SailPoint Connector for RACF

Version 4.0.01

Rev 1.4

Administration Guide

Copyright and Trademark Notices

Copyright © 2023 SailPoint Technologies, Inc. All Rights Reserved.

All logos, text, content, including underlying HTML code, designs, and graphics used and/or depicted on these written materials or in this internet website are protected under United States and international copyright and trademark laws and treaties, and may not be used or reproduced without the prior express written permission of SailPoint Technologies, Inc.

“SailPoint Technologies,” (design and word mark), “SailPoint,” (design and word mark), “Identity IQ,” “IdentityNow,” “SecurityIQ,” “Identity AI,” “Identity Cube,” and “SailPoint Productive Identity” are registered trademarks of SailPoint Technologies, Inc. “Identity is Everything,” “The Power of Identity,” and “Identity University” are trademarks of SailPoint Technologies, Inc. None of the foregoing marks may be used without the prior express written permission of SailPoint Technologies, Inc. All other trademarks shown herein are owned by the respective companies or persons indicated.

SailPoint Technologies, Inc. makes no warranty of any kind regarding these materials, or the information included therein, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. SailPoint Technologies shall not be liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Patents Notice. https://www.sailpoint.com/patents

Restricted Rights Legend. All rights are reserved. No part of this document may be published, distributed, reproduced, publicly displayed, used to create derivative works, or translated to another language, without the prior written consent of SailPoint Technologies. The information contained in this document is subject to change without notice.

Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 for DOD agencies, and subparagraphs (c)(1) and (c)(2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies.

Regulatory/Export Compliance. The export and re-export of this software is controlled for export purposes by the U.S. Government. By accepting this software and/or documentation, licensee agrees to comply with all U.S. and foreign export laws and regulations as they relate to software and related documentation. Licensee will not export or re-export outside the United States software or documentation, whether directly or indirectly, to any Prohibited Party and will not cause, approve or otherwise intentionally facilitate others in so doing. A Prohibited Party includes: a party in a U.S. embargoed country or country the United States has named as a supporter of international terrorism; a party involved in proliferation; a party identified by the U.S. Government as a Denied Party; a party named on the U.S. Department of Commerce’s Entity List in Supplement No. 4 to 15 C.F.R. § 744; a party prohibited from participation in export or re-export transactions by a U.S. Government General Order; a party listed by the U.S. Government’s Office of Foreign Assets Control as ineligible to participate in transactions subject to U.S. jurisdiction; or any party that licensee knows or has reason to know has violated or plans to violate U.S. or foreign export laws or regulations. Licensee shall ensure that each of its software users complies with U.S. and foreign export laws and regulations as they relate to software and related documentation.

Contents

Integrating SailPoint with RACF Source

Revised Date: 16 January 2024

> **Note**
>
> IdentityIQ connector information is now available as online help and PDF. The online help describes the latest updates for the connector.
>
> To find documents related to a specific version of IdentityIQ, refer to the Supported Connectors for IdentityIQ page on Compass.
>
> Configuration details for connectors may vary not only by release version but also by patch version. Be sure to refer to the correct documentation for your specific release and patch level.

This document is designed to give specific information about the requirements and field definitions needed to get a working instance of a RACF Source connector.

Operating as a security administration connector running on various platforms in the enterprise, connector processes and transfers security commands and data between managed systems and SailPoint.

SailPoint Connector for RACF includes the following features:

- Full aggregation
- Provisioning
- Monitoring of RACF activities to update SailPoint in real time with User and password changes

The following topics are discussed:

Connector Facilities

The Connector facilities enable the Managed System (MS) to be monitored and managed by SailPoint. The Connector facilities include:

- **Managed System data aggregation to SailPoint**: The data aggregation procedure which is initiated from the SailPoint is controlled by the Connector. The Connector ensures that relevant data is aggregated from the RACF Managed System database to the SailPoint database. After all the data of the RACF Managed System has been aggregated, a consolidated picture of SailPoint data can be viewed in SailPoint.

- **Translation and execution of SailPoint commands**: Security-related commands (for example, add user, change password or phrase) which are initiated by SailPoint are handled by the Connector. The Connector translates these commands into the format and language recognized by the Managed System and executes them in the Managed System.

- **Managed System activity monitoring**: The Connector intercepts events that occur in the managed system which are initiated from within the platform environment. For example, the MS administrator adds, modifies, or deletes MS users and groups or an MS user changes their password or Managed System administrator changes a password for a user. When a significant event occurs, either data defining the event or an up-to-date updated entity is sent by the Connector to IdentityIQ. This functionality is accomplished using the Online Interceptor component.

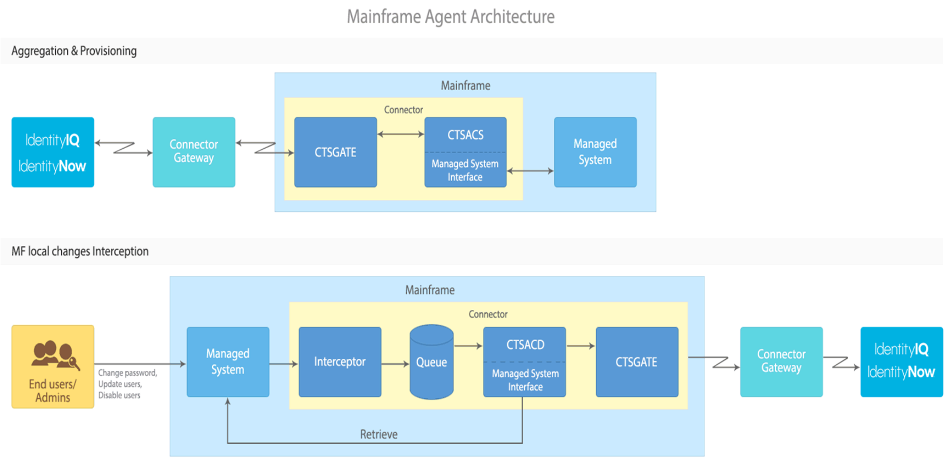
  > **Note**
  > Online Interceptor requires IdentityIQ setting. For more information, see SailPoint Quick Reference Guide for Gateway Connectors.

- **Secured Communication**: Secured communication can be performed externally using AT-TLS or internally using Transmitted Data Encryption. For more information, see Secured Communication.

- **Stored Data Encryption**: All sensitive data which is stored temporarily in Connector for RACF (for example, sensitive security information that is written to the Connector queue file) is encrypted using a stored data encryption key.

Connector Components in Detail

The following figure illustrates the major components of the Connector, their relationship with one another, and the flow of data between them. This diagram represents the connection between SailPoint and a single Connector installation with a single MS. In practice, multiple MS on different platforms can be administered by multiple Connector installations.



Component	Description
Connector Gateway	Resides between SailPoint and Mainframe Connector (CTSGATE) and is responsible for the communication between these two components.
Connector	Enables the interception of managed system events and the translation of SailPoint commands to each specific managed system terminology. The Managed System Interface component of the Connector is a flexible API which is customized for each managed system.
CTSGATE	Mainframe side communicator gateway. Responsible for communication with Connector Gateway and CTSACS /CTSACD. It is also responsible for starting and stopping CTSACS and CTSACD.
CTSACS	Transaction Server - is responsible for SailPoint transactions handling. **Note** May be 1 to 3 transaction servers.
Managed System Interface	Responsible on the interface with RACF itself. It translates SailPoint transactions into RACF commands (provisioning transactions). It uses RACF's API to aggregate RACF's entities from RACF to provisioning module.
Managed System	RACF
CTSACD	**Notification Server** - Reads events written to Queue by Interceptor, retrieve relevant entity up-to-date status from RACF and pass entity data to CTSGATE.
Interceptor	Responsible for intercepting Mainframe local changes done by RACF administrators and end-users and writes them to Queue. Two types of interceptors can be used in the Connector: • **Online Interceptor**: Detects security administration events as they occur. • **Offline Interceptor**: Detects security administration events in batch.

Component	Description
Connector Queue	The Connector queue is a dataset in which all security data is saved before it is sent to SailPoint via the Notification Server. If communication between Connector and SailPoint fails, Managed System events continue to be stored in the Connector queue and are sent to SailPoint when communication is re-established.

Installation

This chapter provides the required information and step-by-step instructions involved in the installation of the RACF Mainframe connector.

It is recommended that before beginning the installation procedure, you review the content of Before Installing the Connector section, installation steps in this chapter and the contents of RACF Support Customization

The following topics are discussed in this chapter:

Before Installing the Connector

Read the following topics before starting the connector installation.

Hardware and Software Requirements

The following table outlines the software and hardware requirements to support the RACF connector.

Component	Description
Computer	The Connector operates on any hardware configuration supported by any of the supported operating systems. • Supported systems: This connector can be used to manage RACF on z/OS release 2.4 through 2.5. Software requirements are as follows: • Job Entry Subsystem: JES2 or JES3 • TSO/E • SMP/E • TCP/IP • ISPF or any other text editor that allows submitting jobs and checking their output

Component	Description
Disk Type	The Connector datasets may reside on any disk that is supported by MVS. For example, **3390** IBM disk type is supported.
Disk Space	700 cylinders of a 3390 device are sufficient to store the datasets installed by the Connector installation procedure.

> **Caution**
>
> Any Mainframe tool which handles x37 abends should be avoided or disabled when using the SailPoint Mainframe connector.
>
> If the tool allows it, you should add SailPoint Mainframe connector to the tool's exclude list.

Pre-Installation Considerations

This section describes the issues to be considered before beginning the installation.

Communication Parameters Coordination

For the RACF Connector to communicate successfully with a provisioning module (SailPoint), several parameters must be coordinated between provisioning module's Application or Source Definition, Connector Gateway and RACF Connector.

The table below summarizes these parameters. Each row in the table describes a set of parameters in all or some of the components which must be coordinated. For more information and description of Connector Gateway parameters, see *SailPoint Integration Guide* or *SailPoint Quick Reference Guide for Gateway Connectors* depending on Connector Gateway release. For full description of IdentityIQ Application definition, see *SailPoint IdentityIQ Administration Guide*. For full description of IdentityNow Source definition, see *RACF Source Configuration for IdentityNow*.

Summary of Required Parameter Coordination

For Connector to communicate successfully with SailPoint, the following Connector installation/environment parameters must be coordinated with parameters specified in the Connector Gateway and in SailPoint:

Parameter name	RACF Connector	Connector Gateway	IdentityIQ Application definition	IdentityNow Source definition
RSSNAME	MSCS in RSSPARM member in PARM lib-		MSCS Name parameter in Connector Gateway/Connec-	Connector name in RACF source

Parameter name		RACF Connector	Connector Gateway	IdentityIQ Application definition	IdentityNow Source definition
		rary		tor Manager Settings	
RSSTYPE		RSSP_TYPE in RSSPARM member in PARM library. Must be RACF		MS Type selected for Application Type field should be RACF - Full	RACF source
MF_PORT		PORT parameter in ECAPARM member in PARM library	**port** parameter in SM section in **init.xml** file.		
SECURED	TRANSMITTED DATA ENCRYPTION			Encryption parameter in Connector Gateway/ Connector Manager Settings	Not supported
	TLS	AT-TLS implementation	Implementation steps have to be performed	Implementation steps have to be performed	Implementation steps have to be performed

- **RSSNAME**: In RACF Connector, the name is set in the **DEFPARMS %RSSNAME%** parameter in the INSTALL library during installation. After installation, this name is automatically set as the MSCS name in RSSPARM member in the PARM library. The MSCS name appears in column 1 of each parameter line (unless **ALL_RSS** is set in the line).

  This name can be up to 32 characters long. However, SailPoint recommends that you use RSSNAME configurations with eight characters or fewer. If you must configure the RSSNAME parameter with more than eight characters, special adjustments must be performed at the end of the RACF Connector installation procedure. For more information on the additional steps, see Step 11 – Adjust for Longer Managed System Names.

  The same name must be specified for **%RSSNAME%** and for the MSCS Name parameter in Connector Gateway/Connector Manager Settings in Provisioning Module Application Definition (for IdentityIQ) or for the Connector name in RACF Source definition (for IdentityNow).

- **RSSTYPE**: During installation RSSTYPE is specified in the DEFPARMS **%RSSGTYPE%** parameter in the INSTALL library. After installation, the RSSTYPE can be found in the RSSPARM **RSS_TYPE** parameter in the PARM library. For RACF connector, the value must be RACF. The MS Type in Provisioning module Application definition must be RACF- Full.

