.

- Update an existing SCHEDxx member by performing the following tasks:
  1. Merge the contents of member OSZSCHED into an existing SCHEDxx member.
  2. Refresh the PPT by issuing the following z/OS operator command:

     ```
     SET SCH=(xx)
     ```

     The variable xx is the suffix of any existing SCH member.

- Create a new SCHEDyy member by performing the following tasks:
  1. Copy the sample OSZSCHED member into SYS1.PARMLIB (or a logical parameter library data set), and name it SCHEDyy.
  2. Update the appropriate IEASYSnn member to specify this new SCHEDyy member and any existing SCH members.
  3. Refresh the PPT to have the additional entries in the new member become effective by issuing the following z/OS operator command:

     ```
     SET SCH=(xx,yy)
     ```

     The variable xx is the suffix of any existing SCH member (named SCHEDyy).

**WARNING**

The contents of all SCH member suffixes in a SET command replace any existing entries in storage. Therefore, you must include any existing PPT member suffixes in the list in addition to any new suffixes that you add.

Selecting an SSID for the RTCS subsystem control address space

Select an available, 4-character z/OS subsystem ID (SSID) for use by the RTCS subsystem control address space.
BMC recommends using RTCS for the SSID, but you can select any sequence of four characters.

**NOTE**
You need to completed this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

### Updating the subsystem name table

*(optional)* To guarantee the availability of an SSID for exclusive use by the RTCS subsystem, update the z/OS subsystem name (SSN) table.

**NOTE**
You need to completed this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

1. Add the SSID that was selected in step on page 52 to the IEFSSNxx member in SYS1.PARMLIB (or the appropriate Logical PARMLIB data set for the target z/OS image).

   For example, to add an entry with the SSID of RTCS, add the following statement to the appropriate IEFSSNxx member of SYS1.PARMLIB:

   ```
   SUBSYS SUBNAME(RTCS)
   ```

2. Update the z/OS Subsystem Name Table by either performing an IPL of the target z/OS image or by using the following z/OS operator command:

   ```
   SETSSI ADD,SUBNAME=RTCS
   ```

### Creating and customizing the RTCS Initiator started task PROC

Create and customize the RTCS Initiator started task PROC by performing the following steps.
Select a name for the RTCS Initiator started task PROC. This name will be used in an z/OS START command to start the RTCS Initiator address space, preferably under the Master Subsystem (MSTR). BMC recommends using the name OSZINIT, but this is not required.

Copy member OSZ$INIT from the SMP/E TOSZCNTL target library to SYS1.PROCLIB or a JES-managed started task procedure library using the member name that you selected in step 1.

BMC recommends that you start RTCS under the Master Subsystem (by using SUB=MSTR on the START command). But this is not absolutely required.

If you start RTCS under the Master Subsystem and you use an unmodified MSTJCL00, you must use SYS1.PROCLIB specifically for those started tasks that are started under the Master Subsystem. Otherwise, the PROC does not need to be created in SYS1.PROCLIB, and you can use any started task procedure library that is defined in the IEFPDSI DD statement concatenation in the Master Scheduler JCL (member MSTJCLxx of SYS1.PARMLIB).

If you start RTCS under your primary JES, you can place the PROC in any JES-managed started task PROC LIB.

Modify the STEPLIB DD statement in the new PROC to specify the DSNAME of the RTCS subsystem program library (POSZRTCS). For example, you can use the PDSE that was created in Figure 3 on page 44.
Creating the RTCS subsystem started task PROC

Create the RTCS subsystem started task PROC by performing the following steps.

1. Select a name to be used for the RTCS subsystem started task PROC.

   BMC recommends using the name OSZRTCS, but this is not required. This name will be used in an z/OS START command that the RTCS Initiator (OSZSIRIS) issues to start the RTCS subsystem address space.

2. Copy member OSZ$RTCS from the SMP/E TOSZCNTL target library to SYS1.PROCLIB or a JES-managed started task procedure library using the member name that you selected in step 1.

   If you start RTCS under the Master Subsystem and you use an unmodified MSTJCL00, you must use SYS1.PROCLIB specifically for those started tasks that are started under the Master Subsystem. Otherwise, the PROC does not need to be created in SYS1.PROCLIB, and you can use any started task procedure library that is defined in the IEFPDSI DD statement concatenation in the Master Scheduler JCL (member MSTJCLxx of SYS1.PARMLIB).

   If you start RTCS under your primary JES, you can place the PROC in any JES-managed started task PROCLIB.

Creating the RTCS Generalized Server started task PROCs

The RTCS Generalized Server PROC must reside in some data set (but not necessarily in the same data set) that is allocated to each of the following PROCLIB concatenations:

- Master Subsystem procedure library
- JES-managed started task procedure library
Creating the RTCS Generalized Server started task PROCs

The PROC might reside in a single data set that appears in both concatenations, depending on your site’s PROCLIB configuration. If the PROC is to reside in two separate data sets (one for each PROCLIB concatenation), then each PROC must be identical and must have the same member (PROC) name.

**NOTE**

- You need to complete this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.
- A model RTCS Generalized Server started task PROC (OSZEXEC) is provided in Figure 16 on page 105.

Perform the following steps to create the RTCS Generalized Server started task PROCs:

1. Select a name to be used for the RTCS Generalized Server started task PROCs.
   - BMC recommends using the name OSZEXEC, but this is not required.
   - If this PROC needs to be placed in two concatenations, you must use the same name for both PROCs. This name will be used in START commands that will be issued internally by RTCS.

2. Copy member OSZ$EXEC from the SMP/E TOSZCNTL target library to any valid Master Subsystem PROCLIB and use the member name that you selected in step 1.
   - This PROC must exist in a Master Subsystem PROCLIB, because RTCS Generalized Server address spaces are sometimes started by RTCS with a specification of SUB=MSTR.
   - If you use an unmodified MSTJCL00, you must use SYS1.PROCLIB specifically for those started tasks that are started under the Master Subsystem. Otherwise, the PROC does not need to be created in SYS1.PROCLIB, and you can use any started task procedure library that is defined in the IEFPSI DD statement concatenation in the Master Scheduler JCL (member MSTJCLxx of SYS1.PARMLIB).

3. Copy member OSZ$EXEC from the SMP/E TOSZCNTL target library to any valid JES-managed started task PROCLIB and use the member name that you selected in step 1.
   - This PROC must exist in a JES-managed PROCLIB, because RTCS Generalized Server address spaces are usually started by RTCS without specifying SUB=MSTR. In this case, the RTCS Generalized Server runs under the primary JES, and its PROC is expanded by using the usual JES-managed started task PROCLIBs.
Establishing ESM user IDs for RTCS address spaces

Establish ESM user IDs for RTCS address spaces using the procedure that is correct for the ESM that is executing on the z/OS image.

**NOTE**

You need to completed this task only once per sysplex if you are sharing data sets. Otherwise, this task needs to be completed for each target z/OS image.

Appropriately defined, valid ESM user IDs must be assigned to the three RTCS started task address spaces (RTCS Initiator, RTCS subsystem, and RTCS Generalized Server) by z/OS and the ESM when these address spaces are started. The procedure to assign these user IDs depends on the specific ESM that is executing on the z/OS image. These user IDs must be given access to the production product libraries that will be accessed by the address spaces that are running under the assigned user IDs.

Figure 8 (IBM RACF®), Figure 9 on page 60 (CA ACF2), and Figure 10 on page 63 (CA Top Secret) provide examples that you can customize. Use the example that applies to the ESM that is executing on the z/OS image. Because of the complex and sometimes subtle interaction between the data set naming conventions in place and existing profiles, permits, or rules, the following examples might require extensive modifications for use by your site. The sample JCL is contained in member OSZ$RACF, OSZ$ACF2, or OSZ$TSS in the TOSZCNTL target library.

**Figure 8** RACF example (part 1 of 4)

```bash
/* These commands are intended as EXAMPLES ONLY, to be edited by the */
/* customer system programmer or security administrator for each MVS    */
/* image or shared ESM data base, as required by your configuration. */

/* The commands provided may not work at your site exactly as they    */
/* have been illustrated here, or they may not even be necessary. */

/* Define data set profiles for the PDSE libraries & System Registry. */
/* We define them initially with UACC(ALTER) so that the current ID   */
/* does not have to have the SPECIAL attribute to issue the PERMIT    */
/* commands that immediately follow. We will alter the UACC in the    */
/* data set profiles after ALTER access by the current user is set. */

ADDSD 'SYS2.RTCS.POSZRTCS' -
    GENERIC UACC(ALTER)
ADDSD 'SYS2.RTCS.POSZLINK' -
    GENERIC UACC(ALTER)
ADDSD 'SYS2.RTCS.POSZHTML' -
    GENERIC UACC(ALTER)
ADDSD 'SYS2.RTCS.POSZPSWD' -
    GENERIC UACC(ALTER)
ADDSD 'SYS2.RTCS.REGISTRY' -
    GENERIC UACC(ALTER)
```
Establishing ESM user IDs for RTCS address spaces

Figure 8  RACF example (part 2 of 4)

/* Ensure that the system programmer executing the RTCS configuration */
/* JOBS has ALTER access to the RTCS Subsystem and Product Libraries. */
/* the Hypertext Document Library, and the RTCS System Registry VLDS. */
/* The absence of OWNER(id) does not grant such access unless the "*" */
/* UserID has the SPECIAL attribute, anyway. The point of all of this */
/* convolution is to obviate the requirement that a system programmer */
/* running these JOBS have the SPECIAL attribute in the first place. */
PERMIT 'SYS2.RTCS.POSZRTCS' -
    GENERIC ID(*)    ACCESS(ALTER)
PERMIT 'SYS2.RTCS.POSZLINK' -
    GENERIC ID(*)    ACCESS(ALTER)
PERMIT 'SYS2.RTCS.POSZHTML' -
    GENERIC ID(*)    ACCESS(ALTER)
PERMIT 'SYS2.RTCS.POSZPSWD' -
    GENERIC ID(*)    ACCESS(ALTER)
PERMIT 'SYS2.RTCS.REGISTRY' -
    GENERIC ID(*)    ACCESS(ALTER)

/* Update data set profiles for the PDSE libraries & System Registry. */
/* We set the minimum UACC needed by a typical, non-strict, customer. */
ALTDSD 'SYS2.RTCS.POSZRTCS' -
    GENERIC UACC(NONE)
ALTDSD 'SYS2.RTCS.POSZLINK' -
    GENERIC UACC(READ)
ALTDSD 'SYS2.RTCS.POSZHTML' -
    GENERIC UACC(READ)
ALTDSD 'SYS2.RTCS.POSZPSWD' -
    GENERIC UACC(READ)
ALTDSD 'SYS2.RTCS.REGISTRY' -
    GENERIC UACC(NONE)

/* Define an STC UserID for use by RTCS Initiator AND RTCS Subsystem. */
/* In particular, take care to properly specify the OMVS parameters. */
ADDUSER OSZRTCS -
    NAME('RTCS Subsystem') -
    NOPASSWD    -
    DFLTGRP(SYS1) -
    UACC(READ)    -
    LANGUAGE(PRIMARY(ENU)) -
    OMVS( UID(10010) HOME('/home') PROGRAM('/bin/sh') )

/* Define an STC UserID for use by RTCS Generalized Server (products). */
/* In particular, take care to properly specify the OMVS parameters. */
ADDUSER OSZEXEC -
    NAME('RTCS General Server') -
    NOPASSWD    -
    DFLTGRP(SYS1) -
    UACC(READ)    -
    LANGUAGE(PRIMARY(ENU)) -
    OMVS( UID(10011) HOME('/home') PROGRAM('/bin/sh') )

/* Connect these two new STC UserIDs to the STC PROCs used by them. */

BMC Runtime Component System Configuration and Administration Guide
Establishing ESM user IDs for RTCS address spaces

Chapter 3 Configuring RTCS

59

Figure 8  RACF example (part 3 of 4)

SETROPTS GENERIC(STARTED) /* We assume this is in effect already. */
RDEFINE STARTED OSZINIT.* STDATA( USER(OSZRTCS) )
RDEFINE STARTED OSZRTCS.* STDATA( USER(OSZRTCS) )
RDEFINE STARTED OSZEXEC.* STDATA( USER(OSZEXEC) )
SETROPTS RACLIST(STARTED) REFRESH

/* Allow READ access by the RTCS Subsystem to RTCS Subsystem Library. */
/* the RTCS Product Library, and Hypertext Document Library PDSEs. */
PERMIT 'SYS2.RTCS.POSZRTCS' -
   GENERIC ID(OSZRTCS) ACCESS(READ)
PERMIT 'SYS2.RTCS.POSZLINK' -
   GENERIC ID(OSZRTCS) ACCESS(READ)
PERMIT 'SYS2.RTCS.POSZHTML' -
   GENERIC ID(OSZRTCS) ACCESS(READ)

/* Allow READ access by the RTCS Subsystem to the */
/* Product Authorization (License) Table Library. */
PERMIT 'SYS2.RTCS.POSZPSWD' -
   GENERIC ID(OSZRTCS) ACCESS(READ)

/* Allow UPDATE access by RTCS Subsystem to the RTCS System Registry. */
PERMIT 'SYS2.RTCS.REGISTRY' -
   GENERIC ID(OSZRTCS) ACCESS(UPDATE)

/* Allow READ access by RTCS Client Product Generalized Server STCs */
/* to the RTCS Product Library and Hypertext Document Library PDSEs. */
PERMIT 'SYS2.RTCS.POSZLINK' -
   GENERIC ID(OSZEXEC) ACCESS(READ)
PERMIT 'SYS2.RTCS.POSZHTML' -
   GENERIC ID(OSZEXEC) ACCESS(READ)

/* Allow SYSLOG access by RTCS Client Product Generalized Server STCs */
RDEFINE JESSPOOL -
   &RACLNDE.+MASTER+.SYSLOG.*.*.? UACC(READ)
PERMIT &RACLNDE.+MASTER+.SYSLOG.*.*.? -
   CLASS(JESSPOOL) ID(OSZEXEC) ACCESS(READ)

/* Identify applications defined by RTCS and RTCS client products */
/* and allow access by all users in general. */
RDEFINE APPL (RTCSHTTP) UACC(READ)
RDEFINE APPL (EXPLORER) UACC(READ)

/* Define profiles for modules defined and loaded by Dynamic LPA ADD */
/* Permit RTCS Initiator and RTCS Subsystem to add OSZ modules to LPA */
SETROPTS GENERIC(FACILITY)
RDEFINE FACILITY CSVDYLPA.ADD.OSZ* UACC(NONE)
PERMIT CSVDYLPA.ADD.OSZ* -
   CLASS(FACILITY) ID(OSZRTCS) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
Establishing ESM user IDs for RTCS address spaces

Figure 8  RACF example (part 4 of 4)

/* Allow access to certain TCP/IP resources by the RTCS */
/* Subsystem and the RTCS HTTP Server which runs in the */
/* RTCS Generalized Server address space. */

SETROPTS CLASSACT(SERVAUTH)
RDEFINE SERVAUTH EZB.STACKACCESS.* UACC(READ)
RDEFINE SERVAUTH EZB.PORTACCESS.* UACC(READ)
RDEFINE SERVAUTH EZB.NETACCESS.* UACC(READ)
RDEFINE SERVAUTH EZB.TN3270.* UACC(READ)
PERMIT EZB.STACKACCESS.* -
   CLASS(SERVAUTH) ID(OSZRTCS) ACCESS(READ)
PERMIT EZB.PORTACCESS.* -
   CLASS(SERVAUTH) ID(OSZRTCS) ACCESS(READ)
PERMIT EZB.NETACCESS.* -
   CLASS(SERVAUTH) ID(OSZRTCS) ACCESS(READ)
PERMIT EZB.TN3270.* -
   CLASS(SERVAUTH) ID(OSZRTCS) ACCESS(READ)
PERMIT EZB.STACKACCESS.* -
   CLASS(SERVAUTH) ID(OSZEXEC) ACCESS(READ)
PERMIT EZB.PORTACCESS.* -
   CLASS(SERVAUTH) ID(OSZEXEC) ACCESS(READ)
PERMIT EZB.NETACCESS.* -
   CLASS(SERVAUTH) ID(OSZEXEC) ACCESS(READ)
PERMIT EZB.TN3270.* -
   CLASS(SERVAUTH) ID(OSZEXEC) ACCESS(READ)
SETROPTS RACLIST(SERVAUTH)
SETROPTS RACLIST(SERVAUTH) REFRESH

Figure 9  CA ACF2 example (part 1 of 4)

ACF

* These commands are intended as EXAMPLES ONLY, to be edited by the *
* customer system programmer or security administrator for each MVS *
* image or shared ESM data base, as required by your configuration.