- **MF_PORT**: TCP/IP port number defined for the RACF Connector for communication with Provisioning Module. Connector for RACF's CTSGATE uses two consecutive TCP/IP ports to communicate with Provisioning Module. By default, the ports used are 2470 and 2471. Verify that these ports are not already in use. If they are in use, locate two other consecutive ports which are available.

  Specify the lower of these two ports when you are instructed to provide a value for parameter PORT during the Connector installation. The same port number must be specified in the port field in the SM section of the Connector Gateway **init.xml** file. For more information, see Step 9 – Customize Communication Settings.

- **SECURED_COMMUNICATION**: Secured communication can be implemented by using Transmitted Data Encryption or TLS. One of the following options can be selected. When TLS is selected, Transmitted Data Encryption must be set off in all components.

  - **TRANSMITTED DATA ENCRYPTION**: Communication security is gained by encrypting the transmitted data using an encryption key dataset.

    For more information, see 9.4 – Set Up Secured Communication.

  - **TLS**: Communication is secured using TLS. Requires implementation of steps in all components.

    In the Mainframe, TLS communication must be configured using AT-TLS. With AT-TLS, the TLS processing is performed by TCP/IP and is transparent to the application (CTSGATE). Hence no settings are required in Connector for RACF parameters, except for setting the Transmitted Data Encryption to **Off** as described above.

    For more information, see Secured Communication.

RACF Considerations

Protecting Temporary Datasets

Temporary data sets are considered protected from any access except by the job or session that created them, and hence are not required to be protected by RACF. These files are allocated as new files, held with exclusive SYSDSN ENQ. However in situations like system failure, a temporary data set could be left unprotected. Such file can be accessed by any user, unless protected by RACF.

The Connector for RACF handles the temporary file allocated by the EXECOUT DD statement in a special way, which causes failures due to security violation when temporary files are protected by RACF.

The EXECOUT file is defined in the connector procedure and is created when the connector starts, under the connector User ID. But, when processing requests received from SailPoint, this file is accessed by the Connector for RACF under the Managed System Administrator User ID. When the temporary files are protected by RACF, the only user that can access temporary files is the user that allocates them. So, when the connector tries to write to the file under the Managed System Administrator User ID. If this User ID does not have permission to access the file, RACF fails the request.

During installation, the EXECOUT DD statement allocates the file on VIO. Temporary datasets allocated to VIO are not protected by the RACF so no error occurs. But, if VIO is not used in the system, or if VIO is not allowed for large files, the problem occurs.

The above problem can be prevented as follows:

- Allow the files allocated by the Connector for RACF to be on VIO.

  Or

- Allocate the EXECOUT DD statement to a permanent file. The Connector for RACF utilizes source JCL for started tasks (STCJOB) for starting the procedures to maintain these permanent datasets.

To use this option, set the name of the STCJOBs library (defined in IEFJOBS DD statement in MSTJCLxx system parameter) to which Connector for RACF STCJOBs would be copied, as the value of the **%STCJOBS%** DEFPARMS parameter.

Installation Considerations

Enhanced Data Integrity Considerations

If Enhanced Data Integrity function is active, ensure that all Connector for RACF files are set in its Exclude list. For more information, refer to the following:

- *z/OS DFSMS Using Data Sets*
- ‘IFGPSEDI (enhanced data integrity)’ section in *z/OS Initialization and Tuning Reference Guide*.

Using STCJOBs

STCJOBS can be used to start the Connector for RACF started tasks. They are required when temporary datasets are protected, but can be used regardless of this protection.

To use STCJOBs, specify the name of the system library for source JCL for started tasks (STCJOB) to which the Connector for RACF STCJOBs would be copied. This library must be defined in the IEFJOBS DD statement in the MSTJCLxx system **parmlib** member.

The STCJOBs would be copied to this library, while renamed using the DEFPARMS **%PROCPREFS%** value as the first 3 characters of their names.

When using STCJOBs, the Connector for RACF started tasks procedures can be placed in one of the following libraries, according to DEFPARMS **%PROCLIB%** parameter value:

- Copy the Connector for RACF started tasks procedures to the system JCL procedures library (PROCLIB). When this option is used, the name of the system JCL procedures library, must be specified in the DEFPARMS **%PROCLIB%** parameter. The started tasks procedures would be copied to this library and renamed using the **%PROCPREFS%** value as the first 3 characters of their name. When the STCJOBs are started, the system would search the standard procedure libraries for the started tasks procedures.

  Or

- Leave the Connector for RACF started tasks procedures in the Connector PROCLIB and use the JCLLIB JCL statement to point to this library. When this option is used, LOCALCOPY must be specified in the DEFPARMS **%PROCLIB%** parameter. The procedures are copied to the Connector PROCLIB library and renamed using the **%PROCPREFS%** value as the first 3 characters. When the STCJOBs are started, the system would search for the procedures using the JCLLIB statement in the STCJOB.

> **Note**
>
> When STCJOBs are used:
>
> - The files allocated by EXECOUT DD statements which are temporary by default, are allocated as permanent files.
> - The IGD17054I message could be displayed which is not an error message and can be ignored.
>
> In z/OS V1R13, issuance of the IGD17054I message is controlled by the value specified for the SUPPRESS_DRMSGS parameter, in the IGDSMSxx PARMLIB member. Beginning in z/OS V2R1, issuance of the IGD17054I message is controlled by the new SUPPRESS_SMSMSG parameter, also in the IGDSMSxx PARMLIB member. For more information, see *z/OS Initialization and Tuning Reference Guide* of the appropriate z/OS version.

New Installation

This section describes the procedures involved in creating a new installation of Connector for RACF.

Installation Summary

The installation of Connector for RACF consists of the following procedures:

The following table summarizes the procedures for installing Connector for RACF.

New Installation Procedure – Summary

Procedure	Step	Job/Member Name	Description
1	**Set the Parameter Values**		
2	**Prepare Installation IMAGE from TRS file**		
	2.1		Transfer the INSTALL.TRS file Using FTP binary
	2.2		UNCOMPRESS the TRS File
	2.3	$RECEIVE	Tailor the $RECEIVE Job
	2.4	$RECEIVE	RECEIVE the Installation **IMAGE**
3	**Allocate and Load Connector INSTALL Library**		
	3.1	$LOADINS	Copy, edit and run member $LOADINS
4	**Allocate and Load Connector for RACF Installation libraries**		
	4.1	LOADCTS	Tailor member LOADCTS
	4.2	LOADCTS	Submit job to allocate and load Connector for RACF libraries
5	**Tailor Connector for RACF members with site parameters**		
	5.1	DEFPARMS	Assign installation parameters
	5.2	CHNGEPRS	Modify Connector for RACF members
6	**Copy Connector for RACF procedures, STCJOBs and others members**		
	6.1	CPYMRACF	Submit the CPYMRACF job to perform the copy
7	**Customize Connector for RACF installation parameters**		
	7.1	CTSPARMJ	Create CTSPARM module
	7.2	RSSPARM	Assign RSSPARM parameter values
8	**Format Connector for RACF datasets**		
	8.1	FORMCTS	Edit and Run Member FORMCTS
9	**Customize Communication Settings**		
	9.1		Verify TCP/IP connectivity
	9.2	ECAPARM	Connector for RACF Gateway Communication parameters
	9.3		Define TCP/IP DATA file
	9.4		Set up secured communication

Procedure	Step	Job/Member Name	Description
10	**Define Connector for RACF in RACF**		
	10.1	CTSRACF	Define Connector for RACF Started Tasks in RACF
	10.2	CTSRACF	Set Permissions to Connector Datasets
	10.3	CTSRACF	Protect the Encryption Key Datasets
	10.4	CTSRACF	Define an OMVS Segment
	10.5	CTSRACF	Grant CTSGATE with authority to use TCP/IP stack
11	**Adjust for Managed System names longer than 8 characters**		
12	**Adjusting Managed System Administrator attributes**		
	12.1	Provide Managed System Administrator Passwords	
	12.2	Verify Managed System Administrator permissions	
13	**Add Connector for RACF Libraries to the MVS Authorized Libraries List**		
14	**Review the Installation**		
	14.1		Verify that the Connector for RACF Gateway is installed properly
	14.2		Verify that the RACF Connector communicates successfully with SailPoint
	14.3		Verify RACF Connector RACF interface
15	**(Optional) Local changes migration**		
16	**Configure automated startup of Connector for RACF**		
17	**Customizing RACF Support**		
18	**Post Installation Checks**		

Step 1 – Set the Parameter Values

This section lists the various tables containing all the parameters for which values must be set.

Set the **Value** column with the selected values, which would later be used to set the installation jobs and parameters.

Datasets Allocation Considerations

The first step of the installation process creates **IMAGE** files from which the Connector files are loaded by the later steps.

Ensure that you select different prefix + version combinations to the **IMAGE** files (**%instpref%** in the Installation Upload and IMAGE Datasets Parameters table below) and to the **Connector** files (xPREFx + xVERx in the Connector for RACF Datasets Allocation Parameters table). Otherwise the installation fails due to duplicate datasets.

When selecting high level qualifiers for the files, ensure the product installer has ALTER authority for these files.

Some of the parameters are the unit name and volume serial number to be used for product datasets allocation. If the datasets must be SMS managed, leave these parameters empty (except for SPCVOL which must be *). If the datasets must not be SMS managed, specify the values for unit and volume serial number to ensure proper handling of the file.

Installation Upload and IMAGE Datasets Parameters

Parameter	Description	Value
Upload dataset name	The name of the dataset into which the product would be transferred using FTP.	
%xmitlib%	Name of the library into which the Connector for RACF uploaded file would be uncompressed.	
%instpref%	Prefix selected for Connector for RACF installation **IMAGE** datasets that are later used to install Connector for RACF.	
%UNIT%	For non-SMS managed datasets specify UNIT(unitname) where unitname is the name of DASD unit where the Connector for RACF Installation **IMAGE** datasets would be placed. For SMS-managed datasets, specify null	
%VOLUME%	For non-SMS managed datasets specify VOLUME(volser) where volser is the volume serial number on which Connector for RACF Installation **IMAGE** datasets would be placed. For SMS-managed datasets, specify null.	

Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs)

Considerations before setting Connector for RACF file allocation parameters:

Each Connector for RACF file and library used to install and operate the product is assigned a type according to its usage. Each type has separate allocation parameters (prefix (HLQ), version, unit, volser) and is identified by a prefix assigned to its allocation parameters variables.

File types are:

- Installation - IL
- Operation – OL

- SMP/E CSI – SPC
- SMP/E files - SPA
- DLIBs – SPD

Assign values to the allocation parameters in the following table according to the file types and site standards.

- Some sites use special naming conventions for Load Module libraries. Therefore, the name of the Connector for RACF Load library is a user-defined parameter.
- The datasets installed by the Connector for RACF installation procedure are described in Appendix C: Connector for RACF Datasets and JCL Procedures. The total DASD space they require is listed in Hardware and Software Requirements.

Connector for RACF Datasets Allocation Parameters

Parameter	Description	Default Value	Value
DLPREFS	Installation **IMAGE**datasets prefix. The value of this parameter should be the same value specified for %**instpref**% above.	-	
JOBNAME	Job name prefix (1 to 6 characters) to be used for jobs submitted during the Connector for RACF installation process.	CTLSA	

JOBCARD	Job card data to be used for all jobs submitted during the Connector for RACF installation procedure. Maximum length is 43 characters. The value must not contain blanks and must be enclosed in apostrophes.
	Example of the use of JOBNAME and JOBCARD parameters:
	- `JOBNAME=CTSINS`
	- `JOBCARD=‘,CTSINST,CLASS=A,MSGCLASS=X’`
	Resulting job card:
	`//CTSINS01`
	`JOB,CTSINST,CLASS=A,MSGCLASS=X`
		SA,CLASS=A,MSGCLASS=X	

STEPLIB	The Connector for RACF Load Module library. Note that	CTLSA.V400.LOAD	

Parameter	Description	Default Value	Value
	the last qualifier of this library name must contain a maximum of four characters.		
ILPREFS	High level dataset name qualifier (prefix) of the Connector for RACF installation libraries.	CTLSA	
ILVERS	Second level dataset name qualifier (version) of the Connector for RACF installation libraries.	V400	
ILUNITS	Name of DASD unit where Connector for RACF installation libraries will be placed. • For non-SMS managed datasets, specify a generic unit (for example, 3390). • For SMS-managed datasets, it is recommended to specify a null value (that is, specify **ILUNITS=,**).	@@@@	
ILVOLS	• For non-SMS managed datasets specify the volume serial number on which Connector for RACF installation libraries will be placed. • For SMS-managed datasets, if a null value was specified in ILUNITS, specify a null value for this parameter as well (that is, specify **ILVOLS=,**).	####	
OLPREFS	High level dataset name qualifier (prefix) of the Connector for RACF operation datasets.	CTLSA	
OLVERS	Second level dataset name qualifier of the Connector for RACF operation datasets.	V400	
OLUNITS	Name of DASD unit where Connector for RACF operation datasets will be placed. • For non-SMS managed datasets, specify a generic unit (for example, 3390). • For SMS-managed datasets, it is recommended to specify a null value (that is, specify **OLUNITS=,**).	@@@@	
OLVOLS	• For non-SMS managed datasets specify the volume serial number on which Connector for RACF operation datasets will be placed.	####	

Parameter	Description	Default Value	Value
	- For SMS-managed datasets, if a null value was specified in OLUNITS, specify a null value for this parameter as well (that is, specify **OLVOLS=,**).		
SPCPREF	High level dataset name qualifier (prefix) of the Connector for RACF SMP/E CSI dataset.	CTLSA	
SPCVER	Second level dataset name qualifier of the Connector for RACF SMP/E CSI dataset.	V400	
SPCVOL	- For non-SMS managed datasets specify the volume serial number on which Connector for RACF SMP/E CSI dataset will be placed. - For SMS-managed datasets, you may specify an asterisk (that is, specify **SPCVOL=*,**).	####	
SPAPREF	High level dataset name qualifier (prefix) of the Connector for RACF SMP/E datasets.	CTLSA	
SPAVER	Second level dataset name qualifier of the Connector for RACF SMP/E datasets.	V400	
SPAUNIT	Name of DASD unit where Connector for RACF SMP datasets will be placed, - For non-SMS managed datasets, specify a generic unit (for example, 3390). - For SMS-managed datasets, it is recommended to specify a null value (that is, specify **SPAUNIT=,**).	@@@@	
SPAVOL	- For non-SMS managed datasets specify the volume serial number on which Connector for RACF SMP/E datasets will be placed. - For SMS-managed datasets, if a null value was specified in SPAUNIT, specify a null value for this parameter as well (that is, specify **SPAVOL=,**).	####	
SPDPREF	High level dataset name qualifier (prefix) of the Connector for RACF SMP/E distribution libraries.	CTLSA	
SPDVER	Second level dataset name qualifier of the Connector for RACF SMP/E distribution libraries.	V400	

Parameter	Description	Default Value	Value
SPDUNIT	Name of DASD unit where Connector for RACF SMP/E distribution libraries will be placed. • For non-SMS managed datasets, specify a generic unit (for example, 3390). • For SMS-managed datasets, it is recommended to specify a null value (that is, specify **SPDUNIT=,**).	@@@@	
SPDVOL	• For non-SMS managed datasets specify the volume serial number on which Connector for RACF SMP/E distribution libraries will be placed. • For SMS-managed datasets, if a null value was specified in SPDUNIT, specify a null value for this parameter as well (that is, specify **SPDVOL=,**).	####	

DEFPARMS Parameters

Parameters	Description	Default value	Value
%HOLDCLASS%	Single-character Held output class for RACF Connector procedures and jobs.	X	
%DUMPCLASS%	Single-character Held output class for dumps, diagnostic messages and snaps. This type of output should go to a held output-class which is cleaned frequently (if not printed) in order not to cause spool fill-out.	X	
%WORKUNIT%	Unit name for temporary datasets (for example, SYSDA, SORTWORK).	SYSALLDA (exists in all z/OS systems)	
%BRODCAST%	Name of the system BRODCAST dataset. This dataset is required in the Connector for RACF started task for RACF API execution	SYS1.BRODCAST	
%LPALIB%	Name of the library to which modules that must reside in LPA would be copied. This dataset is optional for Installation of Online Interceptor exits for RACF.	SYS1.LPALIB	

Parameters	Description	Default value	Value
%RSSGTYPE%	Type of security product installed. Default value must not be changed.	RACF	
%RSSNAME%	Managed System name. The name must correspond to the Managed system name defined in SailPoint. It is recommended to use a name which is up to 8 characters long. If the name is longer than 8, follow the instructions mentioned in Step 11 – Adjust for Longer Managed System Names. For more information, see Communication Parameters Coordination.	MVSRACF	
%PROCPREFS%	First three characters of the Connector for RACF JCL procedure, and optional STCJOB names, after they are copied to the requested library, as specified in the **%PROCLIB%** and **%STCJOBS%** parameters. This determines the name of started tasks used by Connector for RACF. When LOCALCOPY is set for **%PROCLIB%**, the value for this parameter must not be CTS, ACF, TSS or RCF. **Note** When %PROCLIB% is not DONTCOPY or LOCALCOPY, ensure that there are no procedures in the PROCLIB concatenation which start with the value set for this parameter. If STCJOBs are used, verify the same for the system STCJOBs concatenation.	CTS	

Parameters	Description	Default value	Value
	Refer the following tables for the inter-relations between **%PROCPREFS%**, **%PROCLIB%** and **%STCJOBS%** parameters: • Allowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters • Disallowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters		
%PROCLIB%	Name of the System JCL Procedure library to which the Connector for RACF procedures will be copied. The copy will be done while renaming the procedures using the value specified for **%PROCPREFS%** as the first 3 characters. Other possible values: • DONTCOPY – This reserved value can be specified to prevent the Connector for RACF procedures from being copied to any procedures library. **Note** DONTCOPY in **%PROCLIB%** forces DONTCOPY in **%STCJOBS%**. • LOCALCOPY - This reserved value can be specified to copy the Connector for RACF procedures to the RACF Connector procedures library while renaming them using the **%PROCPREFS%** value as the first 3 characters. LOCALCOPY can be used when **%STCJOBS%** value is not DONTCOPY.	SYS2.PROCLIB	

Parameters	Description	Default value	Value
	The procedures will be used by the STCJOBs using the JCLLIB statement. Refer the following tables for the inter-relations between **%PROCPREFS%**, **%PROCLIB%** and **%STCJOBS%** parameters: - Allowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters - Disallowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters		
%STCJOBS%	Name of the System library for source JCL for started tasks (STCJOB) to which the Connector for RACF STCJOBs would be copied. This library should be defined in the IEFJOBS DD statement in the MSTJCLxx system PARMLIB member. The reserved value DONTCOPY can be specified to prevent the Connector for RACF STCJOBs from being copied. DONTCOPY must not be used when temporary datasets are protected by RACF. For more information see RACF Considerations and *Using STCJOBs* in Installation Considerations. Refer to the following tables for the inter-relations between **%PROCPREFS%**, **%PROCLIB%** and **%STCJOBS%** parameters: - Allowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters	DONTCOPY	

Parameters	Description	Default value	Value
	- Disallowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters		

Allowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters

The following table describes the allowed combination of values for **%PROCPREFS%**, **%PROCLIB%**, and **%STCJOBS%** parameters within DEFPARMS member.

Allowed combination of values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% parameters

%PROCPREFS%	%PROCLIB%	%STCJOBS%	Results / notes
CTS/xxx	<proclib>	DONTCOPY	Procedures are copied with specified prefix to <proclib> and STCJOBs are not copied.
CTS/xxx	<proclib>	<stcjobs library>	Procedures are copied with specified prefix to <proclib> and STCJOBs are copied with specified prefix to <stcjobs library>.
CTS/xxx	DONTCOPY	Any value	Nothing is copied. If required customer must manually copy the procedures and STCJOBs.
xxx	LOCALCOPY	<stcjobs library>	Procedures are copied with specified prefix to Connector PROCLIB and STCJOBs are copied with specified prefix to <stcjobs library>. The procedures will be used by the STCJOBs using the JCLLIB statement.

Disallowed Combination of Values for %PROCPREFS%, %PROCLIB%, and %STCJOBS% Parameters

The following table describes the combination of values that are not allowed for **%PROCPREFS%**, **%PROCLIB%**, and **%STCJOBS%** parameters within DEFPARMS member.

%PROCPREFS%	%PROCLIB%	%STCJOBS%	Notes
CTS/xxx	LOCALCOPY	DONTCOPY	Prevented with an error message by CHNGEPRS job as the procedures cannot be started.
CTS/RCF/ACF/TSS	LOCALCOPY		Prevented with an error message by CHNGEPRS job as there are already members with these names in Connector PROCLIB.

RSSPARM Parameters

Parameters	Description	Default value	Value
CTSA_ID	Unique 4-character identifier for Connector for RACF to use. If more than one instance of Connector for RACF is installed on the platform, each instance should have a unique ID.	`<xxx>R` where <xxx> is the value specified for **%PROCPREFS%** in DEFPARMS	
RSS_TYPE	Managed System type. Specify: RACF	RACF	RACF
RSS_WORK_DIR	The prefix used to dynamically allocate working datasets	By default, the prefix used consists of the following: `<prefix>.<version>.<RSS_NAME>` where • `<prefix>` is the value specified for OLPREFS • `<version>` is the value specified for OLVERS • `RSS_NAME` is the value specified for **%RSSNAME%** in DEFPARMS	

Communication Parameters (CTSPUSR and ECAPARM Parameter Members)

Refer to the Communication Parameters Coordination table before setting values for the parameters mentioned in the table below.

Parameter	In Member	Description	Default value	Value
PORT	ECAPARM	Lower of the two consecutive port numbers to be used for TCP/IP communication.	2470	
NUMSRV	ECAPARM	The number of Transaction Servers (CSs) defined for Connector for RACF. This number determines the number of SailPoint requests	2	

Parameter	In Member	Description	Default value	Value
		that can be processed concurrently. Each Transaction Server handles a single request at a time. The maximum number of CS's that can be specified is 3. **Note** The number of Notification Servers (CDs) is one. This number cannot be changed.		
IPLIST	ECAPARM	(*Optional*) Used to validate the incoming IP addresses by CTSGATE. For full description and syntax, see Configuring Incoming IP Address Validation.	-	

Step 2 – Prepare Installation IMAGE from the TRS File

Use this procedure to prepare an installation **IMAGE** from the TRS file.

2.1 – Transfer the INSTALL.TRS file using FTP Binary

Using FTP, upload file INSTALL.TRS from the TRS file to the target system, using BINARY format.

- The name of the uploaded dataset must be the name set for **upload dataset name** in Installation Upload and IMAGE Datasets Parameters.
- The uploaded dataset should be pre-allocated with LRECL=1024, BLKSIZE=6144, RECFM=FB
- The dataset size is approximately 45 cylinders on a 3390 device.

At the command prompt, specify the following commands to upload the dataset where `<mvssystem>` is the DNS name of your host system:

```
FTP <mvssystem>
```

Specify the user ID and password or phrase using the following commands:

```
bin
put <drive>:\RACF\Install\install.trs
'<upload_dataset_name>'
```

2.2 – UNCOMPRESS the TRS file

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

1. Tailor the following TRSMAIN job. Replace upload_dataset_name and **%xmitlib%** with the values set in Installation Upload and IMAGE Datasets Parameters.

```
//UNTERSE  JOB ,'UNTERSE',CLASS=A,MSGCLASS=X
//*
//UNTERSE  EXEC PGM=TRSMAIN,PARM=UNPACK
//SYSPRINT DD   SYSOUT=*
//INFILE   DD   DISP=SHR,DSN=<upload_dataset_name>     <== Customize
//OUTFILE  DD   DISP=(NEW,CATLG),UNIT=SYSALLDA,
//       DSN=%xmitlib%,                                <== Customize
//       SPACE=(CYL,(120,10,10),RLSE)
```

2. Run the job.

   The job step should end with a condition code of `0`.

2.3 – Tailor the $RECEIVE Job

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

1. Edit member $RECEIVE in library **%xmitlib%**. The job performs RECEIVE operations to convert the uncompressed files from XMIT format to the installation **IMAGE** format.

2. Replace **%xmitlib%**, **%instpref%**, **%UNIT%**, and **%VOLSER%** with the values set in Installation Upload and IMAGE Datasets Parameters.

   > **Note**
   > Verify that your SMS does not impose attributes on the installation files by SMS DATACLAS or by pre-allocation.

2.4 – RECEIVE the Installation IMAGE

1. Run the $RECEIVE job.

   All job steps should complete with a condition code of `0`.

2. Check the job output and verify that all RECEIVE and COPY instructions ended successfully and all members were received and copied.

Step 3 – Allocate and Load Connector INSTALL Library

Create the INSTALL library and load it from the **IMAGE** INSTALL library.

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

3.1 – Copy, Edit and Run Member $LOADINS

Copy member $LOADINS from library **%xmitlib%** to a new member in the **%xmitlib%** library. The job creates the INSTALL library and loads it from the **IMAGE** INSTALL library.