* The commands provided may not work at your site exactly as they *
* have been illustrated here, or they may not even be necessary.

* Define a started task LOGONID for use by the RTCS Initiator. *
* Take care to customize this properly for your ACF2 configuration. *
* In particular, take care to properly specify the OMVS parameters.

SET LID
INSERT USING(ACFSTCID) OSZINIT -
   STC NOJOB NOTSO -
   NAME(RTCS Initiator) -
   RESTRICT -
   GROUP(SYS1)
SET PROFILE(USER) DIV(OMVS)
INSERT OSZINIT UID(07827) HOME(/) OMVSPGM(/BIN/SH)
SET PROFILE(GROUP) DIV(OMVS)
INSERT OSZINIT GID(07827)

* Define a started task LOGONID for use by the RTCS Subsystem. *
* Take care to customize this properly for your ACF2 configuration. *
* In particular, take care to properly specify the OMVS parameters.
---

**Figure 9 CA ACF2 example (part 2 of 4)**

<table>
<thead>
<tr>
<th>SET LID</th>
<th>INSERT USING(ACFSTCID) OSZRTCS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>STC NOJOB NOTSO</td>
</tr>
<tr>
<td></td>
<td>NAME(RTCS Subsystem)</td>
</tr>
<tr>
<td></td>
<td>RESTRICT</td>
</tr>
<tr>
<td></td>
<td>GROUP(SYS1)</td>
</tr>
<tr>
<td>SET PROFILE(USER) DIV(OMVS)</td>
<td>INSERT OSZRTCS UID(17827) HOME(/) OMVSPGM(/BIN/SH)</td>
</tr>
<tr>
<td>SET PROFILE(GROUP) DIV(OMVS)</td>
<td>INSERT OSZRTCS GID(17827)</td>
</tr>
</tbody>
</table>

* Define an STC ACID for use by RTCS Generalized Server (products).
* Take care to customize this properly for your ACF2 configuration.
* In particular, take care to properly specify the OMVS parameters.

SET LID

| INSERT USING(ACFSTCID) OSZEXEC |
|---|---|
|    | STC NOJOB NOTSO |
|    | NAME(RTCS General Server) |
|    | MUSASS |
|    | RESTRICT |
|    | GROUP(SYS1) |
| SET PROFILE(USER) DIV(OMVS) | INSERT OSZEXEC UID(27827) HOME(/) OMVSPGM(/BIN/SH) |
| SET PROFILE(GROUP) DIV(OMVS) | INSERT OSZEXEC GID(27827) |

* The following command needs to be entered at the operator console:
*   F ACF2,REBUILD(USR),CLASS(P)

* The following items document potential additions to existing data set rules that may be necessary for RTCS and RTCS-based products.

* The RTCS Initiator and the RTCS Subsystem need READ access to the RTCS Subsystem Library: SYS2.RTCS.POSZRTCS
*   * SYS2.RTCS.POSZRTCS UID(OSZINIT) R(A)
*   * SYS2.RTCS.POSZRTCS UID(OSZRTCS) R(A)

* The RTCS Subsystem needs READ access to the RTCS Product (License) Authorization Table Library: SYS2.RTCS.POSZPSWD
*   * SYS2.RTCS.POSZPSWD UID(OSZINIT) R(A)
*   * SYS2.RTCS.POSZPSWD UID(OSZRTCS) R(A)

* All three RTCS Started Task LOGONIDs need READ access to the RTCS Product Library: SYS2.RTCS.POSZLINK
*   * SYS2.RTCS.POSZLINK UID(OSZINIT) R(A)
*   * SYS2.RTCS.POSZLINK UID(OSZRTCS) R(A)
*   * SYS2.RTCS.POSZLINK UID(OSZEXEC) R(A)
Establishing ESM user IDs for RTCS address spaces

* All three RTCS Started Task LOGONIDs need READ access to the Hypertext Doc. Library: SYS2.RTCS.POSZHTML
  * SYS2.RTCS.POSZHTML UID(OSZINIT) R(A)
  * SYS2.RTCS.POSZHTML UID(OSZRTCS) R(A)
  * SYS2.RTCS.POSZHTML UID(OSZEXEC) R(A)

* The RTCS Initiator and the RTCS Subsystem need ALLOC access to the RTCS System Registry: SYS2.RTCS.REGISTRY
  * SYS2.RTCS.REGISTRY UID(OSZINIT) R(A) W(A) A(A)
  * SYS2.RTCS.REGISTRY UID(OSZRTCS) R(A) W(A) A(A)

* The RTCS Client Product Generalized Server Started Task LOGONID needs READ access to the standard JESSPOOL SYSLOG SYSOUT data set:
  * sysname.-.SYSLOG UID(OSZEXEC) SERVICE(READ) ALLOW

* Identify APPLlications defined by RTCS and RTCS client products. Allow access by all users in general. NOTE: By default, ACF2 maps SAF Resource CLASS 'APPL' to ACF2 Resource Rule TYPE(SAF).
  * SET RESOURCE(SAF)
  * COMPILE * LIST STORE
  * $KEY(RTCSHTTP) TYPE(SAF)
  * UID(-) ALLOW
  * END
  * COMPILE * LIST STORE
  * $KEY(EXPLORER) TYPE(SAF)
  * UID(-) ALLOW
  * END

* Permit RTCS Initiator and RTCS Subsystem to add OSZ modules to LPA using Dynamic LPA faility. NOTE: The following assumes that ACF2 maps SAF Resource CLASS 'FACILITY' to ACF2 Resource Rule TYPE(FAC).
  * SET RESOURCE(FAC)
  * COMPILE * LIST STORE
  * $KEY(CSVDYLPA) TYPE(FAC)
  * ADD.OSZ- UID(OSZRTCS) ALLOW
  * END

---

Figure 9  CA ACF2 example (part 3 of 4)
Establishing ESM user IDs for RTCS address spaces

Chapter 3 Configuring RTCS

* Allow access to certain TCP/IP resources by the RTCS Subsystem and the RTCS HTTP Server which runs in the RTCS Generalized Server address space.

* SET RESOURCE(SER)
  * COMPILE * LIST STORE
  * $KEY(EZB) TYPE(SER)
    * NETACCESS.- UID(OSZRTCS) SERVICE(READ) ALLOW
    * NETACCESS.- UID(OSZEXEC) SERVICE(READ) ALLOW
    * STACKACCESS.- UID(OSZRTCS) SERVICE(READ) ALLOW
    * STACKACCESS.- UID(OSZEXEC) SERVICE(READ) ALLOW
    * PORTACCESS.- UID(OSZRTCS) SERVICE(READ) ALLOW
    * PORTACCESS.- UID(OSZEXEC) SERVICE(READ) ALLOW
    * TN3270.- UID(OSZRTCS) SERVICE(READ) ALLOW
    * TN3270.- UID(OSZEXEC) SERVICE(READ) ALLOW
  * END

* In order to use the SERVAUTH SAF resource CLASS, which is assumed to be mapped to ACF2 rule type SER in the example above, the rule set for rule type 'SER' must be resident.

* SET CONTROL(GSO)
  * INSERT CLASMAP.SERVAUTH RESOURCE(SERVAUTH) +
    * RSRCTYPE(SER) +
    * ENTITYLN(64)
  * CHANGE INFODIR REP TYPES(R-RSER)

* The following commands need to be entered at the operator console:

* F ACF2, REFRESH(GSO)
* F ACF2, REBUILD(SER)

END

Figure 9  CA ACF2 example (part 4 of 4)

* These commands are intended as EXAMPLES ONLY, to be edited by the customer system programmer or security administrator for each MVS image or shared ESM data base, as required by your configuration. The commands provided may not work at your site exactly as they have been illustrated here, or they may not even be necessary. Select an available Top Secret Facility Matrix entry for use by the RTCS Subsystem and Generalized Server address spaces. It is renamed to FACILITY(RTCS), and required attributes established. In this EXAMPLE, we illustrate that the USER69 Facility Matrix entry is not already in use and has been selected for renaming. Note that a Facility Matrix entry you may already have defined (such as BBI3) for the MAINVIEW CAS and PASs can also be used.
### Establishing ESM user IDs for RTCS address spaces

**Figure 10 CA Top Secret example (part 2 of 4)**

```plaintext
TSS MODIFY(FACILITY(USER69=NAME=RTCS))
TSS MODIFY(FACILITY(RTCS=PGM=OSZ))
TSS MODIFY(FACILITY(RTCS=KEY=0))
TSS MODIFY(FACILITY(RTCS=MULTIUSER))
TSS MODIFY(FACILITY(RTCS=NOABEND))
TSS MODIFY(FACILITY(RTCS=ACTIVE))
TSS MODIFY(FACILITY(RTCS=SIGN(M))
TSS MODIFY(FACILITY(RTCS=NONPWR))
TSS MODIFY(FACILITY(RTCS=NOTSOC))
TSS MODIFY(FACILITY(RTCS=NEOPROMPT))
TSS MODIFY(FACILITY(RTCS=RES))
TSS MODIFY(FACILITY(RTCS=MAXUSER=500))
TSS MODIFY(FACILITY(RTCS=ASUBM))
TSS MODIFY(FACILITY(RTCS=MODE=FAIL))
TSS MODIFY(FACILITY(RTCS=WARNPW))
TSS MODIFY(FACILITY(RTCS=DORMPW))
TSS MODIFY(FACILITY(RTCS=NORMPW))
TSS MODIFY(FACILITY(RTCS=SHRPRF))
TSS MODIFY(FACILITY(RTCS=LUMSG))
TSS MODIFY(FACILITY(RTCS=STMSG))
TSS MODIFY(FACILITY(RTCS=TRACE))
TSS MODIFY(FACILITY(RTCS=LCFCMD))

/* This FACILITY Matrix Entry definition should be put in TSSPARMS */
/* to ensure that it is always defined, but MODIFY works to define */
/* the entry temporarily so that TSS does not have to be REINITed. */

/* Define an STC ACID for use by RTCS Initiator and RTCS Subsystem. */
/* Take care to customize this properly for your TSS configuration. */
TSS CREATE(OSZRTCS) +
    NAME('RTCS Subsystem') +
    FACILITY(STC) +
    TYPE(USER) +
    PASSWORD(NOPW,0) +
    MASTFAC(RTCS) +
    DEPT(STCPROCS)
TSS ADDTO(OSZRTCS) +
    UID(17827) HOME(/) +
    OMVSPGM(/BIN/SH) +
    DFLTGRP(OMVSGRP) +
    GROUP(OMVSGRP)

/* Define an STC ACID for use by RTCS Generalized Server (products).*/
/* Take care to customize this properly for your TSS configuration. */
TSS CREATE(OSZEXEC) +
    NAME('RTCS General Server') +
    FACILITY(STC) +
    TYPE(USER) +
    PASSWORD(NOPW,0) +
    MASTFAC(RTCS) +
    DEPT(STCPROCS)
TSS ADDTO(OSZEXEC) +
    UID(27827) HOME(/) +
    OMVSPGM(/BIN/SH) +
    DFLTGRP(OMVSGRP) +
    GROUP(OMVSGRP)
```
/* Connect these two new STC UserIDs to the STC PROCs used by them. */
TSS ADDTO(STC) PROCNAME(OSZINIT) ACID(OSZRTCS)
TSS ADDTO(STC) PROCNAME(OSZRTCS) ACID(OSZEXEC)
TSS ADDTO(STC) PROCNAME(OSZEXEC) ACID(OSZEXEC)

/* Allow READ access by the RTCS Subsystem to RTCS Subsystem Library, */
/* the RTCS Product Library, and Hypertext Document Library PDSEs. */
TSS PERMIT(OSZRTCS) DSN(SYS2.RTCS.POSZRTCS) ACC(READ)
TSS PERMIT(OSZRTCS) DSN(SYS2.RTCS.POSZLINK) ACC(READ)
TSS PERMIT(OSZRTCS) DSN(SYS2.RTCS.POSZHTML) ACC(READ)

/* Allow READ access by the RTCS Subsystem to the */
/* Product Authorization (License) Table Library. */
TSS PERMIT(OSZRTCS) DSN(SYS2.RTCS.POSZPSWD) ACC(READ)

/* Allow READ access by the RTCS Subsystem to the RTCS System Registry. */
TSS PERMIT(OSZRTCS) DSN(SYS2.RTCS.REGISTRY) ACC(READ)

/* Allow READ access by RTCS Client Product Generalized Server STCs */
/* to the RTCS Product Library and Hypertext Document Library PDSEs. */
TSS PERMIT(OSZEXEC) DSN(SYS2.RTCS.POSZLINK) ACC(READ)
TSS PERMIT(OSZEXEC) DSN(SYS2.RTCS.POSZHTML) ACC(READ)

/* Allow ALL access by RTCS Client Product Generalized Server STCs */
/* to the RTCS System Registry. */
TSS PERMIT(OSZEXEC) JESSPOOL(sysname.+MASTER+.SYSLOG) ACC(READ)

/* Establish minimal access for the PDSE libraries & System Registry. */
/* We set the minimum access needed by a typical non-strict customer. */
TSS PERMIT(ALL) DSN(SYS2.RTCS.POSZRTCS) ACC(NONE)
TSS PERMIT(ALL) DSN(SYS2.RTCS.POSZLINK) ACC(READ)
TSS PERMIT(ALL) DSN(SYS2.RTCS.POSZHTML) ACC(READ)
TSS PERMIT(ALL) DSN(SYS2.RTCS.POSZPSWD) ACC(READ)
TSS PERMIT(ALL) DSN(SYS2.RTCS.REGISTRY) ACC(NONE)

/* Identify applications defined by RTCS and RTCS client products */
/* and allow access by all users in general. */
TSS ADDTO(OSZRTCS) APPL(RTCSHTTP)
TSS ADDTO(OSZEXEC) APPL(EXPLORER)
TSS PERMIT(ALL) APPL(RTCSHTTP)
TSS PERMIT(ALL) APPL(EXPLORER)

/* Permit RTCS Initiator and RTCS Subsystem to add OSZ modules to */
/* LPA using Dynamic LPA facility. */
TSS PERMIT(OSZRTCS) IBMFAC(CSVDYLPA.ADD.OSZ) ACC(UPDATE)

/* Permit access to certain TCP/IP resources by the RTCS */
/* Subsystem and the RTCS HTTP Server which runs in the */
/* RTCS Generalized Server address space. */
Starting RTCS

Start RTCS by issuing the following MVS operator command:

\texttt{S OSZINIT,SUB=MSTR}

This command starts the RTCS Initiator, which internally starts the RTCS subsystem. The \texttt{OSZINIT} value is the name of the RTCS Initiator started task PROC that you selected in “Creating and customizing the RTCS Initiator started task PROC” on page 53.