1. Edit the member.

2. Replace **%instpref%** with the value set in Installation Upload and IMAGE Datasets Parameters.

3. Set the values for **ILPREFS**, **ILVERS**, **ILUNITS** and **ILVOLS** procedure parameters with the values set for these parameters in the Connector for RACF Datasets Allocation Parameters table.

4. Run the job.

   The job step should end with a condition code of `0`.

5. Check the job output and verify that the INSTALL library members were copied successfully.

Step 4 – Allocate and Load Connector for RACF Installation Libraries

Use the values assigned in the Connector for RACF Datasets Allocation Parameters table for setting the required values in this step.

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

4.1 – Tailor the LOADCTS Member

1. Edit LOADCTS member in INSTALL library.

   This member contains JCL for the following:

   - Allocating Connector for RACF libraries.
   - Loading Connector for RACF libraries.
   - Performing JCL adaptations of a few installation jobs.

2. Tailor the job card.

3. Specify values for all the procedure parameters, using the values set in the Connector for RACF Datasets Allocation Parameters table.

4.2 – Submit Job to Allocate and Load Connector for RACF Libraries

1. Check the JCL in member LOADCTS.

2. Submit the job.

   All the job steps must end with a condition code of `0`.

The following error message generated by this job is not an error and can be disregarded:

```
"CTS914E - Modifying of cards ended"
```

Datasets which are allocated by the job are listed in Appendix C: Connector for RACF Datasets and JCL Procedures.

Step 5 – Tailor Connector for RACF Members with Site Parameters

This step involves assigning installation parameters and then adjusting RACF members to suit your organizational needs.

5.1 – Assign Installation Parameters

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

1. Edit member DEFPARMS in the Connector INSTALL library.

   > **Note**
   > The member contains keywords which must be assigned the required installation values.

> (Keywords are parameters which start and end with percent signs; for example, **%HOLDCLASS%**.)

2. Assign values using the values set in DEFPARMS Parameters.

3. Save the member (if it was modified).

> **Note**
> Throughout this book, the samples are based on the assumption that you have defined your Started Task procedures using the default **CTS** prefix. If you are using a prefix other than **CTS**, adapt the started task name in the samples to match the prefix specified in the parameter %PROCPREFS%.

5.2 – Modify Connector for RACF Members

Member CHNGEPRS in the Connector INSTALL library contains JCL cards to modify Connector for RACF members so that they conform to the installation naming conventions.

The JCL of this job should have already been set for submission by an earlier installation procedure.

> **Note**
> The job modifies members in the Connector for RACF libraries according to values set in member DEFPARMS in 5.1 – Assign Installation Parameters and values set in the previous installation procedures. Any errors in the parameter definitions in DEFPARMS may result in a need for restarting the installation procedure from the beginning. Therefore, prior to submitting the job, verify that the parameters defined in member DEFPARMS are correct.

1. Check the JCL.

2. Submit the job.

   The job updates members in various Connector for RACF libraries.

3. Save the member (if it was modified).

4. Scan the output of the job for information and error messages issued by the job.

   The job step must end with a condition code of `0`.

Step 6 – Copy Connector for RACF Procedures, STCJOBs, and Other Members

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

This step copies members to the appropriate libraries:

- Procedures are copied to the requested procedures library (when **%PROCLIB%** is not DONTCOPY)
- When **%STCJOBS%** is not DONTCOPY:
  - STCJOBs are copied to the system STCJOBs library.
  - INCLUDE members are copied to the requested procedures library.
- Parameter members are copied to the Connector PARM library.

6.1 – Submit the CPYMRACF Job to Perform the Copy

Member CPYMRACF in the Connector INSTALL library copies the required members to the appropriate libraries. When copied, the procedures and, if required, the STCJOBs, are renamed using the value assigned to **%PROCPREFS%** as first 3 characters. The copy is done without replace, therefore, existing procedures or STCJOBs with the same names will not be overwritten.

> **Note**
> If you specified the value **DONTCOPY** in **%PROCLIB%** parameter (in DEFPARMS member), job CPYMRACF does not copy the procedures and STCJOBs, if requested, to your system PROCLIB and STCJOBS libraries. Instead you must copy them manually from the Connector PROCLIB library and set the first three characters to match the value specified in **%PROCPREFS%** parameter in DEFPARMS member.

1. When ready, submit the job.
2. Check the whole sysout. In each copy step, check IEBCOPY utility messages and verify that all members were copied successfully.

   Expected condition codes:

   - When **%PROCLIB%** is DONTCOPY, only steps CHECCPR and COPYPARM are executed and both must end with a condition code of `0`.
   - When **%PROCLIB%** is not DONTCOPY:

- When **%STCJOBS%** is DONTCOPY, steps CHECCPR, COPYPROC and COPYPARM are executed. Step CHECCPR must end with a condition code of 4. The other steps must end with a condition code of `0`.
- When **%STCJOBS%** is not DONTCOPY, steps CHECCPR, COPYPROC, COPYSTCJ, COPYSTCI and COPYPARM are executed. Step CHECCPR must end with a condition code of 4. All other steps must end with a condition code of `0`.

See Appendix C: Connector for RACF Datasets and JCL Procedures for a list of the JCL procedures and STCJOBs that are copied by this job.

> **Note**
>
> If JES3 is active in your environment, update all SYSOUT DD cards in all the procedures, and optionally STCJOBs, copied by this job. The update is to drop the whole DCB parameter from ALL SYSOUT DD cards in ALL STCs.
>
> For example, instead of:
>
> ```
> //STDMSG DD SYSOUT=&OUT,DCB=(RECFM=FA,LRECL=133,BUFNO=1)
> ```
>
> You should now have:
>
> ```
> //STDMSG DD SYSOUT=&OUT
> ```

Step 7 – Customize Connector for RACF Installation Parameters

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

7.1 – Create CTSPARM Module

Member CTSPARMJ in the Connector INSTALL library should already be set for submission by earlier installation procedures.

1. Check the JCL.
2. Submit the job.

   The job creates load module CTSPARM in the Connector LOAD library.

   All job steps must end with a condition code of `0`.

> **Caution**

> If the job returns with a condition code of `12`, you may need to make the following change in the `DCB` of the `SYSPUNCH DD` statement. The issue may have been caused by a change in the Binder which occurred during an update. This file is used as the input for the Binder. Under certain conditions, this may fail. This process converts the file to a sequential file, which is handled differently, and it allows the job to process correctly under the conditions which caused the `12` return code.
>
> 1. Locate the following script:
>
> ```
> //SYSPUNCH DD DSN=&&OBJECT,UNIT=&UNIT,SPACE=(80,(200,50)),
> // DCB=(DSORG=PO,RECFM=FB,LRECL=80,BLKSIZE=800),
> // DISP=(,PASS)
> ```
>
> 2. Update it to the following:
>
> ```
> //SYSPUNCH DD  DSN=&&OBJECT,UNIT=&UNIT,SPACE=(80,(200,50)),
> //DCB=(RECFM=FB,LRECL=80,BLKSIZE=800),
> //DISP=(,PASS)
> ```

7.2 – Assign RSSPARM Parameter Values

1. Edit member RSSPARM in the Connector PARM library.

2. Assign a value to the parameters using the values set in the RSSPARM Parameters table.

Step 8 – Format Connector for RACF Datasets

> **Note**
> Wait for the previous installation job to finish before continuing with this procedure.

8.1 – Edit and Run Member FORMCTS

Member FORMCTS in the Connector INSTALL library contains a job which allocates and formats the Connector for RACF operation datasets described in the Connector for RACF Operation Datasets table below. The JCL should already be set for submission by earlier installation procedures.

Consider the following parameters in the Connector for RACF Operation Datasets table:

- `<prefix>` - Value set for the OLPREFS parameter in the Connector for RACF Datasets Allocation Parameters table.

- `<version>` - Value set for the OLVERS parameter in the Connector for RACF Datasets Allocation Parameters table.

Connector for RACF Operation Datasets

`<prefix>.<version>.QUEUE`
`<prefix>.<version>.DIAGLVL`
`<prefix>.<version>.RSSOFLI`
`<prefix>.<version>.ENCRINT`
`<prefix>.<version>.ENCREXT`
`<prefix>.<version>.RSSKWDS`
`<prefix>.<version>.USER.CLIST`
	`<prefix>.<version>.CAREGRP`
	`<prefix>.<version>.CARECNN`
	`<prefix>.<version>.CAREOE`
	`<prefix>.<version>.CAREUSR`
	`<prefix>.<version>.AITOUT`
	`<prefix>.<version>.AITLIST`
	`<prefix>.<version>.AITDONE`
	`<prefix>.<version>.AITMSG`
	`<prefix>.<version>.AITIN`
	`<prefix>.<version>.RCFDELRQ`

1. Check the JCL.

2. Submit the job.

   All job steps must end with a condition code of `0` except:

   - CHECACF2 – This step ends with a condition code of 4.
   - CHECKTSS – This step ends with a condition code of 4.
   - ALLOCACF – This step is not executed.
   - INITACF – This step is not executed.
   - ALLOCTSS – This step is not executed.

> **Note**
> Message IEC031I, indicating system D37 failure, appears during execution of the JCL step FORMQUE. This message can be ignored since it is expected behavior and does not present a problem.

Step 9 – Customize Communication Settings

Communication customization should be performed at this time to enable the Connector for RACF to communicate with SailPoint through the Connector Gateway (CG).

9.1 – Verify TCP/IP Connectivity

The Connector for RACF communicates with the Connector Gateway using the TCP/IP protocol.

A functional TCP/IP connection between Connector for RACF and the Connector Gateway is required. Any network topology configuration that supports TCP/IP (hardware and software) can be used, as long as TCP/IP connections can be established between Connector for RACF and the Connector Gateway. Connectivity should be verified before you start the Connector Gateway (for example, use the ping command, Telnet commands or other TCP/IP applications).

9.2 – Specify Connector for RACF Gateway Communication Parameters

Member **ECAPARM** in the Connector PARM library is used to define Connector for RACF Gateway communication parameters.

Edit member ECAPARM in the Connector PARM library and set up the parameters using the values set in the Communication Parameters (CTSPUSR and ECAPARM Parameter Members) table.

9.3 – Define the TCP/IP DATA File

> **Important**
> You might not need to update TCP/IP data file to enable CTSGATE to communicate with SailPoint. You can skip this step now and return to it only if you encounter an issue establishing a communication between CTSGATE and SailPoint.

z/OS TCP/IP regards the CTSGATE started task as a client application requiring a client profile dataset. This profile dataset is referred to in MVS documentation as **hlq.TCPIP.DATA** (**hlq** is the high-level qualifier for the dataset). This dataset is the main resolver configuration dataset as set up in the local TCP by the MVS/TCP systems programmer.

The TCP/IP profile dataset contains information such as the host name, domain origin and the TCPIPJOBNAME parameter. This information identifies the TCP/IP stack to use. (For more information regarding this dataset, see the IBM document *z/OS Communications Server IP Configuration Guide*.

When attempting to locate the TCP/IP profile dataset, MVS searches using the following sequence of names:

- `<jobname>.TCPIP.DATA` (for batch jobs and started tasks)
- `SYS1.TCPPARMS(TCPDATA)`

- `TCPIP.TCPIP.DATA`

Once located, the dataset is dynamically allocated. The default value assigned for the high-level qualifier for the TCP/IP profile dataset during TCP/IP setup is **TCPIP**.

If the high-level qualifier for this dataset in your system has been assigned a different value or if this dataset has not been assigned one of the standard names listed above, the dataset name must be specified in parameter TCPDATA in the CTSGATE started task. This parameter is referred to by the //SYSTCPD DD statement.

This issue should be coordinated with the MVS/TCP systems programmer in your organization.

> **Note**
> If the high-level qualifier for TCPIP.DATA at your site is TCP01, Modify the TCPDATA parameter in the Connector for RACF Gateway JCL procedure (CTSGATE):
>
> ```
> // TCPDATA=TCP01.TCPIP.DATA,
> ```
>
> If the high-level qualifier of this dataset is TCPIP (the default), this parameter must be left with its default value (NULLFILE).

9.4 – Set Up Secured Communication

Secured communication can be implemented using TLS secured communication or Transmitted Data Encryption.

For more information, see Communication Parameters Coordination for descriptions of each option before selecting the secured communication method.

1. Install the selected secured communication method using the steps described in Secured Communication.

Step 10 – Define Connector for RACF in RACF

> **Note**
>
> Sample RACF definitions for the following can be found in member CTSRACF in the INSTALL library. Read the notes in member CTSRACF carefully and tailor according to site standards before submitting the job or using the commands from this member. After submitting the job or executing the commands, check the whole output and verify that all the commands were processed successfully.
>
> It is assumed that the user who installed Connector for RACF has full authority to all Connector for RACF files, assigned to him at the beginning of the installation process.

10.1 – Define Connector for RACF Started Tasks in RACF

Started task	Description
CTSGATE	Connector for RACF Gateway
CTSACS	Connector Transaction Server (CS)
CTSACD	Connector Notification Server (CD)
CTSAONI	Connector Online Interceptor
CTSAOFI	Connector Offline Interceptor

> **Note**
> In the list of started tasks used in this section, it is assumed that the default value CTS was accepted for the DEFPARMS parameter, PROCPREFS. If you assigned a different value to this parameter, modify the started task names accordingly.

In RACF Connector, Get operations are performed under the STC user. Since RACF Connector is APF authorized, no other permissions are required.

There is an exception however, the Target Aggregation operation requires AUDITOR or SPECIAL privilege in RACF. Therefore, the Connector Transaction Server (CS) and the Connector Notification Server (CD) must be defined with the AUDITOR or SPECIAL attribute.

10.2 – Set Permissions to Connector Datasets

Permit READ access for the Connector DIAGLVL and CLIST libraries for your MVS system programmers, z/OS staff, or SailPoint Mainframe support team who should be able to see them.

> **Important**
> Do not allow users access to any DIAGLVL or CLIST libraries in the CTSRACF installation job.

Permit all Connector for RACF installation libraries to be accessed by Connector for RACF started tasks listed above with read and write authorizations.

10.3 – Protect the Encryption Keys Datasets

Protect transmitted and stored data encryption key data.

Transmitted Data Encryption Keys Dataset

> **Note**
> This permission is only required when Transmitted Data Encryption is implemented.

Set RACF to permit only Connector for RACF servers (CTSACS and CTSACD) READ access to the encryption key dataset ENCREXT created in 9.4 – Set Up Secured Communication. No other accounts, other than the installer User ID, must be authorized to access this dataset (not even READ authorization).

Stored Data Encryption Keys Dataset

Set RACF to permit only Connector for RACF servers (CTSACS and CTSACD) and Connector Interceptors (CTSAONI and CTSAOFI) READ access to the encryption key dataset ENCRINT created in Step 8 – Format Connector for RACF Datasets. No other accounts, other than the installer User ID must be authorized to access this dataset (not even READ authorization).

10.4 – Define an OMVS Segment

Define an OMVS segment for the user ID and group ID associated with the CTSGATE started task. For more information, see details provided within the CTSRACF member.

10.5 – Grant CTSGATE with Authority to Use TCP/IP Stack

This permission is required only when RACF SERVAUTH resource class is defined to protect TCP/IP resources from unauthorized access. For more information, see details provided within the CTSRACF member.

Step 11 – Adjust for Longer Managed System Names

The name assigned to the Managed System may be up to 32 characters long. However, if the Managed System name length is longer than 8 characters, certain adjustments must be performed.

If the length of the Managed System name is greater than 8 characters, do the following after completing the installation of Connector for RACF:

1. Select a short name (up to 8 characters) for the Managed System name (referred to as the *short MS name*). This name must be unique for each Managed System managed by the instance of Connector for RACF.

2. Edit the RSSPARM member in the Connector PARM library. The value for parameter RSS_WORK_DIR contains the prefix for dynamically allocated datasets. The prefix contains the Managed System name as a qualifier. Change the Managed System name to the short Managed System name so that the value conforms to MVS dataset naming conventions.

3. Edit the member CTSOFLI in Connector JCL library. The member contains DELETE and LISTCAT commands regarding Offline Interceptor datasets. The dataset names contain the Managed System name as a qualifier. Change the Managed System name to the short Managed System name so that the dataset names conform to MVS dataset naming conventions.

4. Edit members CTSOFLMI and CTSOFLMR in Connector for RACF JCL library. Add the parameter RSSQ= with the value used as the short Managed System name to the EXEC statement in both members. Save the members.

> **Note**
> The short RSSNAME qualifier must be identical in all modifications.

Step 12 – Adjusting Managed System Administrator Attributes

12.1 - Provide Managed System Administrator Passwords

This section is relevant when you define a new Application or when you set a new Managed System Administrator in SailPoint.

The RACF Connector does not save the Managed System Administrator passwords in a file, as most of the other Connectors do.

When a new Managed System Administrator is defined in **Application** definition in IdentityIQ or in **Source** definition in IdentityNow, the Managed System Administrator password or phrase is required for verification, unless a protected user is defined as the Managed System Administrator. The Managed System Administrator User ID and password or phrase are sent from SailPoint to the RACF Connector where the password or phrase is verified. After verification, the password or phrase is not saved anywhere on the Connector's platform or in SailPoint.

Before setting the managed System Administrator user and password or phrase in SailPoint, ensure that the password or phrase of the user is not expired (a new password or phrase set using the INSERT or CHANGE command is usually expired automatically, and must be changed when logging on for the first time). If the password or phrase is expired, the password or phrase verification done by RACF Connector will fail.

12.2 - Verify Managed System Administrator Permissions

Set operations, activated from SailPoint, are done in RACF under the Managed System Administrator User ID service account. Therefore, this account requires SECURITY privileges in RACF. To limit the scope of the service account, the group-SPECIAL attribute may be set to the user’s connection to the group as needed.

Step 13 – Add Connector for RACF Libraries to the MVS Authorized Libraries List

Add the Connector LOAD library and Connector CTRANS library to the MVS APF authorized libraries list.

1. Edit member PROG*nn* in the SYS1.PARMLIB library. Add the Connector LOAD and CTRANS libraries and their volumes to the list using the APF statement.

2. Add the libraries to the active APF list by specifying the following operator command. Replace *nn* with the suffix of the PROG*nn* member that is updated:

```
SET PROG=nn
```

Step 14 – Review the Installation

You have completed the Connector for RACF installation procedure. Before starting Connector for RACF, it is recommended that you refer to the Installation checklist and review the installation procedures to make sure nothing was omitted.

14.1 – Verify the Connector for RACF Gateway is Installed Properly

Verify that the Connector for RACF Gateway is installed properly by issuing the following operator command:

```
S CTSGATE
```

The Connector for RACF Gateway starts, and then automatically starts the Connector servers: Transaction Server (CS) and Notification Server (CD). The Connector for RACF Gateway waits for communication to be established with SailPoint.

> **Note**
> For more information about starting and stopping Connector processes, see Operations.

14.2 – Verify that the RACF Connector Communicates with SailPoint

Perform the required configuration activities in Connector Gateway and SailPoint.

For more information, see Communication Parameters Coordination and the appropriate Connector Gateway and SailPoint guides for detailed configuration description.

Verify that all components communicate successfully.

> **Note**
> For IdentityIQ, perform the **Test Connection** to synchronize the keywords file, perform account/group aggregation, and so on.

14.3 – Verify RACF Connector RACF Interface

To test Connector for RACF without involving SailPoint, see Appendix F: Connector for RACF Batch Utility.

It is not recommended to perform LISTUSER to all accounts with all attributes as this produces a very large sysout file.

Step 15 – (*Optional*) Local Changes Migration

Local changes made by customer in previous version of RACF Connector environment may be copied manually to new RACF Connector environment.

Such local changes may refer to:

- Parameters set in RSSPARM member in PARM library
- RSSAPI member in PARM library
- Scripts in **USER.CLIST** library

> **Note**
> If there are parameters set in CTSPARM member in PARM library of the previous version and there is a need to migrate them to the current version, contact SailPoint Customer Support.

Step 16 – Configure Automated Startup of Connector for RACF

If an automatic startup tool is used to start all the required applications after system initialization, add `CTSGATE` to this automatic startup tool definitions. Otherwise, add the following command to `member COMMND`*`nn`*in `SYS1.PARMLIB` to start Connector for RACF:

```
S CTSGATE
```

> **Important**
> If the Connector for RACF starts automatically during system initialization, it must be started after TCP/IP and the RACF subsystem complete their initialization.

Step 17 – Customizing RACF Support

Customization of RACF support must be performed at this time.

For information and step by-step instructions, see RACF Support Customization. Perform this customization before continuing to the next procedure in the installation.

Step 18 – Post Installation Checks

1. Start the Mainframe Connector.
2. Start the Connector Gateway and ensure that in the Mainframe **CTSGATE**, a connection is established with the Connector Gateway.
3. To synchronize keywords after Mainframe Connector installation, perform the following:

- Check application schema of configured RACF application and perform **Test Connection**.
- For any existing application:
  - Check application schema of configured RACF application.
  - To use a new schema attribute, add it in to account or group schema attribute accordingly.

> **Note**
> If you install a new Mainframe Connector while retaining the same IdentityIQ application, then there may be an issue of keywords not being synchronized. In this case, update the description of any schema attribute for any minor changes.

4. Save the application.
5. Performing any operation would first perform the Test Connection and Keywords synchronization as the first operation and it would update keywords on Mainframe Connector in **RSSKWDS** file.

Uninstalling Connector

To uninstall the Connector for RACF, perform all of the following procedures.

1. Delete all Connector datasets.
2. Delete all Connector procedures from the system PROCLIB.
3. Remove references to Connector LOAD and CTRANS libraries from PROGxx member in SYS1.PARMLIB.
4. If STCJOBs are installed, delete them from the system STCJOBs library.
5. If you configured the connector for automatic start, remove the relevant definition.
6. Delete the following RACF definitions:
   - Security definition for Connector started tasks
   - Connector dataset permissions
   - RACF accounts defined for Connector

RACF Support Customization

> **Note**
> It is highly recommended that you review all of the topics discussed in this chapter to determine which are relevant for your implementation of Connector for RACF.

The following topics are discussed in this chapter:

Overview of Connector for RACF Interceptors

This chapter provides background material relevant to several of the customization procedure described in this chapter.

The Connector for RACF detects changes and events in RACF via the following components:

- Online Interceptor (CTSAONI)
- Offline Interceptor (CTSAOFI)

These Connector for RACF components, described below, process data in cooperation with RACF and various components within your z/OS system. This chapter describes the required customization of these components.

> **Note**
> Online Interceptor and Offline Interceptor are not supported for IdentityNow

Online Interceptor Logic

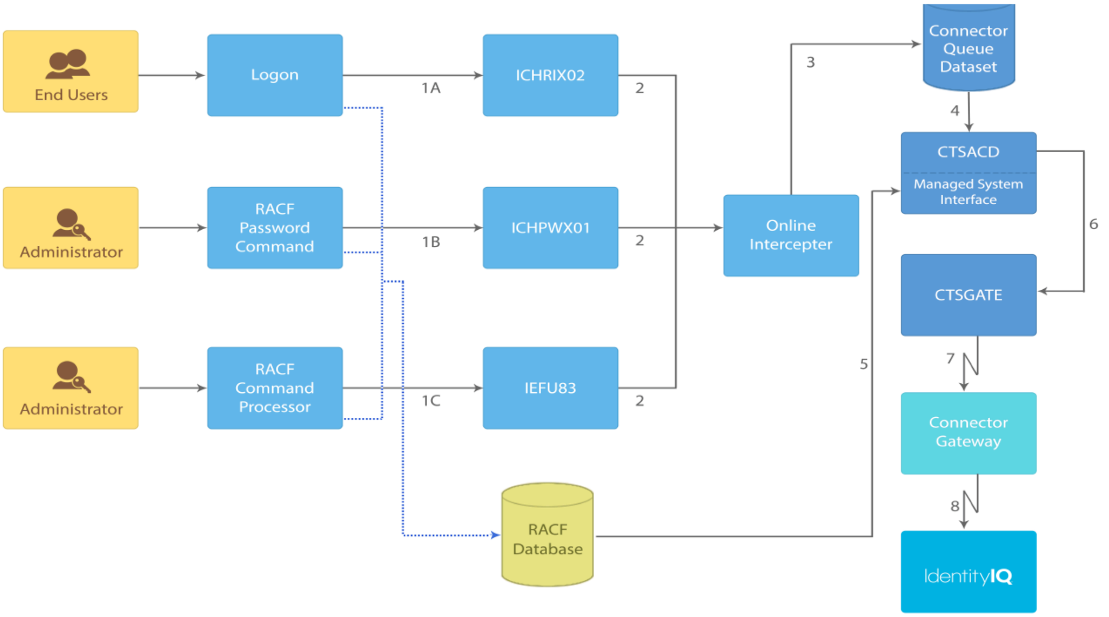
The Connector Online Interceptor detects, in real-time, RACF administration events that occur on the platform, and records them so that they can be reported to IdentityIQ. To accomplish this, the SMF record exit (IEFU83) is used to intercept every RACF command issued in the system, and transfer information regarding the commands to the Online Interceptor.