\textbf{NOTE}

Ensure that the RTCS Initiator address space and the RTCS subsystem address space start successfully. The RTCS subsystem \textit{must} be successfully initialized before you can proceed with the next configuration step.

Adding the RTCS start command to SYS1.PARMLIB

Determine the appropriate COMMND\texttt{xx} member of SYS1.PARMLIB (or the logical PARMLIB data set for the \textit{z/OS} image), and modify the member to include \textit{one} of the following lines:

\begin{verbatim}
COM='S OSZINIT,SUB=MSTR'
COM='S OSZINIT'
\end{verbatim}

In the commands, the \textit{OSZINIT} value represents the name of the RTCS Initiator started task PROC.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
TSS PERMIT(OSZRTCS) & SERVAUTH(EZB.NETACCESS) & ACC(READ) \\
TSS PERMIT(OSZEXEC) & SERVAUTH(EZB.NETACCESS) & ACC(READ) \\
TSS PERMIT(OSZRTCS) & SERVAUTH(EZB.STACKACCESS) & ACC(READ) \\
TSS PERMIT(OSZEXEC) & SERVAUTH(EZB.STACKACCESS) & ACC(READ) \\
TSS PERMIT(OSZRTCS) & SERVAUTH(EZB.PORTACCESS) & ACC(READ) \\
TSS PERMIT(OSZEXEC) & SERVAUTH(EZB.PORTACCESS) & ACC(READ) \\
TSS PERMIT(OSZRTCS) & SERVAUTH(EZB.TN3270) & ACC(READ) \\
TSS PERMIT(OSZEXEC) & SERVAUTH(EZB.TN3270) & ACC(READ) \\
\hline
\end{tabular}
\caption{CA Top Secret example (part 4 of 4)}
\end{table}
NOTE
The RTCS subsystem should usually be started before the primary JES but after the ESM has initialized.

At some sites, this will not be possible because the ESM is started after or under the primary JES. In such cases, the RTCS subsystem should be started as soon as possible, even if it is to be started under the Master Subsystem (SUB=MSTR). If this is still not possible (because the RTCS subsystem needs to be started before the primary JES and run under the Master Subsystem so that it will be available for product address spaces that also start before the primary JES and which are also started under the Master Subsystem), after the ESM has completely initialized on the z/OS image, issue the following operator command:

F RTCS,REFRESH,SECURITY
This chapter contains information about administrative functions that you might need to perform at your site to ensure continued availability and proper functioning of the RTCS subsystem.

- Backing up the RTCS system registry: 70
- Recovering the RTCS system registry: 70
- Starting RTCS: 71
- Stopping RTCS: 73
- RTCS operator commands: 75
  - cscbName parameter: 75
  - command parameter: 76
  - operand parameter: 82
- RTCS Security Manager: 83
  - Global security parameters: 84
  - Customizing RTCS global security parameters: 86
  - Global class properties: 87
  - Customizing RTCS global class properties: 88
Backing up the RTCS system registry

Because the registry is a read/write data set that is updated during normal system and product operation, consider making a backup of the volume on which the registry resides.

You usually cannot make a backup of the VSAM LDS using data-set-oriented utilities while the RTCS subsystem has it allocated and open. You can, however, unallocate the registry temporarily by using the REGISTRY UNALLOCATE command (see “REGISTRY subcommand [registry] [operand]” on page 78). You can then use any data set-level VSAM cluster backup utility to make a backup of the registry VLDS.

If you do not regularly back up the DASD volume on which the registry resides and you make any site-specific customizations to the parameters that are held in the registry, retain the original input that you used to allocate the LDS and all of the subsequent customizations in case you need to reallocate and reinitialize the LDS in the event of a failure.

To make a backup copy of the registry

1. Issue the following command:
   
   F RTCS,REGISTRY UNEXPOSE

2. Issue the following command:
   
   F RTCS,REGISTRY UNALLOC

3. Run your backup job.

4. After the back up is completed, issue the following command:
   
   F RTCS,REGISTRY ACQUIRE

Recovering the RTCS system registry

You can re-create the RTCS and product configuration information in the registry by executing the RTCS Registry Import Utility (RIU) and using the INCLUDE statement to reapply your original customizations.

If the DASD volume on which the registry VSAM LDS for an z/OS image was allocated fails, use one of the following methods to recover or rebuild the VSAM LDS contents:
If a DASD volume-level backup is not available, allocate another VSAM LDS to back the registry. Reallocate or acquire the registry VLDS by using the \texttt{REGISTRY} operator command (see “\texttt{REGISTRY subcommand [registry] [operand]}” on page 78). If necessary, reinitialize the registry LDS by using the RTCS RIU, providing the original customized input.

If a DASD volume-level or data set-level backup is available, use the backup to restore the registry VSAM LDS. Reallocate or acquire the registry VLDS by using the \texttt{REGISTRY} operator command (see “\texttt{REGISTRY subcommand [registry] [operand]}” on page 78).

If the VSAM LDS that is used to back the registry on an z/OS image or its catalog becomes damaged or is unavailable for some reason when you IPL the z/OS image and the RTCS initiator is started, the RTCS subsystem will not initialize. As a result, products cannot execute. If this situation occurs, use one of the following methods to resolve the problem:

- Correct the problem and start RTCS.
- Restore the VSAM LDS and start RTCS.
- Allocate a new VSAM LDS, start RTCS, and initialize the new registry using any existing, retained customizations before starting any products.

## Starting RTCS

RTCS should be started as soon as possible during the IPL of a z/OS image. RTCS should be started either prior to or concurrently with the primary JES. RTCS does not use any JES services.

The RTCS subsystem must be available for products to execute. If the RTCS kernel has not been loaded and initialized, products that use RTCS services usually terminate abnormally (abend) during their initialization process. Products that use the RTCS Generalized Server (program \texttt{OSZEXEC[n]}) to obtain control in the proper execution key abend immediately, even before any product code is loaded. This action alerts you that the RTCS facilities are not available for the involved z/OS image.

To ensure that products will function properly, BMC Software recommends starting RTCS after the external security manager (ESM) has been started and successfully initialized. If you have a valid reason to start RTCS prior to ESM initialization, ensure that the ESM or your automation facility issues the following command after the ESM has completed initialization:

\texttt{F RTCS,REFRESH,SECURITY}
In the preceding command, RTCS represents the SSID of the RTCS subsystem.

Start RTCS by using the z/OS START command to initiate the RTCS Initiator address space. The actual START command depends on the member name of the RTCS Initiator started task PROC (usually located in SYS1.PROCLIB, but can be in a JES-managed started task PROCLIB) that was selected during z/OS image configuration.

For example, if the member name OSZINIT is selected and the OSZINIT PROC JCL has been properly customized to specify the correct POSZRTCS DSNAME to be used by RTCS, one of the following z/OS START commands is used to start the RTCS Initiator address space:

S OSZINIT, SUB=MSTR
S OSZINIT

The RTCS Initiator performs the following tasks before internally starting the RTCS subsystem address space:

- reads and processes the RTCS Initialization member from the Logical PARMLIB data set
- determines and validates the DSNAMEs of the production data sets to be used by the RTCS subsystem and the Generalized Server address spaces

A series of messages is issued to SYSLOG. Figure 11 on page 73 provides an example of a subset of the messages.
### Stopping RTCS

RTCS provides non-stop services to client applications, which requires that all critical control blocks be anchored in the RTCS address space. If the RTCS subsystem is taken down, its services will not be available and tasks or address spaces using RTCS will abnormally terminate.

BMC does not recommend that the RTCS subsystem address space be shut down for any reason. Nevertheless, installations have found it desirable to shut RTCS down in order to release the dynamic allocation on the system registry VSAM linear data set (LDS) so that it could be backed up or reallocated, or to change the DSNAMEs of the OSZLINK or OSZRTCS production libraries defined in the OSZINIT started task.

#### Figure 11  RTCS subsystem messages

<table>
<thead>
<tr>
<th>Message ID</th>
<th>Message Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSZ0050I</td>
<td>RTCS ALLOCATOR VER 01.02.00 PTF BP00076 INITIALIZATION</td>
</tr>
<tr>
<td>OSZ0069I</td>
<td>//TASKLIB DD DISP=SHR,DSN=SYS1.RTCS.POSZRTCS</td>
</tr>
<tr>
<td>OSZ0001I</td>
<td>RTCS SUBSYSTEM INITIALIZATION IN PROGRESS</td>
</tr>
<tr>
<td>OSZ0005I</td>
<td>RTCS SUBSYSTEM PC SERVICES NOW AVAILABLE</td>
</tr>
<tr>
<td>OSZ0029I</td>
<td>OBJECT MANAGEMENT SERVICES NOW AVAILABLE</td>
</tr>
<tr>
<td>OSZ0143I</td>
<td>NONBACKED DATA SPACE ACCESSED USING DSP BY RTCS REGISTRY 246</td>
</tr>
<tr>
<td></td>
<td>DATA SPACE CREATED AND INITIALIZED AND IS UNAVAILABLE FOR XCF SHARING</td>
</tr>
<tr>
<td></td>
<td>DSNAME=RTCS.SUBSYS.MEMORY.REGISTRY.NOSHAKE.RT22</td>
</tr>
<tr>
<td></td>
<td>VOL=SER= DDNAME= SCOPE=GLOBAL.EXCLUSIVE RS_TCB=006E5400</td>
</tr>
<tr>
<td></td>
<td>ALET=01FF0019/0000007400000000 TRACE=00000000/133C5000 DS_TCB=<strong>NONE</strong></td>
</tr>
<tr>
<td>OSZ0022I</td>
<td>RTCS SUBSYSTEM MEMORY REGISTRY IS NOW AVAILABLE</td>
</tr>
<tr>
<td>OSZ0069I</td>
<td>//SYS00001 DD DISP=OLD,DSN=SYS1.RTCS.REGISTRY</td>
</tr>
<tr>
<td>OSZ0147I</td>
<td>SYS00001 ALET=01FF001A DIV SERVICES ATTACHED TCB=006E4C58 251</td>
</tr>
<tr>
<td></td>
<td>BATCH UPDATE WINDOW: 000001 SEC 000006 SEC 000060 SEC, LIMIT: 4000</td>
</tr>
<tr>
<td></td>
<td>CURRENT VALUE: 000060 SEC (DIV SAVE NOT PENDING) COUNT: 0000</td>
</tr>
<tr>
<td></td>
<td>DIV SERVICES IS ONLY UPDATER WAITERS QUEUE CONTAINS NO TASKS</td>
</tr>
<tr>
<td>OSZ0149I</td>
<td>SYS00001 ALET=01FF001A DIV SAVE CPU=00.0216 ELAP=03.5958</td>
</tr>
<tr>
<td>OSZ0143I</td>
<td>VSAM LINEAR DATA SET ACCESSED USING DIV BY RTCS REGISTRY 253</td>
</tr>
<tr>
<td></td>
<td>HAS BEEN OPENED FOR UPDATE ACCESS, AND IS EXPOSED FOR XCF CONNECTIONS</td>
</tr>
<tr>
<td></td>
<td>DSNAME=SYS1.RTCS.REGISTRY</td>
</tr>
<tr>
<td></td>
<td>VOL=SER=DDNAME=SYS00001 SCOPE=GLOBAL,SHARED RS_TCB=006E5400</td>
</tr>
<tr>
<td></td>
<td>ALET=01FF001A/0000007400000000 TRACE=00000000/12F75000 DS_TCB=006E4C58</td>
</tr>
<tr>
<td>OSZ0079I</td>
<td>RTCS SUBSYSTEM SYSTEM REGISTRY IS NOW AVAILABLE</td>
</tr>
<tr>
<td>OSZ0432I</td>
<td>SECURITY ESMTYPE(RACF) SELECTED BY ESMTYPE(AUTO)</td>
</tr>
<tr>
<td>OSZ0443I</td>
<td>RTCS SECURITY - SAFSUBSYS(&amp;SSID ) APPLID(&amp;PRODUCT)</td>
</tr>
<tr>
<td>OSZ0444I</td>
<td>UNDEFINEDUSERINHERIT(ACCEPT) UNDEFINEDUSERSIGNON(REJECT)</td>
</tr>
<tr>
<td>OSZ0445I</td>
<td>GROUPINHERIT(ALWAYS) DEFAULTUSERID(&lt;BLANKS&gt;)</td>
</tr>
<tr>
<td>OSZ0446I</td>
<td>SECTRACE(NONE ) TESTMODE(OA)</td>
</tr>
<tr>
<td>OSZ0447I</td>
<td>ESM(RACF) RCVT ID(RCVT) VRMN(7730) FLGS(033D08000000000FF080)</td>
</tr>
<tr>
<td>OSZ1083I</td>
<td>RTCS LICENSE MANAGER ACTIVE</td>
</tr>
<tr>
<td>OSZ0039I</td>
<td>CONSOLE COMMAND INTERFACE AVAILABLE FOR OSZRTCS CSCB=RTCS</td>
</tr>
<tr>
<td>OSZ0030I</td>
<td>RTCS SUBSYSTEM (RTCS) IS NOW AVAILABLE ASID=001D OSZA=13175000</td>
</tr>
<tr>
<td>OSZ13176400AV</td>
<td>ASVT=025DE040</td>
</tr>
</tbody>
</table>
To stop RTCS when no client products are running

If no client product address spaces are running, use the standard STOP command to stop RTCS.

stop cscbName

cscbName is the name of the Command Scheduling Control Block (CSCB) that was built by z/OS. For the RTCS subsystem address space, the CSCB name is the RTCS z/OS Subsystem ID (SSID). For detailed information about the cscbName variable, see “RTCS operator commands” on page 75.

To stop RTCS when client products are running

If any client product address space is using RTCS services, the standard STOP command will not stop the RTCS subsystem address space. The STOP command first checks for any active client product address spaces, and if any are found, they are listed by using message OSZ106E and the STOP command is ignored.

Take the following steps to stop RTCS when client products are running:

1. Stop all MainView address spaces, including PAS, CAS, and TAS (Alternate Access) address spaces.

   If a MainView address space fails to shutdown, see the product’s documentation for additional information.

   **NOTE**

   Failure to stop all of these address spaces prior to stopping RTCS can result in additional problems.

2. Have all MainView TSO users logoff.

   If you cannot determine which TSO users have used MainView, stop all TSO users.

3. Issue a STOP command to RTCS, as described in “To stop RTCS when no client products are running.”
4 If RTCS does not respond with further OSZ0106E messages and does not terminate in a timely manner, you might have to use the FORCE command to stop RTCS.

**NOTE**

See the warning associated with using the FORCE command in the z/OS V1R12.0 MV5 System Commands book.

If a FORCE is deemed necessary issue the following commands (for detailed information about the `cscbName` variable, see “RTCS operator commands.”):

```
C cscbName
FORCE cscbName,ARM
```

RTCS terminates.

### RTCS operator commands

The RTCS subsystem address space and Generalized Server address spaces support a number of operator commands. Each command is entered by using the z/OS MODIFY command as follows:

```
F cscbName,command,operand1[,...,operandn]
```

**cscbName parameter**

In the modify command, `cscbName` is the name of the Command Scheduling Control Block (CSCB) that was built by z/OS for the address space to which the command is being directed. The name is derived as follows:

- For the RTCS subsystem address space, the CSCB name is the RTCS z/OS subsystem ID (SSID).
- For any Generalized Server address space started by RTCS by using the RTCS START,`productIdentifier` command, the CSCB name is the three-character product identifier.
- For a batch job, the CSCB name is the job name.
For any other started task that was not started by RTCS, the CSCB name is determined by z/OS and is usually one of the following:

— the name that is specified after the period following the started task PROC name (if any)

— the same as the PROC name

Figure 12 provides a sample subset of the operator messages that are issued when the Generalized Server begins execution. You can use these messages to determine the CSCB name for the address space. For example, in Figure 12, the CSCB name of the Generalized Server address space is ZSE, as shown in bold.

Figure 12  Generalized Server messages

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F RTCS,START,ZSE</td>
<td>START OSZEXEC.ZSE,KEY=4,P=ZSE9INIT,C=BMCPROD</td>
</tr>
<tr>
<td>OSZ0031I COMMAND WAS PROCESSED</td>
<td></td>
</tr>
<tr>
<td>$HASP100 OSZEXEC ON STCINRDR</td>
<td></td>
</tr>
<tr>
<td>$HASP373 OSZEXEC STARTED</td>
<td></td>
</tr>
<tr>
<td>OSZ0104I GENERALIZED SERVER STARTING 270</td>
<td>IN OSZEXEC (STARTED TASK) ASID=X'0026' CSCB=ZSE</td>
</tr>
<tr>
<td>OSZ0124I GENERALIZED SERVER PARAMETERS 271</td>
<td>PACKAGE=ZSE9INIT COMMAND=(NONE) CONTEXT=BMCPROD</td>
</tr>
<tr>
<td>OSZ0039I CONSOLE COMMAND INTERFACE AVAILABLE FOR OSZEXEC CSCB=ZSE</td>
<td></td>
</tr>
</tbody>
</table>

The following are the valid RTCS subsystem address space and Generalized Server address space operator commands.

**REFRESH,KERNEL**

Valid context: RTCS subsystem only

This command directs RTCS to determine whether an updated RTCS kernel package exists in the RTCS subsystem program library (.POSZRTCS) that was specified when the RTCS Initiator address space was started. If this is the case, the updated RTCS kernel is loaded into extended CSA to replace the older RTCS kernel.

Use this command to refresh the current kernel, if necessary, after applying maintenance to the kernel package in the production RTCS subsystem program library. If no updated kernel exists, no action is performed.
**REFRESH,LIBRARY**

Valid context: RTCS subsystem only

This command directs RTCS to determine whether updated DLL packages exist in the RTCS subsystem program library (.POSZRTCS) that was specified when the RTCS Initiator address space was started. If this is the case, updated DLL packages are loaded into extended CSA to replace the older DLLs.

Use this command to refresh the current DLL packages, if necessary, after applying maintenance to DLL packages in the RTCS subsystem program library. If no updated DLL packages exist, no action is performed.

**REFRESH,SECURITY**

Valid context: RTCS subsystem only

This command directs the RTCS Security Manager to reinitialize every Global Security Parameter from values that are stored in the RTCS system registry.

This command is normally used only when any Global Security Parameters have been changed by the RIU.

Under normal circumstances, you will not need to issue this command. Under specific circumstances, however, if RTCS must be started before the ESM has completely initialized, this command may need to be issued (typically by an automation product) as soon as the ESM has done so.

**SECTRACE,{NONE | ON | SIMPLE | EXTENDED | COMPLETE}**

Valid context: RTCS subsystem only

This command indicates the level of diagnostic trace messages for the RTCS Security Manager component to issue.

NONE indicates that no diagnostic trace messages are to be issued. SIMPLE, EXTENDED, and COMPLETE represent three different levels of message trace. ON is equivalent to COMPLETE.

**WARNMSG,{ON | OFF}**

Valid context: RTCS subsystem or Generalized Server address spaces

This command indicates whether RTCS functions that are executing in the address space should issue the normally-suppressed warning messages.
The default status for every address space is OFF.

**SET field=value**

Valid context: RTCS subsystem only

This command specifies the names of various resources. Table 2 describes the available statements.

**Table 2   SET statements**

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET OSZEXEC=proname</td>
<td>sets the started task PROC name of the Generalized Server address space</td>
</tr>
<tr>
<td></td>
<td>In the RTCS initialization member, this value was specified by the OSZEXEC-PROC option.</td>
</tr>
<tr>
<td>SET POSZRTCS=dsname</td>
<td>sets the DSNAMES of the RTCS subsystem program library</td>
</tr>
<tr>
<td></td>
<td>If a REFRESH command is subsequently entered, this data set is dynamically allocated to</td>
</tr>
<tr>
<td></td>
<td>search for updated RTCS program objects.</td>
</tr>
<tr>
<td>SET POSZLINK=dsname</td>
<td>sets the DSNAMES of the RTCS product program library</td>
</tr>
<tr>
<td></td>
<td>The RTCS product program library is the data set that is dynamically allocated by the</td>
</tr>
<tr>
<td></td>
<td>RTCS Generalized Server in order to load RTCS utility programs.</td>
</tr>
<tr>
<td>SET POSZHTML=dsname</td>
<td>sets the DSNAMES of the RTCS hypertext document library</td>
</tr>
<tr>
<td></td>
<td>This library contains variable-length record members used by RTCS utility programs.</td>
</tr>
<tr>
<td>SET POSZPSWD=dsname</td>
<td>sets the DSNAMES of the Product Authorization Table library</td>
</tr>
<tr>
<td></td>
<td>MainView products do not use this library, so normally you would not need to change this</td>
</tr>
<tr>
<td></td>
<td>data set name.</td>
</tr>
</tbody>
</table>

**REGISTRY subcommand [registry] [operand]**

Valid context: RTCS subsystem, Generalized Server, or MainView PAS

The **REGISTRY** command performs various functions to assist in the management of an RTCS system registry VSAM linear data set (VLDS).

**subcommand**

The *subcommand* variable indicates the action you want to perform. Table 3 on page 79 describes the valid *subcommand* keywords.
## Table 3  Subcommands (part 1 of 2)

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>STATUS</strong></td>
<td>The current status of the RTCS system registry. Valid for shared and nonshared RTCS system registries, RTCS memory registries, and RTCS product registries.</td>
</tr>
</tbody>
</table>
| **EXPOSE** | The RTCS system registry is exposed if all of the following criteria are met:  
- the registry is shared within a sysplex  
- the Local Owner is the current system  
- the RTCS system registry is not already exposed  

Exposing the RTCS system registry makes it available to be accessed from other, remote members of the registry’s XCF group. Valid for shared RTCS system registries and RTCS product registries. |
| **UNEXPOSE** | The RTCS system registry is unexposed (hidden) if all of the following criteria are met:  
- the registry is shared within a sysplex  
- the Local Owner is the current system  
- the RTCS system registry is already exposed  

Unexposing the RTCS system registry makes it unavailable for access from other, remote members of the registry’s XCF group. Valid for shared RTCS system registries and RTCS product registries. |
| **UNALLOCATE** | If the RTCS system registry is currently allocated to this XCF member:  
- the registry is drained  
- pending updates are hardened in the VLDS  
- the VLDS is dynamically unallocated  

The registry VLDS can be unallocated, say for DASD management or VSAM cluster backup purposes, then reallocated. While the registry VLDS remains unallocated, its DSNAME can be altered. Valid for shared and nonshared RTCS system registries and RTCS product registries. |
| **TRANSFER operand** | Transfers ownership of the RTCS system registry. For more information, see “operand parameter” on page 82. |
### Table 3  Subcommands (part 2 of 2)

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Description</th>
</tr>
</thead>
</table>
| RELEASE     | If the RTCS system registry is shared within a sysplex and the Local Owner is the current system  
  ■ the registry is drained  
  ■ pending updates are hardened in the VLDS  
  ■ the VLDS is dynamically unallocated  
  ■ other members in the registry's XCF group that are eligible to become the Local Owner attempt to acquire the registry VLDS and expose it  
  Valid for shared RTCS system registries and RTCS product registries. |
| REALLOCATE  | If the RTCS system registry VLDS is not currently owned by any member of the XCF group, this member dynamically allocates the registry and establishes the registry data space for access on this member only.  
  Other members in the XCF group are not able to access data in the registry until the registry is exposed.  
  Valid for shared and nonshared RTCS system registries and RTCS product registries. |
| ACQUIRE     | If the RTCS system registry VLDS is not currently owned by any member of the XCF group, this member dynamically allocates the registry, establishes the registry data space, and exposes the registry for access by all members of the registry's XCF group.  
  Valid for shared RTCS system registries and RTCS product registries. |
| ALTER operand | The ALTER subcommand performs various functions based on the operand used with it.  
  Valid for product registries and  
  For more information, see “operand parameter” on page 82. |

**registry**

*(optional)* The *registry* operand indicates the registry you want to take action against.

Use the following syntax to specify a registry:

```
[ SYSTEM | MEMORY ]
```

SYSTEM and MEMORY are only valid when issuing commands to the RTCS address space.
If the registry operand is not used and the command is issued in

- the RTCS subsystem address space, the default is SYSTEM
- a PAS, the default is the product’s registry

**PACKAGE, {MAINTenance|STATus}, {package|group}**

Valid context: RTCS subsystem, Generalized Server, or MainView PAS

Using **PACKAGE** with the **MAINT** operand requests that the indicated package or group of packages are to be examined and maintenance that has been applied are to be displayed. Elements that have not had any PTFs or ZAPs applied to them will not be listed in the resulting display.

Using **PACKAGE** with the **STAT** operand displays the status of all elements contained in the indicated package or group of packages. Because all SMP/E elements in the package or in all packages in the group will be displayed, the resulting response will contain a very large number of lines.

The package variable is the name of an RTCS package currently in storage, or previously loaded into DLPA (packages loaded into DLPA currently are: OSZLPALB, OSZKERNL, OSZLIBC, OSZLIBM, and OSZGCC2).

The group variable is KERNEL or LIBRARY.

--- **EXAMPLE**

Use the following command to display the RTCS maintenance levels:

```
F RTCS, PACKAGE MAINT KERNEL
```

Use the following command to display the level of all RTCS modules:

```
F RTCS, PACKAGE STATUS KERNEL
```

**START, productIdentifier[,J=cscb]**

Valid context: RTCS subsystem only

This command directs RTCS to use the z/OS START command to start a Generalized Server address space.

RTCS constructs the necessary START command and issues it internally. The CSCB name for the Generalized Server is the same as its product identifier, unless you specify another value by using the J operand.
Two of the subcommands have optional operands as described in Table 4.

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Operand</th>
<th>Description</th>
</tr>
</thead>
</table>
| TRANSFER   | TO(targetSystem) | If the RTCS system registry is shared within a sysplex and the Local Owner is the current system  
|            |                 | ■ the registry is released by this member  
|            |                 | ■ the target system is explicitly instructed to acquire the RTCS system registry VLDS  
|            |                 | The targetSystem variable is the XCF member name, which can be determined from the output of the REGISTRY STATUS command.  
|            |                 | Valid for shared RTCS system registries and RTCS product registries. |
The RTCS Security Manager provides internal interfaces for products to take advantage of the facilities that are offered by the three external security managers for z/OS: RACF, CA Top Secret, and CA ACF2. Products use the RTCS Security Manager for all ESM interface functions, and security for the products is configured by using the RTCS RIU. Products might use all or part of the RTCS Security Manager or not use it at all. MainView Infrastructure uses the RTCS Security Manager, but maintains its security configuration separately (that is, not in the RTCS system registry).

### Table 4  Operands (part 2 of 2)

<table>
<thead>
<tr>
<th>Subcommand</th>
<th>Operand</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTER</td>
<td>DSNAME(dsname)</td>
<td>If the RTCS system registry VLDS is in an UNALLOCATED state, a new SREGVLDS DSNAME is establish for the registry VLDS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The new SREGVLDS DSNAME is used when a subsequent ACQUIRE or REALLOCATE REGISTRY command is issued.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid for shared and nonshared RTCS system registries and RTCS product registries.</td>
</tr>
<tr>
<td>DIV-SAVE-MINIMUM(nn)</td>
<td>DIV-SAVE-MAXIMUM(nn)</td>
<td>These variants of the ALTER keyword set the DIV SERVICES performance parameters. For more information on these specific parameters, see Appendix D, “RTCS initialization member.”</td>
</tr>
<tr>
<td>DIV-SAVE-IDLE(nn)</td>
<td>DIV-SAVE-LIMIT(nnnnnn)</td>
<td>The scope of all of these parameters are an individual XCF group member. They can be set independently for each member of the group. The values can be changed at any time (allocated or unallocated).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Altering the values on one XCF member does not affect any other member of the XCF group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Valid for shared and nonshared RTCS system registries and RTCS product registries.</td>
</tr>
<tr>
<td>ELIGIBLE-OWNER(YES</td>
<td>NO)</td>
<td>Sets the status of the current XCF member to be eligible or ineligible to become the Local Owner of the RTCS system registry VLDS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Only eligible owners will attempt to acquire the registry VLDS after it has been released by another member in the registry XCF group (specified by the RTCS initialization member REGISTRY-XCF-GROUP option).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The scope of this parameter is an individual XCF group member. Valid for shared RTCS system registries.</td>
</tr>
</tbody>
</table>
Global security parameters

The RTCS Security Manager uses the RTCS system registry to store its configuration structures and any site-specific preferences that have been established during configuration. Two types of configuration structures are supported by the RTCS Security Manager and stored in the RTCS system registry: Global Security Parameters and Global Class Properties.

Each RTCS Security Manager registry structure type consists of two paired, distinct records: BASE and CUST. Each record of each structure type is added to, and updated in, the RTCS system registry by means of the RTCS RIU. Import files in XML format are provided in the TOSZHTML SMP/E target library for each type and each record.

The BASE records of each type are established during the initial RTCS system registry initialization during RTCS configuration for each z/OS image. BASE records contain the default values for each parameter or option that is provided by RTCS. BASE records are never modified by RTCS or customized by the site.

CUST records contain any altered parameters or option values that are customized by the site for that particular RTCS system registry (and the specific z/OS image).

**NOTE**

MainView products maintain their own security configuration and resource definitions independently of RTCS, and do not store this information in the RTCS system registry. This product configuration information is provided to the RTCS Security Manager internally, which uses it to perform the same functions as if the MainView security parameters were stored in the RTCS system registry.

Global security parameters

The RTCS Security Manager maintains global (rather than product-specific) configuration preferences in the Global Security Parameters (GSP) RTCS system registry structure.