If password synchronization support is required, the Online Interceptor notifies IdentityIQ of password change events made by RACF accounts. To intercept password change events, RACF postinit exit ICHRIX02 and RACF new password exit ICHPWX01 are used. Both exits must be installed to intercept all password change events in the system.

When a system exit intercepts an event, it notifies the Online Interceptor started task via cross-memory services that the event has been intercepted. The Online Interceptor records the event in Connector for RACF datasets. The data are then reported to IdentityIQ by the Connector Notification Server (CD), via CTSGATE.

As long as the Online Interceptor is active in the system, RACF events and changes are recorded, even if the Connector for RACF is inactive. When the Connector for RACF is restarted, the recorded data are transmitted to IdentityIQ.

The processing flow of the Connector Online Interceptor is illustrated in the following flowchart.



The Connector Online Interceptor detects RACF events in one of the following manners:

- When a RACF user changes his/her password during the logon process [1A], RACF calls exit ICHRIX02.
- When RACF administrators issue an ALTUSER password command [1B], RACF calls exit ICHPWX01.
- When RACF administrators issue a RACF command [1C], RACF calls SMF exit IEFU83 to log the command.

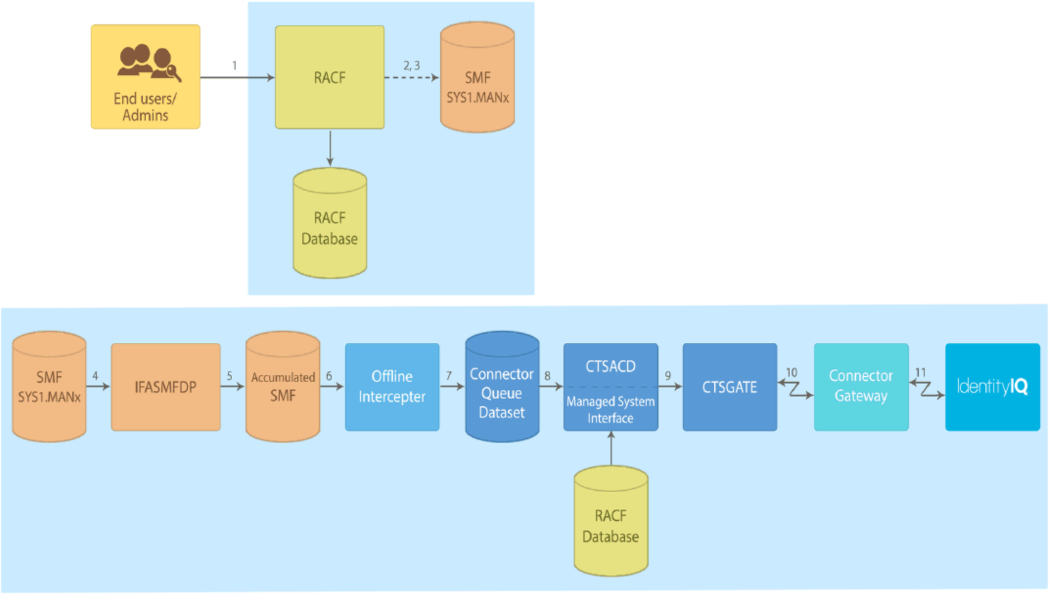
In any of these situations, the exit that intercepts the event passes the event to the Connector Online Interceptor via cross-memory services [2]. The Online Interceptor then writes the event to the Connector QUEUE dataset [3]. The Connector Notification Server (CD) reads the QUEUE dataset [4], gets the updated entity from RACF database, when needed [5] and transfers the event to the Connector for RACF [6] which transfers the event to the Connector Gateway [7] which passes it to IdentityIQ [8].

Offline Interceptor Logic

The Offline Interceptor is intended to be used if the Online Interceptor is not installed or following a period of Online Interceptor inactivity.

While the Online Interceptor intercepts RACF administration events as they occur, the Offline Interceptor reads an SMF dataset in which RACF has logged events that have already occurred.

The processing flow of the Connector Offline Interceptor is illustrated in the following flowchart.



1. When a user issues a RACF command [1], RACF calls SMF [2].
2. SMF logs events to SMF dataset SYS1.MANx [3].
3. Utility IFASMFDP reads dataset SYS1.MANx [4] and writes the event record to a accumulated SMF dataset [5].
4. The Offline Interceptor then reads the accumulated SMF dataset [6] and writes the events to the Connector QUEUE dataset [7].
5. The Connector Notification Server (CD) reads the QUEUE dataset [8] and transfers the event data to the Connector for RACF Gateway [9].
6. This then transfers the event data to Connector Gateway [10] which passes it to IdentityIQ [11].

> **Note**
> Password change events are not handled by the Offline Interceptor. If needed, the Online Interceptor must be used.

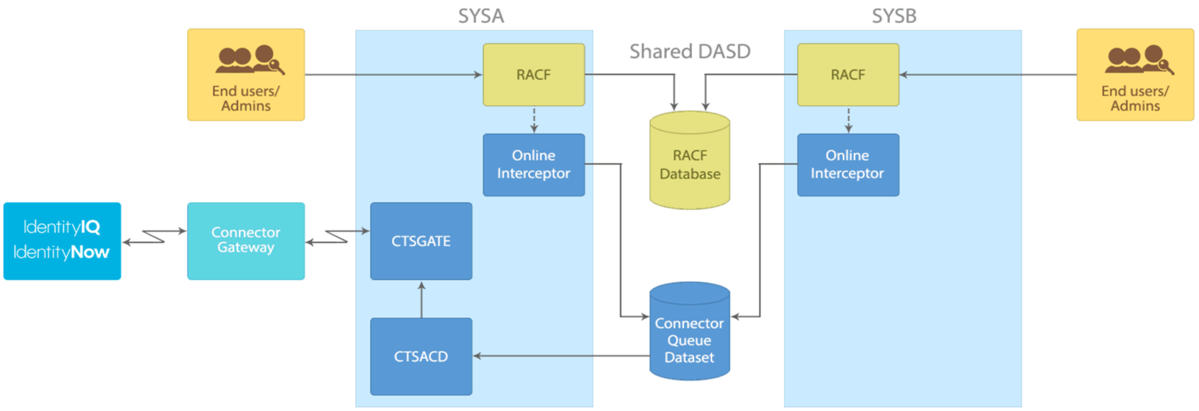
Interceptors in a Shared RACF Database Environment

A shared RACF database environment consists of two or more RACF systems which share the same RACF database.

In a shared database configuration, one of the systems runs a full instance of Connector for RACF. This system is referred to as the primary system. Each additional RACF system sharing the database is referred to as a *secondary system*. Each secondary system only runs one component of Connector for RACF the Interceptor.

The primary system communicates with IdentityIQ and executes the transactions generated by IdentityIQ (in the regular manner). The secondary systems runs the Connector Interceptors to ensure that any updates made to the RACF database from these systems are propagated to IdentityIQ. In this way, the IdentityIQ and RACF databases are kept synchronized.

For example, assume that systems SYSA and SYSB share the RACF database. SYSA is the primary system and runs the complete Connector for RACF. SYSB is the secondary system which runs only the Online Interceptor. This configuration is shown in the following flowchart.



When a transaction is issued by IdentityIQ, it is executed by the Connector Transaction Server (CS) running on the primary system, and the appropriate updates are made to the RACF database.

When a local update is done in either SYSA or SYSB, it is intercepted by the appropriate Interceptor running in the system and written to the Connector QUEUE dataset.

The Notification Server (CD) running in the primary system reads the local update from the Connector QUEUE dataset and forwards it to Connector for RACF Gateway which passes it to IdentityIQ. This ensures that the IdentityIQ and RACF databases are kept synchronized.

Set Logging Options

The Connector Interceptors detect changes made to security definitions and record these changes so that they can be reported to IdentityIQ. To detect these changes, RACF should be set to log all RACF commands that are issued in the system, and SMF should be set to collect these records.

For more information regarding Connector Interceptors, see Overview of Connector for RACF Interceptors.

1 – Set RACF Global Options

RACF global options SAUDIT and AUDIT control RACF command logging as follows:

Global Options	Description
SAUDIT	Instructs RACF to log commands that are issued by users with the SPECIAL or group-SPECIAL attribute
Audit (User Group)	Instructs RACF to log commands, which modify user and group profiles, that are issued by users without the SPECIAL or group-SPECIAL attribute

Set RACF to log these events by issuing the following RACF command:

```
SETROPTS SAUDIT AUDIT (USER GROUP)
```

> **Note**
> Your RACF user ID must be defined with the Auditor and SPECIAL attributes to be able to issue this command.

2 – Set SMF Parameters

RACF generates audit information regarding RACF commands in SMF record type 80 (and 30 if TSO LOGON events are relevant). These records are passed to the SMF record exit (IEFU83 and IEFU84), which transfers information regarding the commands to the Online Interceptor.

To ensure that these records are collected for future processing and are passed to SMF exit IEFU83 and optionally IEFU84, SMF parameters must be set to collect generated records of type 80 and optionally 30.

2A – Edit the SMFPRMnn Member

1. Edit the SMFPRMnn member in your SYS1.PARMLIB library. This member specifies which SMF records are collected by SMF.

   nn is the number specified in member IEASYS in SYS1.PARMLIB, or 00, if it was not specified.

2. Make sure parameter TYPE specifies that record type 80 is collected for all subsystems.

   For more information on SMFPRMnn parameters, refer to the *z/OS Initialization and Tuning Reference Guide*.

3. Save the member (if it was modified).

2B – Activate SMF Parameters

If you modified member SMFPRMnn in Step "2A–Edit the SMFPRMnn member", refresh SMF definitions by issuing the following operator command:

```
SET SMF=nn
```

where *nn* is the suffix of the SMFPRMnn member that was updated.

Install and Customize Online Interceptor

For general information regarding Connector Interceptor exits, see Overview of Connector for RACF Interceptors.

The following exits must be installed in the system:

- **SMF record exit IEFU83** - Intercepts RACF commands and transfers information to the Connector Online Interceptor regarding each command.
- **SMF record exit IEFU84** (*Optional*)- Required if any of the following is relevant:

  - Supporting Security zSecure Admin RESUME/REVOKE
  - AUTOPROF SMF records need to be intercepted
  - Intercept of TSO LOGON events by Online Interceptor is required

The following exits must be installed only if password change event interception is required:

- RACF postinit exit ICHRIX02 - Supports password synchronization
- RACF new password exit ICHPWX01 - Supports password synchronization

Exits ICHRIX02 and ICHPWX01 can be installed using the dynamic or static installation process (described in step 3 – Install RACF Exits ICHRIX02 and ICHPWX01).

> **Note:**
>
> The following steps (1 – Install SMF Exit IEFU83, 2 – (Optional) Install SMF exit IEFU84, and 3 – Install RACF Exits ICHRIX02 and ICHPWX01) in the installation process include the installation of system exits IEFU83, IEFU84, ICHRIX02, and ICHPWX01.
>
> For more information regarding exits IEFU83 and IEFU84, refer to your *Z/OS Installation Exits Guide*.
>
> For more information about exits ICHRIX02 and ICHPWX01, refer to the *System Programming Library: RACF Guide*.
>
> The usage of these exits is described in further detail under Online Interceptor Logic.

1 – Install SMF Exit IEFU83

Installation of Connector IEFU83 exits creates the IEFU83 load module in the system LPA library.

The following members in the INSTALL library involved in this installation procedure:

Member	Description
RCFU83A	Connector IEFU83 SMF exit source code
ASMU83A	Sample job to assemble and link CTSU83A
CPYU83A	Sample job to copy CTSU83A module to the system library as IEFU83A

To Install SMF exit IEFU83:

1. Create the CTSU83A exit module.

   Member ASMU83A in the Connector INSTALL library is a sample member to assemble and link Connector exit IEFU83.

   Review the JCL carefully and submit the job to create the CTSU83A load module in the Connector LOAD library. All job steps must end with a condition code of `0`, except for the following:

   - **CHECRCF** – Ends with the condition code `4`.

   - **ACFU83A** – This step is not executed.

2. (*Optional*) If you wish to have the exit module reside permanently in the LPA, copy CTSU83A to your LPA system library; otherwise proceed directly to step 3.

   Edit member CPYU83A in the Connector INSTALL library. The job copies load module CTSU83A to your system LPA library. Review the job carefully and submit the job. All job steps must end with a condition code of `0`.

3. Define the SMF exit in the MVS dynamic exits facility.

   Edit member PROG*nn* in SYS1.PARMLIB to define the new SMF exit module for all SMF sub-systems.