The RTCS Security Manager provides default Global Security Parameters that are suitable for most sites, ESM products, and z/OS configurations. These Global Security Parameters can be customized when necessary.

The RTCS Global Security Parameters structure supports the following parameters and options:

**ESMTYPE**

The ESMTYPE parameter specifies the type of ESM in use on the z/OS image. Supported values include RACF, ACF2, TSS, NONE, and AUTO. The default value is AUTO, which means that RTCS dynamically determines which ESM is active on the z/OS image.
SAFSUBSYS

The SAFSUBSYS parameter specifies the SAF SUBSYS name to be passed to the ESM by using RACROUTE requests.

APPLID

The APPLID parameter specifies the default APPL name to be passed to the ESM by using RACROUTE VERIFY and AUTH requests when a client product specification is not provided.

DEFAULTUSERID

The DEFAULTUSERID parameter specifies the default USERID, LOGONID, or ACID to be passed to the ESM when an undefined user ID is specified by a user or is being inherited, and undefined user IDs are indicated as being supported. See also “UNDEFINEDUSERINHERIT” and “UNDEFINEDUSERSIGNON.”

UNDEFINEDUSERINHERIT

The UNDEFINEDUSERINHERIT parameter specifies whether an undefined USERID, LOGONID, or ACID may be inherited on the system. This parameter can have the following values:

- REJECT indicates that such an attempt is to be rejected.

- ACCEPT indicates that it is allowed; in this case, the DEFAULTUSERID can be used if the RTCS client product does not provide a product-, context-, or server- (instance-) specific default.

UNDEFINEDUSERSIGNON

The UNDEFINEDUSERSIGNON parameter specifies whether an undefined USERID, LOGONID, or ACID may be signed on to the system. This parameter can have the following values:

- REJECT indicates that the signon is to be rejected.

- ACCEPT indicates that the signon is allowed; in this case, the DEFAULTUSERID can be used if the RTCS client product does not provide a product-, context-, or server- (instance-) specific default.
**GROUPINHERIT**

The GROUPINHERIT parameter specifies the GROUP IDENT inherit scope option. This parameter can have the following values:

- **ALWAYS**, the default value, indicates that when a USERID, LOGONID, or ACID is inherited, the originating security environment’s GROUP IDENT is also inherited.

- **NEVER** indicates that the originating GROUP IDENT is to be ignored, and the GROUP IDENT is allowed to default on the system (that is, as if no GROUP IDENT was specified).

**SECTRACE**

The SECTRACE parameter specifies the default SECTRACE option, which is the one that is set by the `F RTCS, SECTRACE, option` command. The supported values for this option include: NONE, SIMPLE, EXTENDED, and COMPLETE.

---

**Customizing RTCS global security parameters**

You can customize RTCS global security parameters by creating or updating a CUST global security parameter data structure record in the RTCS system registry by using the following steps.

1. Execute the RTCS RIU to import the registry IMPORT file (member) that defines the contents of the fields in the global security parameter data structure.

   Member OSZRSGSP of the TOSZRXML SMP/E target library contains a model registry IMPORT file, and member OSZJSGSP of the TOSZCNTL SMP/E target library contains a sample job to import the file. The SYSIN input stream of the OSZJSGSP sample job describes each of the individual options and parameters and their default and supported optional values.

2. After you make changes to the global security parameters, make the changes effective by entering the following operator command:

   ```
   F RTCS,REFRESH,SECURITY
   ```

---

**NOTE**

RTCS global security parameters do not normally need to be customized. RTCS is capable of determining all of the relevant ESM configuration information properly. In addition, MainView products independently maintain their own security parameters and resource definitions and provide this information to the RTCS Security Manager for its use.
In the preceding command, RTCS is the SSID of the RTCS subsystem.

**Global class properties**

For each supported SAF resource CLASS name, the RTCS Security Manager typically provides a global (that is, common to all products) class property definition, which can be customized for each site, in the Global Class Properties (GCP) RTCS system registry structure. The RTCS Security Manager provides default Global Class Property records for all standard z/OS SAF resource class names that are used in all currently supported products.

The RTCS Global Class Properties structure supports the following parameters and options:

**CLASS**

The CLASS parameter specifies the 8-character SAF resource CLASS name.

**DESC**

The DESC parameter specifies a 64-byte maximum description of the CLASS.

**TSSCLASS**

The TSSCLASS parameter specifies the 8-character internal TSS RDT CLASS name.

**ACTIVE**

The ACTIVE parameter indicates whether the CLASS is active. This parameter can have the following values:

- If the value is YES, authorization requests are processed (passed to the ESM) for the CLASS.

- If the value is NO, authorization requests are not processed and a return code 4 is returned for authorization requests. This return code indicates that the resource manager decides whether to allow resource access. Access is typically allowed, but this is resource CLASS-specific.
MAXLEN

The MAXLEN parameter specifies the maximum resource ENTITY name length. You can specify a value of 1 through 253, CDT, STD, or LONG.

- CDT indicates that the maximum length that is specified in the RACF Class Descriptor Table is to be used.
- STD specifies that the standard TSS RDT length, 8, is to be used.
- LONG indicates that the long TSS RDT length (which is usually 44) is to be used.

LOGOPT

The LOGOPT parameter specifies the default logging option. This parameter can have the following values:

- ASIS, the default value, indicates that the rule, profile, or permit specification is to be used.
- NOLOG indicates that no logging is to be performed.
- NOFAIL indicates that access authorization failures are not to be logged.
- NOAUTH indicates that access authorization successes are not to be logged.

Customizing RTCS global class properties

You can customize RTCS Global Class Properties by creating or updating the Global Class Property data structure CUST record for any given resource class name in the RTCS system registry.

Perform this update by executing the RTCS RIU to import the registry IMPORT file that defines the contents of the fields in the Global Class Property registry data structures.

Member OSZRSGCP of the TOSZRXML SMP/E target library contains a model registry IMPORT file, and member OSZJSGCP of the TOSZCNTL SMP/E target library contains a sample job to import the file. The SYSIN input stream of the OSZJSGCP sample job describes the individual Global Class Properties that may be specified for a SAF resource class name and describes their possible values.
Planning RTCS configuration

This appendix contains information to help you plan your RTCS configuration.

Descriptions of fields used in MainView Customization that pertain to RTCS . . . . . . 90
RTCS worksheet . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 91
Descriptions of fields used in MainView Customization that pertain to RTCS

This section contains a description of the fields used in MainView Customization to configure RTCS. The fields are presented in alphabetical order.

**RTCS EXEC PROC**

The name of the JCL procedure to start the RTCS Generalized Server started task.

**RTCS INIT PROC**

The name of the JCL procedure to start the RTCS initiator started task.

**RTCS PROC**

The name of the JCL procedure to start the RTCS subsystem address space.

**SYSPLEX NAME**

The name (if applicable) of the common sysplex in which the MainView products will run.

The sysplex name can be displayed by issuing the D SYMBOLS command from the operator console.

**SYSTEM NAME**

The name of the z/OS image in which the MainView products will run, typically the '&SYSNAME' system variable.

The system name can be displayed by issuing the D SYMBOLS command from the operator console.
Use the following worksheet to configure RTCS. The columns on the worksheet correspond to the fields on customization panels. For a description of the fields, see “Descriptions of fields used in MainView Customization that pertain to RTCS” on page 90.

Complete one worksheet for each sysplex on which you are going to customize a product. Each row on a worksheet represents an LPAR. The worksheets are presented in alphabetical order.

**Note**

If you are installing and customizing MainView Infrastructure, you can use the MainView Infrastructure worksheet instead of the RTCS worksheet. The MainView Infrastructure worksheet is in the *MainView Installation Guide*.

<table>
<thead>
<tr>
<th>SYSPLEX NAME</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SYSTEM NAME</th>
<th>RTCS PROC&lt;sup&gt;a&lt;/sup&gt;</th>
<th>RTCS INIT PROC&lt;sup&gt;a&lt;/sup&gt;</th>
<th>RTCS EXEC PROC&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<sup>a</sup> If these items are not identical across all LPARs in a sysplex, you must use LPAR-level sharing (specified when specify runtime system values during customization). PROCs can be shared under the same name, and you can distinguish the PROCs by using the JOBNAME= parameter on the START command.
This chapter provides information about how to use the RTCS Registry Import Utility (RIU).

Introduction ................................................................. 94
Registry basics ............................................................... 94
Executing the RTCS RIU .................................................... 95
Required data definition names ........................................... 95
  SYSIN ................................................................. 96
  SYSPRINT ............................................................. 96
Control statements ........................................................ 96
  SET ................................................................. 96
  IMPORT ........................................................... 97
  OPTIONS .......................................................... 98
Return codes and messages .............................................. 99
  Return codes ..................................................... 99
  Messages ......................................................... 100
Introduction

The RTCS RIU facilitates the use of the RTCS system registry for product installation and configuration by using an XML document. The RIU program (OSZRGIMP) processes XML documents (members of the SMP/E TOSZRXML target library) and updates the RTCS system registry data structures.

Data structures that are to be established in the RTCS system registry for RTCS can be defined in an RIU IMPORT file XML document. Field values in these structures can be parameterized at RIU execution time by using a symbol substitution mechanism. An overall set of product configuration data can be described and shipped with the product and then used to configure each individual system or product instance. The values of these substituted symbols are specified by SET control statements in the SYSIN input stream to the RIU.

The RIU verifies the integrity of the complex data structures against their unique type definition before updating the registry. In addition, substituted data values are validated based on the specific data type that is required within a data structure. This action protects the registry data from corruption at the syntax level.

Registry basics

The RTCS system registry is a hierarchical set of keys and values. A registry key is a named container of other keys and values. A registry value is a named data element that has a specific data type.

Each registry key or registry value is uniquely identified by a registry path, which is the set of key names descending from the root key, down to the key or value that is being examined. This registry path can be written in the same way as a hierarchical file system path. For example, a key named software that is immediately under the root key is written /software and a value named foo within that key is written /software/foo.

A field in a registry value can be a complex structured data type that contains other complex data types, or it can be a simple, predefined data type (such as an integer or string data element). As a result, binary data structures can be easily stored and directly retrieved for use by programs. All registry key and registry value data types can be represented as an XML document.
Executing the RTCS RIU

Figure 13 provides an example of the JCL that is needed to execute the RTCS RIU. The text that should be changed is shown in bold. This sample JCL is contained in member OSZINJ91 in the TOSZCNTL target library.

**Figure 13  RTCS registry Import Utility job**

```
//jobname  JOB (acct,room),programmer,NOTIFY=&SYSUID
//IMPORT  EXEC PGM=OSZEXEC8,REGION=4M,
//         PARM='P=OSZRGIMP,C=BMCPROD'
//XMLDOC  DD DSN=USERID.YOUR.PDS,DISP=SHR
//SYSPRINT DD SYSOUT=*  
//SYSIN    DD *
SET ISA.SSID     = RTCS
SET ISA.POSZHTML = SYS2.RTCS.POSZHTML
SET ISA.POSZLINK = SYS2.RTCS.POSZLINK
IMPORT XMLDOC(IMPTFILE)
```

Figure 13 shows sample JCL that will import the IMPTFILE member of the USERID.YOUR.PDS data set. The results and messages are sent to the SYSPRINT SYSOUT data set.

The RIU (OSZRGIMP) is invoked by using the RTCS Generalized Server (OSZEXEC8). You do not need to specify the DSNAME for the RTCS product program library anywhere in this job. The data set will be dynamically allocated by the RTCS Generalized Server, which is loaded into dynamic LPA as part of RTCS subsystem initialization.

**Required data definition names**

The RIU requires a minimal set of data set allocations in order to function properly. This section lists the data definition names (DDNAMEs) of each of the required allocations and explains their purpose.
SYSIN

The RIU executes by interpreting control statements in the data set that is specified by the SYSIN DD statement. Each record is read and interpreted as either an individual control statement or a comment.

A comment record begins with an asterisk (*) as the first non-blank character. Comments are ignored and can be used for documentation purposes as appropriate for the control statements. For more information about control statement records, see “Control statements.”

SYSPRINT

The RIU generates standard listing-type output to the data set that is specified by the SYSPRINT DD statement. This output includes informational and error messages that occur during the processing of the RIU control statements that are read from SYSIN.

Control statements

Control-statement records that are supplied in the SYSIN DD allocation drive all RIU processing. A control statement is a single record of input, which begins with a keyword such as SET, IMPORT, or OPTIONS. The keyword must be the first non-blank characters on the record.

Each control statement is processed completely before the RIU advances to the next statement. If an error is detected in the syntax of a control statement, all processing terminates. An error message is generated to SYSPRINT to indicate the nature of the problem and provide the SYSIN input record number to help you to locate the problem.

SET

The SET control statement creates or replaces a name/value mapping that is used by the RIU to substitute elements in registry IMPORT files. The SET statement is written as

```
SET variable = value
```

Substitution `variable` names can begin with any alphabetic (A-Z) or national symbol ($, #, or @ in United States English).
The value that is established by this control statement is used for any IMPORT files that are subsequently processed.

You can include one or more blank characters in a substitution value if the value is surrounded by either single quotation marks ('') or double quotation marks ("''). If you use the single quotation mark, you cannot use a double quotation mark within the substitution value.

**IMPORT**

The IMPORT control statement directs the RIU to read the registry definitions that are present in the specified file and apply them to the RTCS system registry.

The registry import file can be a sequential data set or a partitioned data set member that is record-oriented. You can specify an explicit data set name or a ddname that has been allocated to the job step.

- To specify a ddname, use the ddname that was allocated in the job step by using the following format:

  **IMPORT ddname**

  To specify a ddname that has been allocated to a partitioned data set, include the specific member name by using the following format:

  **IMPORT ddname(member)**

- To specify a specific data set name, prefix the data set name with the characters DSN: by using the following format:

  **IMPORT DSN:datasetName**

  To specify a partitioned data set, include the specific member name by using the following format:

  **IMPORT DSN:datasetName(member)**
OPTIONS

Use the OPTIONS control statement to adjust the behavior of the RIU for all control statements that follow that OPTIONS control statement. The following syntax should be used for the OPTIONS control statement:

OPTIONS [[NO]option[,...]]

The option variable value can be TRACE, TRACEX, or PRETEND.

Example: OPTIONS TRACE,NOPRETEND

The supported options instruct the RIU to generate additional output to describe the actions that are taken while processing an import file. Each of these options has a corresponding negation that is preceded by NO—NOTRACE, NOTRACEX, and NOPRETEND; these values are the default values. Specify multiple options on a single OPTIONS control statement by separating each option with a comma.

TRACE

The TRACE option enables the generation of detailed diagnostic information regarding the changes or requests that were made to the RTCS system registry while processing the IMPORT file. This information provides a record of what was actually done during RIU processing, such as creating, updating, and verifying keys and values. Each of these messages is generated with informational severity and does not affect any actual RIU processing.

TRACEX

The TRACEX option enables the generation of detailed diagnostic information regarding the parsing and interpretation of the XML structure of the IMPORT file. This information includes messages regarding the start and end of elements, the attributes processed, and the substitutions that were performed. This information is intended for debugging purposes.