   - If you copied CTSU83A to the LPA system library, add the following statements to the member:

     ```
     EXIT ADD EXITNAME(SYS.IEFU83) MODNAME(CTSU83A)
     EXIT ADD EXITNAME(SYSSTC.IEFU83) MODNAME(CTSU83A)
     EXIT ADD EXITNAME(SYSTSO.IEFU83) MODNAME(CTSU83A)
     ```

     Save the member.

   - If you did not copy CTSU83A to the LPA system library, add the following statements to the member:

     ```
     EXIT DELETE EXITNAME(SYS.IEFU83) MODNAME(CTSU83A)
     EXIT ADD EXITNAME(SYS.IEFU83) MODNAME(CTSU83A)
     DSNAME(CTSA Load Library)
     EXIT DELETE EXITNAME(SYSSTC.IEFU83) MODNAME(CTSU83A)
     EXIT ADD EXITNAME(SYSSTC.IEFU83) MODNAME(CTSU83A)
     DSNAME(CTSA Load Library)
     EXIT DELETE EXITNAME(SYSTSO.IEFU83) MODNAME(CTSU83A)
     EXIT ADD EXITNAME(SYSTSO.IEFU83) MODNAME(CTSU83A)
     DSNAME(CTSA Load Library)
     ```

   - If you have more subsystems defined in your SMFPRMnn (see step 5), other than STC and TSO like (JES2), add a proper definition for it, similar to the definitions for TSO and STC.

If JES3 is active in your environment, add the following line as well:

```
EXIT ADD EXITNAME(SYSJES2.IEFU83) MODNAME(CTSU83A)
DSNAME(CTSA Load library)
```

Save the member.

4. If you performed the optional step 2, activate the exit module CTSU83A in LPA system library, otherwise proceed directly to step 5.

   Once the exit module resides in your LPA library, activate the module. This is usually performed by the next IPL with the CLPA option.

   If you have a product which can dynamically load modules to the LPA (for example, RESOLVE, LOOK, OMEGAMON) you can use it to add the exit to the LPA. However, when loaded in this manner, the exit remains active only until the next IPL.

5. Set SMF parameters.

   Member SMFPRMnn in SYS1.PARMLIB defines the SMF exits that are used by SMF.

   Make sure that the exit IEFU83 is defined for all SMF subsystems, and record type 80 is defined to all subsystems.

   Save the member.

6. Activate the SMF exit.

   - If the new IEFU83 exit was dynamically loaded to the LPA (as described in step 5), refresh the SMF definitions by issuing the following operator command:

     ```
     SET SMF=nn
     ```

     where *nn* is the suffix of the SMFPRMnn member that is updated.

   - Otherwise, an IPL should be executed with the CLPA option to activate the new exit. Activate the exit by issuing the following operator command:

     ```
     SET PROG=nn
     ```

     where *nn* is the suffix of the PROGnn member which was updated in step 3

> **Note**
> Installation of the SMF exit IEFU83 is complete.

2 – (*Optional*) Install SMF exit IEFU84

Install SMF exit IEFU84, if any of the following features are required:

- Supporting Security zSecure Admin RESUME/REVOKE (For full description, see "Supporting Security zSecure Admin RESUME/REVOKE" on page 66)
- AUTOPROF SMF records are relevant and need to be intercepted (For full description, see "AUTOPROF SMF records support" on page 72).
- Intercept of TSO LOGON events by Online Interceptor is required (For full description, see page 72).

To install SMF exit IEFU84

1. Create the CTSU84A exit module.

   Member ASMU84A in the INSTALL library is a sample member to assemble and link exit IEFU84.

   Review the JCL carefully and submit the job to create CTSU84A load module in LOAD library. All job steps must end with a condition code of `0`, except for:

   - CHECACF - Ends with the condition code `4`.
   - ACFU84A - Not executed.

2. If you wish to have the exit module reside permanently in the LPA, copy CTSU84A to your LPA system library.

   Job CPYU84A copies load module CTSU84A to your system LPA library.

   Review the job carefully and submit the job. All job steps must end with a condition code of `0`.

3. Define the SMF exit in MVS dynamic exits facility.

   Edit member PROGnn in SYS1.PARMLIB to define the new SMF exit module, for all SMF sub-systems.

   If you copied CTSU84A to the LPA system library, add the following statements to the member:

```
EXIT ADD EXITNAME(SYS.IEFU84) MODNAME(CTSU84A)
EXIT ADD EXITNAME(SYSSTC.IEFU84) MODNAME(CTSU84A)
EXIT ADD EXITNAME(SYSTSO.IEFU84) MODNAME(CTSU84A)
```

   If you did not copy CTSU84A to the LPA system library, add the following statements to the member:

```
EXIT DELETE EXITNAME(SYS.IEFU84) MODNAME(CTSU84A)
EXIT ADD EXITNAME(SYS.IEFU84) MODNAME(CTSU84A)
DSNAME(CTSA Load Library)
EXIT DELETE EXITNAME(SYSSTC.IEFU84) MODNAME(CTSU84A)
EXIT ADD EXITNAME(SYSSTC.IEFU84) MODNAME(CTSU84A)
DSNAME(CTSA Load Library)
EXIT DELETE EXITNAME(SYSTSO.IEFU84) MODNAME(CTSU84A)
EXIT ADD EXITNAME(SYSTSO.IEFU84) MODNAME(CTSU84A)
DSNAME(CTSA Load Library)
```

If you have more subsystems defined in your SMFPRMnn other than STC and TSO (like JES2), add a proper definition for it, similar to the definitions for TSO and STC.

4. If you performed the optional step 2, activate the exit module CTSU84A in LPA system library, otherwise proceed directly to the next step.

   Once the exit module resides in your LPA library, activate the module. This is usually performed by the next IPL with the CLPA option.

   If you have a product which can dynamically load modules to the LPA (for example, RESOLVE, LOOK, OMEGAMON) you can use it to add the exit to the LPA.

   However, when loaded in this manner, the exit remains active only until the next IPL.

   Set SMF parameters

   Member SMFPRMnn in SYS1.PARMLIB defines the SMF exits that are used by SMF. Ensure that the exit IEFU84 is defined for all SMF subsystems and record type 80 and record type 30 if required for TSO logon and logoff time are defined to all subsystems.

5. Activate the SMF exit.

   If the new IEFU84 exit was dynamically loaded to the LPA, refresh SMF definitions by issuing the following operator command, where *nn* is the suffix of the SMFPRMnn member that is updated:

   ```
   SET SMF=nn
   ```

   Otherwise, an IPL should be executed with the CLPA option to activate the new exit. Activate the exit by issuing the following operator command:

   ```
   SET PROG=nn
   ```

   where *nn* is the suffix of the PROGnn member which was updated in step 3.

   > **Note**
   > Installation of SMF exit IEFU84 is complete.

3 – Install RACF Exits ICHRIX02 and ICHPWX01

> **Note**
> Install the new password exit only if support for password synchronization support is required.

RACF exits ICHRIX02 and ICHPWX01 can be installed using either of the following methods:

- Dynamic installation
- Static installation

The *dynamic installation* process implies that when the Connector Online Interceptor is started it loads the exit modules and activates them. When the Connector Online Interceptor is terminated it deactivates the exits and removes them from the system.

If a system exit is already installed at the site, on each exit call, the dynamically-loaded Connector exit is executed first, followed by the existing system exit.

The *static installation* process implies that the exit is compiled and linked to a system LPA library (for example, SYS1.LPALIB) and is activated automatically after an IPL (or using a product which can dynamically load modules to the LPA).

For the static installation process, a *sample multiple exits driver* is provided to handle situations where exit modules for ICHRIX02 or ICHPWX01 already exist at the site. The multiple exits driver can be configured to call the pre-existing exit modules in addition to the exit modules supplied by Connector for RACF.

> **Note**
>
> The dynamic installation method may conflict with several other programs that rely on the absolute addresses of the ICHRIX02 and ICHPWX01 exits in the RACF CVT. As a result, the message ICH66107I will be issued.
>
> *This does not imply a system failure, as the Online Interceptor concatenates ICHRIX01 and ICHPWX01 to the Connector for RACF exit.*
>
> If you want to avoid receiving this message, use the static installation process, described in Method 2 - Static Installation. If two or more instances of Connector for RACF on the same computer use the dynamic installation of the new password exit, restrictions apply to the order in which the Online Interceptors for each instance of Connector for RACF are shut down. For more information, see Starting and Stopping the Online Interceptor.

Method 1 - Dynamic Installation

To perform a dynamic installation of RACF exits ICHRIX02 and ICHPWX01:

Edit the RSSPARM member in the Connector PARM library.

The following parameters define whether the Connector Online Interceptor will dynamically install ICHRIX02 and ICHPWX01 when it is started.

```
ONLI_DYNAM_RIX02
```

```
ONLI_DYNAM_PWX01
```

Set each parameter to `Y` and save the member.

Dynamic installation of RACF exits ICHRIX02 and ICHPWX01 is complete. Exits ICHRIX02 and ICHPWX01 are dynamically installed when the Online Interceptor is started.

Method 2 - Static Installation

> **Note**
> Do not continue with this procedure if you have already used Method 1 (above) to install exits ICHRIX02 and ICHPWX01.

Static installation of Connector ICHRIX02 and ICHPWX01 exits create the ICHRIX02 and ICHPWX01 load modules in the system LPA library.

The following members in the INSTALL library involved in this procedure are listed in the following table.

Member	Description
CTSRIX2A	Connector ICHRIX02 RACF exit source code.
CTSRIX2B	Connector for RACF sample multiple ICHRIX02 RACF exits driver source code.
CTSPWX1A	Connector ICHPWX01 RACF exit source code.
CTSPWX1B	Connector for RACF sample multiple ICHPWX01 RACF exits driver source code.
ASMRIX2A	Sample job to compile and link CTSRIX2A.
ASMRIX2B	Sample job to compile and link CTSRIX2B.
CPYRIX2A	Sample job to copy CTSRIX2A module to the system LPA library as ICHRIX02.
CPYRIX2B	Sample job to copy CTSRIX2B module to the system LPA library as ICHRIX02.
ASMPWX1A	Sample job to assemble and link CTSPWX1A.
ASMPWX1B	Sample job to assemble and link CTSPWX1B.
CPYPWX1A	Sample job to copy CTSPWX1A module to the system LPA library as ICHPWX01.
CPYPWX1B	Sample job to copy CTSPWX1B module to the system LPA library as ICHPWX01.

To perform static installation of RACF exits ICHRIX02 and ICHPWX01:

1. Create the CTSRIX2A exit module.

   Member ASMRIX2A in the Connector INSTALL library is a sample member used to assemble and link Connector for RACF exit ICHRIX02.

   Review the JCL carefully and submit the job to create the CTSRIX2A load module. All job steps must end with a condition code of `0`.

2. Create the CTSPWX1A exit module.

   Member ASMPWX1A in the Connector INSTALL library is a sample member used to assemble and link Connector for RACF exit ICHPWX01.

   Review the JCL carefully and submit the job to create the CTSPWX1A load module in the Connector LOAD library. All job steps must end with a condition code of `0`.

   Customize the multiple exits driver exit table.

   The *multiple exits driver* is provided to handle situations where exit modules for ICHRIX02 or ICHPWX01 already exist at the site. The multiple exits driver can be configured to call the pre-existing exit modules in addition to the exit modules supplied by Connector for RACF.

   If you do not require the multiple exits driver, skip to step 6.

   The multiple exits driver source code contains a table in which you must define the ICHRIX02 and ICHPWX01 exit modules that need to be called. The multiple exits driver calls each module in sequence as specified in the table. Each exit in the table is called, regardless of the return code of the preceding exit. The multiple exits driver returns the highest return code of all the exits that were called.

   Edit members CTSRIX2B and CTSPWX1B in the Connector INSTALL library. Locate the exits table at the end of each source member and customize it to call your local exit modules.

   Save the members.

3. Create the CTSRIX2B exit module.

   Member ASMRIX2B in the Connector INSTALL library is a sample member used to assemble and link the Connector for RACF sample driver for multiple ICHRIX02 exits.

   > **Note**
   > You must add an INCLUDE statement to the linkage editor input stream so that the exits driver is linked with your local exit.

   Review the JCL carefully and submit the job to create the CTSRIX2B load module in the Connector LOAD library. All job steps must end with a condition code of `0`.

4. Create the CTSPWX1B exit module.

   Member ASMPWX1B in the Connector INSTALL library is a sample member to assemble and link the Connector for RACF sample driver for multiple ICHPWX01 exits.