PRETEND

The PRETEND option prevents the RIU from carrying out any updates to the RTCS system registry but otherwise completely processes the IMPORT file. If the TRACE option is enabled, messages are generated that describe the actions that would have been taken against the RTCS system registry. The PRETEND option allows a trial run of one or more IMPORT files to check for errors before committing the changes.
Return codes and messages

During the processing of the RIU, several different forms of feedback are provided to indicate the success or failure of each invocation. This section details the return code and general message categories.

Return codes

When the RIU encounters an error that causes it to terminate without continuing, a return code is generated to indicate the nature of the failure. Each return code is described in Table 5.

Table 5  Registry Import Utility return codes

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>The RIU is unable to open SYSPRINT. Ensure that the SYSPRINT DD statement is correctly specified, and then resubmit the JOB.</td>
</tr>
<tr>
<td>16</td>
<td>The RIU is unable to open SYSIN. Ensure that the SYSIN DD statement is correctly specified, and then resubmit the JOB.</td>
</tr>
<tr>
<td>12</td>
<td>The RIU is unable to open a data set that is associated with an IMPORT control statement. Error messages are generated to SYSPRINT to indicate the IMPORT statement that contains the failing data set or data set with member name. If the failing IMPORT statement uses a ddname, check the allocation of that ddname for errors (for example, ensure that the specified member exists).</td>
</tr>
<tr>
<td>8</td>
<td>An error occurred during the processing of either an RIU control statement or an XML import file. This return code is accompanied by messages that are written to SYSPRINT that diagnose the problem that occurred.</td>
</tr>
</tbody>
</table>
Messages

During the processing of IMPORT control statements and registry IMPORT files, the RIU generates a variety of messages. Each message is prefixed with OSZ and ends with one of the following message severity indicators:

- E - Error
- W - Warning
- I - Informational

All RIU messages are in the range OSZ1500x through OSZ1599x. Each message that is generated by the RIU is documented in the online Runtime Component System Message Reference.

If you have not changed the default values for the OPTIONS control statement, the RIU generates a small set of messages for all invocations, including

- listing of each control statement as it is being processed
- indication of the start and end of processing of each IMPORT file
- completion of the import utility program
- any warning and error messages that were issued during processing

When the TRACE option is specified, the RIU generates additional messages detailing the processing of each control statement, as well as the results of each update to the registry. These additional messages are informational.

When the TRACEX option is specified, the RIU generates additional messages detailing the progress in parsing the elements and attributes of the XML IMPORT file. These additional messages are informational and are intended for debugging problems with the RIU operation.

**NOTE**

A distinct error message is always generated by any failures in the XML document input.
Sample JCL and z/OS image configuration members

This appendix provides reference information regarding the RTCS started task PROCs and required z/OS program property table additions. Refer to these examples for assistance in configuring these elements for an z/OS image.

RTCS Initiator started task PROC .................................................. 102
  STEPLIB DD statement ............................................................ 103
RTCS subsystem started task PROC ............................................. 104
RTCS Generalized Server started task PROC ............................... 105
z/OS Program Properties Table additions ................................. 105
RTCS Initiator started task PROC

Figure 14 is an example of the RTCS Initiator started task PROC. The text in bold represents items that you need to replace with values that are specific to your site; see Table 6 for descriptions of these items. Update these statements to reflect the actual DSNAMES and PROCs to be used. The DSNAMES are established by the Desktop Installation Assistant, if it is used to configure the target z/OS image. If you copy this PROC for use on another z/OS image, update these DD statements, if necessary. This sample JCL is contained in member OSZ$INIT in the TOSZCNTL target library.

Figure 14  RTCS initiator started task PROC

| //OSZINIT PROC |
| EXEC PGM=OSZSIRIS, RTCS Initiator Program |
| TIME=1440, No SMF CPU or wait time limits |
| REGION=0M No limits on virtual storage |
| STEPLIB: RTCS Subsystem Program Library (.TOSZRTCS, or a copy) |
| STEPLIB DD DISP=SHR, DSN=SYS2.RTCS.POSZRTCS |

Table 6 provides descriptions of the values that are shown in bold in Figure 14.

Table 6  RTCS Initiator started task PROC customizations

<table>
<thead>
<tr>
<th>Default Value</th>
<th>Description</th>
<th>Detailed information</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSZINIT</td>
<td>RTCS subsystem started task PROC name</td>
<td>“RTCS subsystem started task PROC” on page 104</td>
</tr>
<tr>
<td>SYS2.RTCS.POSZRTCS</td>
<td>RTCS subsystem program library DSNAME</td>
<td>“STEPLIB DD statement” on page 103</td>
</tr>
</tbody>
</table>

Copy this member into one of the following locations:

- SYS1.PROCLIB or another PROCLIB that is available for started tasks that are started by using the Master Subsystem (MSTR)
- a JES-managed started task PROCLIB.

The RTCS Initiator address space should be started as early as possible in the z/OS IPL process after the security system has been initialized. The following examples illustrate the appropriate command (the RTCS Initiator started task PROC member name is shown in bold).

S OSZINIT
S OSZINIT, SUB=MSTR
The RTCS Initiator address space is not the same address space as the RTCS subsystem address space. The RTCS Initiator starts the RTCS subsystem address space internally after performing validation procedures on the following items:

- z/OS system image
- STEPLIB data set (which becomes the POSZRTCS parameter, internally)
- POSZLINK data set
- POSZHTML data set
- SREGVLDS VSAM LDS used to back the RTCS system registry (for dedicated, nonshared RTCS system registry configurations)
- POSZPSWD product authorization table library

**STEPLIB DD statement**

The RTCS Initiator internally passes the DSNAMEs of the data sets that are used by the RTCS subsystem and the RTCS Generalized Server address spaces in z/OS ECŞA storage. The RTCS Initiator obtains the DSNAME of the RTCS subsystem program library (POSZRTCS) from the STEPLIB DD statement of the RTCS Initiator address space started task PROC JCL. The RTCS subsystem address space dynamically allocates this data set, loads the RTCS kernel and DLLs, and deallocates the program library to enable maintenance to be performed by using SMP/E.

The DSNAME of the RTCS subsystem program library (which is specified by the STEPLIB DD statement of the RTCS Initiator started task PROC JCL) is used as the model to determine the default DSNAMEs of the other RTCS data sets used by the RTCS subsystem and the RTCS Generalized Server. The data set prefix (the part up to but not including the low-level index) is determined and the low-level index is examined to see if it is $xOSZRTCS$, where $x$ is any character. If it is $xOSZRTCS$, the default DSNAMEs of all of the other data sets will be constructed from the POSZRTCS DSNAME prefix and their own individual, appropriate low-level index, such as $xOSZLINK$, $xOSZHTML$, $xOSZPSWD$, and REGISTRY.
**RTCS subsystem started task PROC**

Figure 15 is an example of the RTCS subsystem started task PROC. The text that should be customized is shown in bold. This sample JCL is contained in member OSZ$RTCS in the TOSZCNTL target library.

**Figure 15  RTCS subsystem started task PROC**

```
//OSZRTCS  PROC
//
//OSZRTCS EXEC PGM=OSZMOSYS, RTCS Subsystem Initialization
// TIME=1440. No SMF CPU or wait time limits
// REGION=0M No limits on virtual storage
```

Copy this member into one of the following locations:

- SYS1.PROCLIB or another PROCLIB that is available to started tasks that are started under the Master Subsystem (MSTR)

- a JES-managed started task PROCLIB

Specify the member name of this PROC in the OSZRTCS-PROC option of the RTCS Initialization member of Logical PARMLIB. If the name of the PROC used to START the RTCS Initiator address space contains the four characters INIT, the default name of the RTCS subsystem started task PROC will be the same, except that the characters RTCS will be substituted for the characters INIT in that PROC name. If you name the started task PROCs in this manner, you do not need to update the RTCS Initialization member (for this purpose).

The RTCS subsystem control address space is started internally by the RTCS Initiator address space, as early as possible in the z/OS IPL process. If the RTCS Initiator address space (by default, PROC OSZINIT) was started under the Master Subsystem using SUB=MSTR, then, by default, the RTCS subsystem address space will be started under the Master Subsystem also.
RTCS Generalized Server started task PROC

Figure 16 is an example of the RTCS Generalized Server started task PROC. This sample JCL is contained in member OSZ$EXEC in the TOSZCNTL target library.

Figure 16  RTCS Generalized Server started task PROC

```
//OSZEXEC PROC KEY=8,P=*,C=*,O=
//*
//OSZEXEC EXEC PGM=OSZEXEC&KEY,       Generalized Server execution key
//                  TIME=1440,          No SMF CPU or wait time limits
//                  REGION=0M,         No limits on virtual storage
//                  PARM=('P=&P',     Product or Package to initialize
//                  'C=&C',           General Server RTCS Context Name
//                  '&0',)          Instance-specific Server options
///*
```

Copy this member into the following locations:

- a JES-managed started task procedure library that is available to started tasks that are started under the primary JES
- SYS1.PROCLIB or another PROCLIB that is available to started tasks that are started under the Master Subsystem (MSTR)

Specify the member name of this PROC in the OSZEXEC-PROC option of the RTCS Initialization member of Logical PARMLIB. If the name of the PROC used to START the RTCS Initiator address space contains the four characters INIT, the default name of the RTCS Generalized Server started task PROC will be the same, except that the characters EXEC will be substituted for the characters INIT in that PROC name. If you name the started task PROCs in this manner, you do not need to update the RTCS Initialization member (for this purpose).

z/OS Program Properties Table additions

Figure 17 on page 106 shows entries that must be added to the z/OS program properties table. This sample JCL is contained in member OSZSCHED in the TOSZCNTL target library.
**PRODUCT IDENTIFICATION**

/* PRODUCT_NAME: RTCS COMPONENT_ID: KERNEL */
/* PRODUCT_FMID: ZOSZ120 DIAGNOSIS_PIDS: ZOSZ-120 */
/* CURRENT_RMIC: BASE LAST_UPDATE: 11/07/07 */

**MEMBER_DESCRIPTION**

/* MEMBER: OSZSCHED */
/* STATUS: EXTERNAL INSTALLATION MATERIALS */
/* CONTENTS: THIS MEMBER CONTAINS THE PROGRAM PROPERTIES TABLE */
/* ENTRIES REQUIRED TO PROPERLY INITIATE THE RUNTIME */
/* COMPONENT SYSTEM INITIATOR, ALLOCATOR, AND KERNEL */
/* STARTED TASKS, PLUS GENERALIZED SERVER JOB STEPS. */

---

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZSIRIS)</th>
<th>/* RTCS SUBSYSTEM INITIATOR STC */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(0)</td>
<td>/* RTCS SUBSYSTEM INITIATOR KEY 0 */</td>
<td></td>
</tr>
<tr>
<td>PRIV</td>
<td>/* WLM COMPATIBILITY MODE PGN=000 */</td>
<td></td>
</tr>
<tr>
<td>SYST</td>
<td>/* NON-TIMED STARTED SYSTEM TASK */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZMOSYS)</th>
<th>/* RTCS SUBSYSTEM ALLOCATOR STC */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(0)</td>
<td>/* RTCS SUBSYSTEM ALLOCATOR KEY 0 */</td>
<td></td>
</tr>
<tr>
<td>PRIV</td>
<td>/* WLM COMPATIBILITY MODE PGN=000 */</td>
<td></td>
</tr>
<tr>
<td>NOCANCEL</td>
<td>/* KERNEL CANNOT BE CANCELLED */</td>
<td></td>
</tr>
<tr>
<td>NOSWAP</td>
<td>/* NON-SWAPPABLE, CROSS-MEMORY A/S */</td>
<td></td>
</tr>
<tr>
<td>SYST</td>
<td>/* NON-TIMED STARTED SYSTEM TASK */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZIRIAN)</th>
<th>/* RTCS SUBSYSTEM KERNEL STC */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(0)</td>
<td>/* RTCS SUBSYSTEM KERNEL KEY 0 */</td>
<td></td>
</tr>
<tr>
<td>PRIV</td>
<td>/* WLM COMPATIBILITY MODE PGN=000 */</td>
<td></td>
</tr>
<tr>
<td>NOCANCEL</td>
<td>/* MUST NOT BE CANCELLED */</td>
<td></td>
</tr>
<tr>
<td>NOSWAP</td>
<td>/* NON-SWAPPABLE, CROSS-MEMORY A/S */</td>
<td></td>
</tr>
<tr>
<td>SYST</td>
<td>/* NON-TIMED STARTED SYSTEM TASK */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC0)</th>
<th>/* RTCS SERVER STARTED TASK - 0 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(0)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 0 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC1)</th>
<th>/* RTCS SERVER STARTED TASK - 1 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(1)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 1 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC2)</th>
<th>/* RTCS SERVER STARTED TASK - 2 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(2)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 2 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC3)</th>
<th>/* RTCS SERVER STARTED TASK - 3 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(3)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 3 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC4)</th>
<th>/* RTCS SERVER STARTED TASK - 4 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(4)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 4 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC5)</th>
<th>/* RTCS SERVER STARTED TASK - 5 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(5)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 5 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC6)</th>
<th>/* RTCS SERVER STARTED TASK - 6 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(6)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 6 */</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PPT</th>
<th>PGMNAME(OSZEXEC7)</th>
<th>/* RTCS SERVER STARTED TASK - 7 */</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY(7)</td>
<td>/* PRODUCTS WHICH EXECUTE IN KEY 7 */</td>
<td></td>
</tr>
</tbody>
</table>
This appendix presents the contents of the RTCS initialization member.

**Figure 18  RTCS initialization member  (part 1 of 11)**

<table>
<thead>
<tr>
<th>Runtime Component System (RTCS) Initialization Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSZ$PARM       Distributed in SMP/E distribution library .DOSZCNTL</td>
</tr>
<tr>
<td>OSZ$PARM       Installed into SMP/E target library .TOSZCNTL</td>
</tr>
<tr>
<td>OSZINI00       Customized in the z/OS Logical PARMLIB data set</td>
</tr>
</tbody>
</table>

OSZ$PARM is customized by the installation and then copied into the z/OS image Logical PARMLIB data set (usually DSN=SYS1.PARMLIB). The RTCS Initiator will attempt to locate and read the RTCS Initialization Parameters member from an MVS Logical PARMLIB data set (or SYS1.PARMLIB if no MVS image-specific data set has been established).

The default member name is OSZINI00, which can be changed by specifying the INI=nn parameter in the RTCS Initiator START command parameter field (the 4th positional parameter). For example:

```
START OSZINIT,,,(INI=42),SUB=MSTR
```

---

MVS Subsystem Name to be used by the RTCS Subsystem.

+ **SSID=RTCS**

The MVS Subsystem Name (SSID, or Subsystem ID) that is to be used by the RTCS Subsystem address space (OSZRTCS).

Installation Verification Procedure (IVP) Mode
In IVP mode, the RTCS Initiator performs all normal parameter verification and processing but does not START the RTCS Subsystem address space. IVP mode can also be specified in the parameter field (the 4th positional parameter) of the RTCS Initiator START command, as follows:

```
START OSZINIT...,(IVP=Y,LIST=Y)
```

If IVP mode is requested on the START command, a specification of NOIVP in the Logical PARMLIB member will NOT disable IVP mode. Once IVP mode is in effect (either from the START command or from this member), it cannot then be disabled by specifying NOIVP (in this member).

**RTCS Subsystem address space started task PROC name.**

```
+ OSZRTCS-PROC=OSZRTCS
```

If not specified, then the default is the same PROC name that was used to start the RTCS initiator with 'RTCS' substituted for 'INIT', provided that 'INIT' appears in the RTCS Initiator PROC name. Else the default is OSZRTCS.

**RTCS Generalized Server started task PROC name.**

```
+ OSZEEXEC-PROC=OSZEEXEC
```

If not specified, the default is the RTCS Subsystem PROC name with 'EXEC' substituted for 'RTCS', provided that 'RTCS' appears in the RTCS Subsystem PROC name. Else the default is OSZEEXEC.

**RTCS Product Program Library (.TOSZLINK, or a copy)**

```
+ POSZLINK=
```

If not specified, then this value will default to STEPLIB-DSName-prefix.[xosz]LINK, provided that the low-level qualifier of //STEPLIB is '.[xosz]RTCS'. Else the default is the same DSNAME as //STEPLIB.

**RTCS Hypertext Document Library (.TOSZHTML, or a copy)**
+ POSZHTML=
  If not specified, then this value will default to
  STEPLIB-DSName-prefix.[xosz]HTML, provided that the
  low-level qualifier of //STEPLIB is '.[xosz]RTCS'.

*     BMC Product (License) Authorization Table Library
* + POSZPSWD=
  If not specified, then this value will default to
  STEPLIB-DSName-prefix.[xosz]PSWD, provided that the
  low-level qualifier of //STEPLIB is '.[xosz]RTCS'.

The Product (License) Authorization Table Library
is a partitioned data set that contains product
license table members that are created & updated
by the BMC Licensing Facility batch password
processing utility, OSZPATLU, or by the legacy
ISPF-based BMC product password update facility.

If no RTCS-based licensed products are being used
then this DD statement may define any partitioned
data set.

MAINVIEW CAS-based product-only customers should simply allocate an empty, DSORG=PO,RECFM=U PDS[E]
for this production library. The indicated data
set will by dynamically allocated by the RTCS
Initiator to ensure its existence and validity, but no MAINVIEW product will ever cause it to
be subsequently dynamically allocated by the RTCS
Subsystem or Generalized Server address spaces.

*     DNAME of RTCS System Registry VSAM Linear Data Set (VLDS)
+ SREGVLDS=SYS2.SHARED.RTCS.SYSTEM.REGISTRY

If not specified, then this value will default to STEPLIB-DSName-prefix.REGISTRY, provided that the low-level qualifier of //STEPLIB is '.[xosz]RTCS'. Otherwise, there is no default, and this parameter must be specified.

The System Registry contains configuration data for RTCS, the MAINVIEW CAS, RTCS-based products, and any RTCS-dependent product which has elected to use it. It must be a VSAM Linear Data Set (VSAM LDS or VLDS).

The Registry data set MUST be cataloged, since it is a VSAM cluster. It is allocated using only its DSNAME. The System Registry VSAM LDS is read/write and cannot physically be shared, although it may be allocated on shared DASD. Only one RTCS Subsystem will be able to allocate a VSAM LDS for use as a Registry because it will be allocated DISP=OLD as required by the MVS DIV service.

But the data in the System Registry VLDS can be shared among members of a Sysplex using XCF. When the System Registry is being shared among RTCS Subsystems running in a Sysplex, then only one RTCS Subsystem will have dynamically allocated the Registry VLDS. That system is called the Local Owner. Other systems can access data in the Registry (which is allocated to the Local Owner) using XCF to transmit requests and retrieve the requested data in response. An RTCS Subsystem that is accessing data in the System Registry on the Local Owner via XCF is termed a Remote [Registry instance].

It is not recommended, but it is possible to have a private, dedicated System Registry VLDS for each RTCS Subsystem. But the MAINVIEW CAS will not be able to share data with other CASs in the Sysplex if you do.

The first time an RTCS Subsystem becomes the Local Owner of the VLDS and encounters a newly-allocated, uninitialized, never-before-used Linear Data Set, the RTCS Subsystem will initialize the contents of the Registry, populating it with all the structures required for RTCS Subsystem components and products.

* System Registry DIV Services Default Performance Parameters
DIV-SAVE-MINIMUM = 1 /* DIV Services interval 1: minimum */
DIV-SAVE-MAXIMUM = 6 /* DIV Services interval 2: maximum */
DIV-SAVE-IDLE = 60 /* DIV Services interval 3: idle */
DIV-SAVE-LIMIT = 4000 /* DIV Services batch update limit */

The above parameters indicate the time that the Registry DIV Services subtask will wait prior to requesting that changes to the System Registry data space be hardened in the backing VSAM LDS. After the VLDS is updated, it will wait a minimum amount of time before the next request to update the VLDS will again be made, but no longer than the indicated maximum (after which a VLDS update will be forced). If the Registry is idle (not being changed) but potentially only being accessed [existing data retrieved], then the DIV Services subtask will idle for the indicated interval before waking up to check for pending requests. The amount of time that pending Registry VLDS updates are delayed is heuristically determined according to request frequency and arrival pattern, and will never less than the indicated minimum value, nor greater than the maximum. Regardless of the enforced minimum and maximum intervals that will cause the VLDS to be updated, if the number of changes exceeds the specified limit then the backing VLDS will be updated, hardening all pending changes on DASD.

A MINIMUM interval of zero (0) indicates that all changes to the System Registry are to be immediately hardened in the backing VSAM LDS, without waiting or attempting to batch multiple changes together into a single update. We recommend that you do NOT specify DIV-SAVE-MINIMUM = 0.

RTCS System Registry Sysplex Sharing Parameters
-----------------------------------------------

+ REGISTRY-XCF-GROUP = OSZRTCSR /* System Registry XCF Group Name */
  /* This parameter will be used only */
  /* if some form of Sysplex Registry */
  /* Sharing (see below) is specified. */
Figure 18  RTCS initialization member (part 6 of 11)

ELIGIBLE-OWNER
/* This member is ELIGIBLE to */
/* ALLOCATE (and then EXPOSE) */
/* the System Registry VLDS. */

/* RTCS Subsystem XCF members which */
/* are not eligible to own the RTCS */
/* System Registry VLDS will not be */
/* able to assist in recovery when */
/* the image that does own the */
/* Registry fails for any reason. */
/* */
/* RTCS Subsystems on small or */
/* slow MVS images should only */
/* remotely access an exposed */
/* System Registry and should */
/* not be eligible to allocate */
/* and expose/own the Registry. */

* *
RTCS System Registry Sysplex Sharing Options

The following five options are mutually exclusive. Only one of them should be specified without the "NO" prefix. The other four may be omitted (or specified with the "NO" prefix as illustrated below). If more than one positive [not prefixed with "NO"] option is specified, then the most restrictive one will become effective. The Registry sharing options are listed below in that precedence order (the most restrictive first, the least restrictive last). In other words, the first (in the order they are listed below) positive (not prefixed with "NO") option specified is the one that will be effective and override any others.

NOPRIVATE-REGISTRY
/* Exclusively allocate the System */
/* Registry VLDS on this image but */
/* do not establish any capability */
/* to share it with other images. */
/* The Registry cannot subsequently */
/* be exposed to other MVS images. */
/* If the System Registry VLDS can */
/* not be allocated DISP=OLD, then */
/* RTCS initialization will fail. */

NOALLOC-REGISTRY
/* ALLOCate the System Registry on */
/* this MVS image, but do not (yet) */
/* EXPOSE it for the REMOTE images */
/* to be able to CONNECT to it. It */
/* can be exposed at a later time */
/* via an RTCS operator command. */
/* If the System Registry VLDS can */
/* not be allocated DISP=OLD, then */
/* RTCS initialization will fail. */
Figure 18  RTCS initialization member  (part 7 of 11)

NOEXPOSE-REGISTRY
/* ALLOCATE and EXPOSE the System */
/* Registry on this MVS image. */
/* If the System Registry VLDS can */
/* not be allocated DISP=OLD, then */
/* RTCS initialization will fail. */

SHARED-REGISTRY
/* Setup a REMOTE Registry on this */
/* MVS image, then CONNECT to the */
/* System Registry if it is already */
/* EXPOSEd on another MVS image. */
/* If the Registry is not already */
/* EXPOSEd, then (if it has not yet */
/* been ALLOCated) ALLOCATE and */
/* EXPOSE the System Registry on */
/* this MVS image (if possible). */
/* If the System Registry VLDS can */
/* not be allocated DISP=OLD or if */
/* this system is unable to CONNECT */
/* to an already-EXPOSEd Registry */
/* on some other MVS image, then */
/* RTCS initialization will fail. */

NOREMOTE-REGISTRY
/* CONNECT to a System Registry */
/* that has already been EXPOSEd */
/* on another MVS image. If this */
/* system is unable to CONNECT to */
/* an EXPOSEd System Registry, then */
/* RTCS initialization will fail. */

*                                    
*    External Security Manager (ESM) Type
*                                    
+ ESMTYPE=AUTO

External Security Manager (ESM) that is in use on this MVS image and which RTCS is to interface with. The default, and the value which most installations should specify, is AUTO. RTCS is usually able to determine the ESM that is being used on an MVS image, but under certain circumstances, it cannot do so. If this situation occurs, you may explicitly indicate which ESM is (or will be) used on this MVS image using the ESMTYPE= parameter.

<table>
<thead>
<tr>
<th>ESMTYPE=</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTO</td>
<td>RTCS is to automatically determine, if possible, which ESM is in use.</td>
</tr>
<tr>
<td>RACF</td>
<td>RTCS is to assume RACF will be used.</td>
</tr>
<tr>
<td>ACF2</td>
<td>RTCS is to assume ACF2 will be used.</td>
</tr>
<tr>
<td>TSS</td>
<td>RTCS is to assume TSS (Top Secret) will be used.</td>
</tr>
</tbody>
</table>
Accept or reject attempts to use undefined ESM User IDs

+ UNDEFINEDUSERINHERIT=ACCEPT
+ UNDEFINEDUSERSIGNON=REJECT

These two parameters specify the behavior when an undefined ESM User ID is provided (either by an end user or an application) as part of original system entry signon, or when attempting to inherit credentials on this image from an existing environment.

The following values can be specified:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEPT</td>
<td>RTCS is to allow the INHERIT or the SIGNON to proceed. RTCS or the ESM will provide a default User ID for the security environment that will be created as a consequence.</td>
</tr>
<tr>
<td>REJECT</td>
<td>RTCS is disallow the INHERIT or the SIGNON.</td>
</tr>
</tbody>
</table>

By default, RTCS allows INHERITs to proceed, since it would then be assumed that the User ID being presented was at least valid in the Sysplex or CASplex somewhere, and disallows SIGNONs with an undefined User ID, since it is usually the case that an invalid User ID (that is, one that is not defined) is never to be allowed.

Default ESM User ID used in place of an undefined one

+ DEFAULTUSERID=' '  

This parameter specifies the ESM User ID that is to be substituted for an undefined/invalid User ID presented for authentication, and the UNDEFINEDUSERINHERIT or the UNDEFINEDUSERSIGNON option, as appropriate, specifies ACCEPT, which indicates that the INHERIT or the SIGNON is to be allowed. The value of this parameter should usually be set to blanks, which triggers ESM-specific behavior to generate its own default User ID (USER, LOGONID, or ACID) according to well-documented ESM behavior or if allowed by ESM-specific security options and parameters.

How to process GROUP IDENT credential during an INHERIT
**GROUPINHERIT=ALWAYS**

This parameter specifies how the GROUP IDENT credential (GROUP name) is to be processed when a security identity is being INHERITed.

The following values can be specified:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALWAYS</td>
<td>RTCS is to pass the GROUP IDENT to the ESM unchanged. In order to be successful, the ESM must allow the use of that specific GROUP IDENT. In the case of RACF, specifically, this means that the User ID being INHERITed must be CONNECTed to the same, exact GROUP IDENT specified in the credentials in the RACF data base being used on this system image.</td>
</tr>
<tr>
<td>NEVER</td>
<td>RTCS is to ignore any specification of a GROUP IDENT (GROUP name) in the authentication credentials presented when attempting to INHERIT a User ID.</td>
</tr>
</tbody>
</table>

**SECTRACE=NONE**

This parameter specifies the default level of diagnostic tracing that is to be performed for the RTCS External Security Manager interface.

The following values can be specified:

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>No ESM interface tracing is to be done.</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>Issue only simple trace messages.</td>
</tr>
<tr>
<td>EXTENDED</td>
<td>Extended tracing is to be performed.</td>
</tr>
<tr>
<td>COMPLETE</td>
<td>Complete tracing is to be performed.</td>
</tr>
</tbody>
</table>

Security diagnostic tracing can be activated dynamically via an RTCS operator command, so NONE should normally be specified. However, if there is a need to perform ESM diagnostic tracing during RTCS initialization, this can be initially activated using this parameter.

**ESM SAF Subsystem (RACROUTE SUBSYS) to be used by RTCS**
This parameter specifies the RACROUTE SUBSYS (SAF Subsystem ID) that is to be used by the RTCS Security Manager. This value is normally required only when ACF2 is the indigenous ESM. If ACF2 has already been customized (for the MAINVIEW Security Service interface prior to release 6.0) to process RACROUTE invocations using a different SAF SUBSYS, such as BBI3, then that value may still be used, and should be specified here.

The following values can be specified:

<table>
<thead>
<tr>
<th>SAFSUBSYS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;SSID</td>
<td>The MVS SSID that was specified in this initialization member, or was provided as a default by RTCS.</td>
</tr>
<tr>
<td>&amp;PRODUCT</td>
<td>The internal (security) product name in whose product address space an ESM security environment is being created (via SIGNON or INHERIT).</td>
</tr>
<tr>
<td>'' or ' '</td>
<td>Null, or one or more blanks: No SUBSYS= parameter is to be used on any SAF RACROUTE invocation.</td>
</tr>
<tr>
<td>'xxxxxxxx'</td>
<td>A specific RACROUTE SUBSYS= value, consisting of up to 8 characters.</td>
</tr>
</tbody>
</table>

Under most circumstances, RACF and Top Secret will effectively ignore the RACROUTE SUBSYS= specification. However, a value may still be usefully provided here, since the SUBSYS can be used to filter requests to be traced when using the Top Secret SECTRACE facility (as well as the ACF2 SECTRACE facility), and to subset RACROUTE requests to be traced if an IBM or OEM vendor facility (using RACF exits) has been installed to facilitate such tracing. So, a suitable value should be provided, even if RACF or TSS is in use, simply in support of any existing ESM-level diagnostic tracing facility.

* Default ESM RACROUTE system entry validation APPL ID
**Appendix D** RTCS initialization member

+ SECURITYAPPLID=&PRODUCT

This parameter specifies the RACROUTE APPL (application name) that is to be used as the default SAF APPLication by the RTCS Security Manager (if none is specified by the product or the caller does not have an RTCS product definition, or the definition in the product or System Registry is null or blanks).

The following values can be specified:

<table>
<thead>
<tr>
<th>SAFSUBSYS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;SSID</td>
<td>The MVS SSID that was specified in this initialization member, or was provided as a default by RTCS.</td>
</tr>
<tr>
<td>&amp;PRODUCT</td>
<td>The internal (security) product name in whose product address space an ESM security environment is being created (via SIGNON or INHERIT) or a resource being authorized.</td>
</tr>
<tr>
<td>&amp;PRDAPPL</td>
<td>The default product security APPL name, usually specified in the RTCS product definition in the package, or via the RTCS System Registry product, context, or server (instance) definition.</td>
</tr>
<tr>
<td>'' or ' '</td>
<td>Null, or one or more blanks: No APPL= parameter is to be used on any SAF RACROUTE invocation.</td>
</tr>
<tr>
<td>'xxxxxxxx'</td>
<td>A specific RACROUTE APPL= value, consisting of up to 8 characters.</td>
</tr>
</tbody>
</table>

== Runtime Component System (RTCS) Initialization Parameters ==
Index

A
abending 71
ACCEPT parameter value 85
ACCEPTDEFAULTUSERID parameter 85
ACF2 57, 83, 84
ACID parameter value 85, 86
ACTIVE parameter 87
ALWAYS parameter value 86
AOSC70ID 31
AOSZCNTLN 31
AOSZHTML 31
AOSZLINK 31
AOSZRTCS 31
AOSZRXML 31
APF-authorization
  permanent 48
  product program library 48
  subsystem program library 48
  temporary 48
APPL, default name 85
APPLID parameter value 85
ASIS parameter value 88
AUTO parameter value 84

REFRESH,KERNEL 76
REFRESH,SECURITY 77, 86
refreshing Global Security Parameters 77, 86
SECTRACE 77, 86
START 66, 72, 78, 81
STOP 73
WARNMSG 77, 78
comments 96
COMMNDxx member 66
common area data spaces, specifying maximum 51
COMPLETE parameter value 86
components, RTCS 19
configuration
  introduction 40
  manual 41
  modifying 41
continuous system operation 18
control statements
  description 96
  errors 96
  IMPORT 97
  OPTIONS 98
  SET 96
conventions, documentation 15
CSCB name
  batch job 75
  description 75
  example 75
  Generalized Server address space 75, 81
  started task 76
  subsystem address space 75
CUST records 84, 86, 88
customer support 3
customizing Global Security Parameters 86

B
BASE records 84
batch job, CSCB name 75
blank characters, in SET statement 97
BMC Product Authorization Table library, allocating 50
BMC Software, contacting 2

C
CA ACF2 57, 83, 84
CA Top Secret 57, 63, 83
CLASS
  active, indicating 87
  description 87
  SAF resource name 87
CLASS parameter 87
Command Scheduling Control Block. See CSCB name commands
  MODIFY 75
  operator 75

CA data definition names. See ddnames
data set names
  specifying in IMPORT control statement 97
Data-In-Virtual 21, 22, 49
DD statements
  IEFPDSI 54, 55, 56
  REGISTRY 35, 103
  STEPLIB 33, 35, 54, 103

Index 119
DD statements (continued)
  SYSIN 96
  SYSPRINT 96
  TASKLIB 35, 103
ddnames
  required 95
  specifying in IMPORT control statement 97
  SYSIN 96
  SYSPRINT 96
DEFAULTUSERID parameter 85
DESC parameter 87
diagnostic trace messages 77
DISP=OLD 32
distribution libraries
  description 31
  illustration 30
  list 31
DIV 21, 22, 49
DLL
  date and time stamp 19
  refreshing packages 19, 77

E
electronic documentation 13
ENTITY name length 88
ESM (external security manager)
  specifying type 84
  start, when to 71
  user ID 57
ESMTYPE 84
EXEC parameter 36
EXTENDED parameter value 86

G
GCPs
  customizing 88
  description 87
  parameters 87, 88
Generalized Server
  assigning ESM user IDs 57
  messages example 76
Generalized Server address spaces
  code, executing 22
  CSCB name 75, 81
  description 22
  multiple 22
  protection keys 22
  sample JCL 36
  starting new 78, 81
  warning messages 77, 78
Generalized Server started task PROC
  creating 56
  default value 105
example 105
location 105
naming 56
specifying 105
Global Class Properties
  customizing 88
  description 87
  parameters 87, 88
Global Security Parameters
  APPLID 85
  customizing 86
  description 84
  parameters
    DEFAULTUSERID 85
    ESMTYPE 84
    GROUPINHERIT 86
    SAFSUBSYS 85
    SECTRACE 86
    UNDEFINEDUSERINHERIT 85
    UNDEFINEDUSERSIGNON 85
  refreshing 77, 86
Global Security Parameters. See GSPs
  GROUP IDENT 86
  GROUPINHERIT parameter 86
  GRSples 33
H
Help, online 13
HTML documents, library in 34
hypertext document library (TOSZHTML)
  allocation 34
  creating 34, 46
  description 31, 34
  sample JCL 47
  used, when 34
I
ICF catalog 43, 45, 47, 49
IEASYSn member 52
IEASYSxx 51
IEFPDSI DD statements 54, 55, 56
IEFSSNxx member 53
IMPORT control statements 97, 100
importing
  data set name, using 97
  ddname, using 97
Initiator started task PROC
  creating 53
  customizing 53
  default values 102
  example 102
  location 102
  naming 54
Initiator started task PROC (continued)
  sample JCL 35
  start when to 102
  validation procedures 103
Initiator, assigning ESM user IDs 57

J

J operand 81
JCL samples
  hypertext document library, to create 47
  library with 35
  product program library, to create 45
  Registry Import Utility
to execute 37, 95
RTCS Generalized Server address space started task PROC 36
RTCS Initiator started task PROC 35
RTCS subsystem started task PROC 36
subsystem program library, to create 43
JES, primary 56, 71
JES-managed started task PROCLIB 55, 56

K

kernel
  date and time stamp 19
  description 20
  maintenance 19
  refreshing 19, 76
  updating 76
kernel and DLL refresh facility 19
key, registry 94

L

libraries
  categories 30
  distribution 30, 31
  hypertext document library (TOSZHTML) 34, 46
  illustration 30
  JES-managed started task procedure library 55
  Master Subsystem procedure library 55
  Product Authorization Table library 50
  product program library (TOSZLINK) 45
  production 30, 32
  runtime 30
  sample JCL and parameter library (TOSZCNTL) 35
  subsystem program library (TOSZRTCS) 33, 42
target 30, 31, 33
  XML document library (TOSZXML) 35
logging parameter 88
LOGONID parameter value 85, 86
LOGOPT parameter 88
LXs 51

M

MainView Customization
  field descriptions 90
Master Subsystem procedure library 55
MAXCAD parameter 51
MAXLEN parameter 88
messages
diagnostic trace 77
documentation 100
Generalized Server example 76
generating list output of RIU 96
RIU, generated by 100
Subsystem starting 72
valid range 100
warning 77, 78
MODIFY command 75
MSTJCL00 54, 55, 56

N

NEVER parameter value 86
NOAUTH parameter value 88
NOFAIL parameter value 88
NOLOG parameter value 88
NONE parameter value 84, 86
NOPRETEND parameter value 98
NOTRACE parameter value 98
NOTRACEX parameter value 98
NSYSLX parameter 51

O

objects, PM3 program 33, 34
online Help 13
operator commands 75
OPTIONS control statements
default values 98
description 98
multiple values 98
NOPRETEND 98
NOTRACE 98
NOTRACEX 98
PRETEND 98
TRACE 98, 100
TRACEX 98, 100
OSZ$EXEC
copying 56
customizing 36
description 36
locations 36
naming 36
OSZ$INIT
copying 54
customizing 35
OSZ$INIT (continued)
  default values 35
description 35
location 35
naming 35
OSZ$REGI
  customizing 37
description 37
location 37
OSZ$RTCS
  copying 55
customizing 36
default values 36
description 36
location 36
naming 36
OSZEXEC 36, 56
OSZEXEC8 95
OSZEXECn 34, 47
OSZINIT 35, 54
OSZJSGCP 88
OSZJSGSP 86
OSZLINK 48
OSZMOSYS 22, 43, 45, 49
OSZRGIMP 94
OSZRSGCP 88
OSZRSGSP 86
OSZRTCS 36, 48, 55
OSZSCHED 37
OSZSCHED member 52
OSZSIRIS 55

P

parameter input members, in sample library (TOSZCNTL) 35
parameters
  ACTIVE 87
  APPLID 85
  CLASS 87
  DEFAULTUSERID 85
  DESC 87
  ESMTYPE 84
  EXEC 36
  GROUPINHERIT 86
  LOGOPT 88
  MAXCAD 51
  MAXLEN 88
  SAFSUBSYS 85
  SECTRACE 86
  TSSCLASS 87
  UNDEFINEDUSERINHERIT 85
  UNDEFINEDUSERSIGNON 85
  path, registry 94
  PM3 program objects 33, 34
  PPT

  entries to add 37
  refreshing 52
  sample set of entries 37
  updating existing SCHED member 51

  PPT table additions 105
  PRETEND option 98
  primary JES 56, 71
  product authorization tables library 50
  product program library (OSZLINK)
    APF-authorizing 48
    creating 45
description 31, 34
  sample JCL 45
  product support 3
  production copies 31
  production libraries
    applying maintenance 19, 32
    benefits of using 32
description 30, 32
  illustrated 30
  runtime 32
  updating 19

  Program Properties Table additions 105
  protection keys, in Generalized Server address spaces 22
  PTF APPLY 31

R

RACF 57, 83, 84
RACROUTE 85
records
  BASE 84
  CUST 84, 86, 88
  REFRESH,KERNEL command 76
  REFRESH,LIBRARY command 77
  REFRESH,SECURITY command 71, 77, 86
  refreshing
    DLL packages 19, 77
    Global Security Parameters 77, 86
description 76
  kernel 76
  PPT 52
  REGISTRY command 78
  REGISTRY DD statement 35, 103
  registry import file 97
  registry key 94
  registry path 94
  registry value 94
  REJECT parameter value 85
return codes descriptions 99
RIU
  changing behavior 98
  control statements 94
ddames
    SYSIN 96
    SYSPRINT 96
description 94
RIU (continued)
diagnostic information 98, 100
executing 95
IMPORT control statement 97
job stream example 95
messages 98, 100
OPTIONS control statement 98
required data definition names 95
sample JCL to execute 37
SET control statement 35, 94, 96
started by 95
testing import 98
RTCS
components 19
configuration, introduction 40
configuration, manual 41
configuration, modifying 41
test address space 20, 52
description 18
kernel 20
starting 54, 66, 71
RTCS Allocator (OSZMOSYS) 22, 43, 45, 49
RTCS product registry description 22
RTCS Security Manager 83
RTCS subsystem
DLL packages, refreshing 77
kernel, refreshing 76
warning messages 77, 78
RTCS system registry
backing up 70
customizing Global Class Properties 88
customizing Global Security Parameters 86
data accessibility 21
description 21, 94
DSNAME, selecting 49
end-user profiles 70
example definitions 23
importing files 97
initializing 77
initializing problems 71
recovering 70
recreating data 70
retaining original input 70
VSAM LDS, backed by
-calculating size 49
-naming 49
Runtime Component System. See RTCS
runtime libraries
description 31
illustrated 30
listed 31

S
SAF resource CLASS name 87
SAFSUBSYS parameter 85
sample JCL and parameter library (TOSZCNTL)
allocation 35
description 31, 35
members 35, 51
production copies 32
used, when 35
SCH member suffixes 52
SCOPE=COMMON 51
SECTRACE command 77, 86
Security Manager
description 83
Global Class Properties default 87
Global Security Parameters default 84
reinitializing 77
trace messages 77
SET command 78
SET control statements 96
sharing PDSEs
in a sysplex 33
requirements 33
SIMPLE parameter value 86
SSID
guaranteeing availability 53
selecting 52
START command 66, 72, 78, 81
started task, CSCB name 76
starting
Generalized Server address spaces 78, 81
RTCS 66, 71, 81
starting RTCS 71
STELIB DD statement 33, 35, 54, 103
stopping RTCS 73
subsystem address space, RTCS
allocating 21
commands 75
CSCB name 75
data sets, allocated 20
description 20
maintenance 18
operator commands 75
responsibilities 20
RTCS system registry VSAM LDS 22, 49
subsystem ID (SSID) 52
Subsystem Name (SSN) Table
adding SSIDs 53
subsystem program library (TOSZRTCS)
allocation 33
APF-authorizing 48
creating 33, 42
description 31, 33
sample JCL 43
used, when 33
Subsystem started task PROC
creating 55
default values 104
example 104
location 104
Subsystem started task PROC (continued)
  naming 55
  sample JCL 36
  specifying 104
  when to start 104
Subsystem, RTCS
  assigning ESM user IDs 57
  messages, starting 72
  starting, when to 66
  subsystem, RTCS
    updating 18
support, customer 3
syntax statement conventions 15
SYSIN DD statement 96, 99
SYSIN input streams 94
SYSMOD ACCEPT 31
SYSMOD APPLY 31
  sysplex
    sharing PDSEs 33
SYSPRINT DD statement 96, 99
system linkage indexes (LXs) 51

T
  target libraries
    description 33
    illustration 30
    list 31
    maintenance, applying 19
  TASKLIB DD statement 35, 103
  technical support 3
  terminating abnormally 71
Top Secret 57, 63, 83
TOSZCMTL
  allocation 35
  description 31, 35
  members 35, 51
  production copies 32
  used, when 35
TOSZHTML
  allocation 34
  creating 34, 46
  description 31, 34
  sample JCL 47
  used, when 34
TOSZLINK
  allocation 34
  APF-authorizing 48
  creating 45
  description 31, 34
  sample JCL 45
  used, when 34
TOSZRTCS
  allocation 33
  APF-authorizing 48
  creating 33, 42
  description 31, 33
  sample JCL 43
  used, when 33
TOSZRXRL
  allocation 35
  description 31, 35
  production copies 32
  used, when 35
TRACE option 98, 100
TRACEX option 98, 100
TSS 63, 83, 84
TSS RDT CLASS name 87
TSSCLASS parameter 87

U
  UNDEFINEDUSERINHERIT parameter 85
  UNDEFINEDUSERSIGNON parameter 85
  updating
    DLL packages 77
    kernel 76
  user IDs
    default 85
    ESM 57
    undefined 85
  user profile information, recreating 70
  USERID parameter value 85, 86

V
  value, registry 94
  VOLSER name 48
  VSAM LDS, for RTCS system registry
    calculating size 49
    description 21
    recovering 70
    selecting 49

W
  warning messages
    issuing 77, 78
    suppressing 77, 78
  WARNMSG command 77

X
  XML document library (TOSZRXRL)
    allocation 35
    description 31, 35
    production copies 32
    used, when 35
Z

z/OS Program Properties Table additions 105
Third-Party Product Terms

The following terms apply to third-party products that are included with or in a BMC Software product as described in the BMC Software, Inc., License Agreement that is applicable to the BMC Software product.

The Apache Software License, Version 1.1

This product includes the Apache software product found at www.apache.org and the Apache software product is distributed to us pursuant to the following terms and conditions:

Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution if any, must include the following acknowledgement: "This product includes software developed by the Apache Software Foundation (http://www.apache.org/)." Alternately, this acknowledgement may appear in the software itself, if and wherever such third-party acknowledgements normally appear.

4. The names 'Apache', and 'Apache Software Foundation' must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name without prior written permission of the Apache Group.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is
You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

A. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

B. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

C. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

A. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

B. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

C. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program, or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.
8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) <year> name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.