   > **Note**
   > You must add an INCLUDE statement to the linkage editor input stream so that the exits driver is linked with your local exit.

Review the JCL carefully and submit the job to create the CTSPWX1B load module. All job steps must end with a condition code of `0`.

5. Copy CTSRIX2B and CTSRIX2A to your LPA system library.

   a. Edit member CPYRIX2B in the Connector INSTALL library. The job copies CTSRIX2B to your system LPA library as ICHRIX02.

   b. Edit member CPYPWX1B in the Connector INSTALL library. The job copies CTSPWX1B to your system LPA library as ICHPWX01.

   > **Note**
   > Any previous copy of exit ICHRIX02 or ICHPWX01 in the LPA library is overwritten after the job completes. It is recommended that you create a backup copy of each previous exit.

   Review the job carefully and submit the job. All job steps must end with a condition code of `0`. Continue to step 7.

6. Copy CTSRIX2A and CTSPWX1A to your LPA system library.

   > **Important**
   > If you are using the multiple exits driver, skip this step and continue to step 7.

   a. Edit member CPYRIX2A in the Connector INSTALL library. The job copies CTSRIX2A to your system LPA library as ICHRIX02.

   b. Edit member CPYPWX1A in the Connector INSTALL library. The job copies CTSPWX1A to your system LPA library as ICHPWX01.

   > **Note**
   > Any previous copy of exit ICHRIX02 or ICHPWX01 in the LPA library is overwritten after the job completes. It is recommended that you create a backup copy of each previous exit.

   Review the jobs carefully and submit the jobs. All job steps must end with a condition code of `0`.

7. Set the Online Interceptor dynamic exit parameters.

   Edit member RSSPARM in the Connector PARM library.

   The following parameters define whether or not the Connector Online Interceptor will dynamically install ICHRIX02 and ICHPWX01 when it is started.

```
ONLI_DYNAM_RIX02
```

```
ONLI_DYNAM_PWX01
```

Set the value of each parameter to `N` and save the member.

8. Activate the exits in LPA.

   Once the exit modules reside in your LPA library, you need to activate them. This is performed by the next IPL with the CLPA option.

   > **Note**
   > Static installation of RACF exit ICHRIX02 and ICHPWX01 without SMP/E is complete.

4 – Verify that the Online Interceptor is Installed Properly

Verify that the Online Interceptor is installed properly by issuing the following operator command:

```
S CTSAONI
```

The Online Interceptor starts and begins recording RACF events to the RACF Connector Queue dataset.

5 – Update IdentityIQ Definitions

IdentityIQ definitions must be updated to allow asynchronous events to be sent from RACF connector to IdentityIQ.

For more information, see the "Settings for configuring password Interceptor and Online Interceptor" section of *SailPoint Quick Reference Guide for Gateway Connectors*.

6 – Configure Automated Startup of Online Interceptor

Make the appropriate changes to the system startup procedures to start Connector for RACF Online Interceptor automatically after IPL.

If an automatic startup tool is used to start all the required applications after system initialization, add CTSAONI to this automatic startup tool definitions.

Otherwise, add the following command to COMMNDnn member in SYS1.PARMLIB to start the Online Interceptor after IPL:

```
S CTSAONI
```

7 – (*Optional*) Controlling Online Interceptor Memory Consumption

You can control memory size used by **Online Interceptor**. This can be achieved by setting the optional parameter ONLI_MAX_EVENTS in the RSSPARM member in the PARM dataset.

ONLI_MAX_EVENTS

ONLI_MAX_EVENTS set the maximum number of events that the Online Interceptor can accumulate in memory. ONLI_MAX_EVENTS is optional. If this parameter does not exist in the RSSPARM member, the default value of 20,000 events is used (or about 50 MB in the Online Interceptor's address space).

The minimum value for ONLI_MAX_EVENTS is 2,000 events. Each entry occupies 2,560 bytes in memory, depending on the event type (user, group, connection, or password) and length of the userid, group, or password.

Use the following syntax for this parameter:

```
managedSystemName     ONLI_MAX_EVENTS     numberOfEvents
```

ONLI_MIN_NOTIFY_EVENT%

The optional parameter ONLI_MIN_NOTIFY_EVENT% sets a memory threshold before issuing a warning message that the memory for accumulating events may soon be entirely filled. When the stated percentage of available memory for accumulating events is filled, the message CTS4509W (described below) is issued.

Use the following syntax for this parameter:

```
managedSystemName     ONLI_MIN_NOTIFY_EVENT%     value%
```

> **Note**
> The value specified must be followed by the **%** symbol. If the ONLI_MIN_NOTIFY_EVENT% parameter is not specified in the RSSPARM member, the default value of 10% is used.

For example, given the following settings:

```
RSS1     ONLI_MAX_EVENTS            5000
RSS1     ONLI_MIN_NOTIFY_EVENT%     20%
```

You can expect the following:

- Space for 5000 intercepted events is allocated in memory (approximately 12.5 MB).
- Message CTS4509W will be issued when only 20% of the allocated space remains (or, when more than 4,000 events reside in memory).

8 – Configure Event Interception Filtering

Event Interception filtering enables you to control which entity events intercepted in the Managed System by the Online Interceptor are reported to IdentityIQ.

The following RSSPARM parameters enable you to filter the interception of events on the Managed System:

- **ONLI_EVENT_GROUP** – Specifies whether the Online Interceptor should report group events to IdentityIQ.

  Possible values for this parameter are `Y` or `N`.

- **ONLI_EVENT_USER** – Specifies whether the Online Interceptor should report user events to IdentityIQ.

  Possible values for this parameter are `Y` or `N`.

- **ONLI_EVENT_USER_PWD_ONLY** – Specifies which intercepted user events are reported to IdentityIQ.

  > **Note**
  > This parameter is only relevant when ONLI_EVENT_USER is set to `Y`.

  Possible values for this parameter are:

  - **Y** – Only user password change events are reported to IdentityIQ.

  - **N** – (Default) All intercepted user events are reported to IdentityIQ.

To filter the interception of Managed System events:

1. Stop Connector for RACF Online Interceptor.

   For more information, see Starting and Stopping the Online Interceptor.

2. Edit the RSSPARM member in the Connector PARM library.

3. Insert or modify one or more of the following parameters as necessary.

```
rss_name ONLI_EVENT_USER_PWD_ONLY  N      <== Modify as required
rss_name ONLI_EVENT_USER           Y      <== Modify as required
rss_name ONLI_EVENT_GROUP          Y      <== Modify as required
```

4. Save the member.

5. Restart the Connector for RACF Online Interceptor.

   For more information, see Starting and Stopping the Online Interceptor.

Install Offline Interceptor Interface

The Connector for RACF Offline Interceptor is intended to be used following a period of time during which the Online Interceptor was not active, or if the Online Interceptor is not installed.

While the Online Interceptor intercepts RACF administration events as they occur, the Offline Interceptor reads SMF datasets in which RACF has logged events that have already occurred.

Before they are processed, SMF records in the SYS1.MANx datasets should be copied into a sequential dataset by SMF utility IFASMFDP.

JCL library member RCFAOFI includes a sample job for activating the program IFASMFDP. During activation of IFASMFDP, a sequential dataset is created that includes only SMF80 and SMF30 records. These datasets are necessary for the Offline Interceptor started task (CTSAOFS).

Most MVS sites dump the contents of the SYS1.MANx datasets to a sequential dataset as a step in a daily job or when the SYS1.MANx becomes full.

Examination of the SYS1.MANx dataset directly by the Offline Interceptor may not be useful since SYS1.MANx may have already been cleared. This could result in events being missed by the Offline Interceptor.

Therefore, you should integrate the Connector Offline Interceptor into the SMF processing jobs which clear the SYS1.MANx datasets at your site.

For more information about the Offline Interceptor, see Offline Interceptor Logic.

Integrate Offline Interceptor into SMF Processing Jobs

The Offline Interceptor must be activated as a step in the SMF datasets' processing jobs.

Member RCFAOFI in the Connector JCL library is a sample job that demonstrates the use of the CTSAOFI JCL procedure by a job. The procedure activates the Connector Offline Interceptor with an input dataset that is passed as a parameter to the procedure.

Add a step to the SMF processing job to activate JCL procedure CTSAOFI. The value of the SMFDATA parameter should be the name of the SMF dataset holding the records to be processed by the Offline Interceptor.

The Connector Offline Interceptor processes SMF records of type 80 and 30 (Type 30 is relevant only when intercept of TSO LOGON events is required). Verify that records of type 80 and 30 (if needed) are collected in the previous job steps and that they are not filtered, by updating PARMOFI member in PARM library.

Implementing the Standard Offline Interceptor

The Standard Offline Interceptor is a process that can be invoked to intercept security administration events initiated directly in the managed system.

> **Note**
> SailPoint recommends that you continue using the Online Interceptor or the SMF OfflineInterceptor. Standard Offline Interceptor is intended for sites that have developed their own Managed System Interface and do not want to develop a specific interceptor for their own Managed System Interface.

The Standard Offline Interceptor typically scans all Managed System objects it monitors in order to determine what events have occurred since the last Standard Offline Interceptor invocation.

Required RACF Definitions

> **Note**
> Whenever the CTSOFLI procedure is mentioned, CTS should be replaced with the relevant three-letters prefix used for the procedures in the Connector for RACF installation.

The definitions below must be set in RACF if the Standard Offline Interceptor is used.

These definitions were set in the CTSRACF member in INSTALL library as comments.

For more information, see Step 10 – Define Connector for RACF in RACF.

```
ADDUSER CTSOFLI NAME('STANDARD OFFLINE INTERCEPTOR') -
    DFLTGRP(CTSTASKS) OWNER(CTSTASKS)
RDEFINE STARTED CTSOFLI.* -
    STDATA(USER(=MEMBER),GROUP(CTSTASKS))
PERMIT '%OLPREFS%.%OLVERS%.ENCRINT' -
    ACCESS(READ) ID(CTSOFLI) GENERIC
```

In addition, the following permission is required for the catalog in which the Standard Offline Interceptor files will be cataloged.

```
PERMIT 'CATALOG.*.**' -
    ACCESS(UPDATE) ID(CTSOFLI)
```

Setting the Volume for Standard Offline Interceptor Working Datasets

During Standard Offline Interceptor operation, working datasets are created and must be retained until the next run of the Offline Interceptor.

> **Note**
> Whenever the CTSOFLI procedure is mentioned, CTS must be replaced with the relevant three-letters prefix used for the procedures in the Connector for RACF installation.

The Offline Interceptor datasets would be allocated using the values specified for OLUNITS and OLVOLS during installation (see Connector for RACF Datasets Allocation Parameters).

To set other values, change the values set for OLUNITS and OLVOLS in CTSOFLI procedure parameters.

Step **ALLOC1** in member **CTSOFLI** allocates the files listed in the following table.

Member Name	Description
OFLUTMP	Users temporary file
OFLGTMP	Groups temporary file

Member Name	Description
OFLCTMP	Connections temporary file
OFLOTMP	Containers temporary file
OFLRTMP	RSS parameters temporary file
OFLUCMP	File for comparing users
OFLGCMP	File for comparing groups
OFLCCMP	File for comparing connections
OFLOCMP	File for comparing containers
OFLRCMP	File for comparing Managed System parameters

Step **ALLOC2** in member **CTSOFLI** allocates the files listed in the following table.

Member Name	Description
OFLUIMG	Users checksum file
OFLGIMG	Groups checksum file
OFLCIMG	Connections checksum file
OFLOIMG	Containers checksum file
OFLRIMG	RSS parameters checksum file

The size of file allocations should be determined according to number of users, groups and connections in RACF.

In a 3390 device, one cylinder is approximately 840 KB (56 KB per track, 15 tracks per cylinder: 56 KB * 15 = 840 KB).

For each user or group, 20 bytes should be calculated for allocation of **OFLUTMP**, **OFLGTMP**, **OFLUCMP**, **OFLGCMP**, **OFLUIMG** and **OFLGIMG** files.

For each connection, 30 bytes should be calculated for allocation of **OFLCTMP**, **OFLCCMP**, and **OFLCIMG** files.

For example, for 50,000 users 2 cylinders should be allocated:

(20 * 50,000) / 840,000 = 1.19 cylinders.

This exceeds 1 cylinder, and thus should be considered as 2 cylinders.

> **Important**
> Files **OFLOTMP**, **OFLOCMP**, **OFLOIMG**, **OFLRTMP**, **OFLRCMP** and **OFLRIMG** allocation size should not be changed in **CTSOFLI**.

Scheduling the Standard Offline Interceptor

The Standard Offline Interceptor can be scheduled independently of Connector for RACF, either manually or automatically via the Notification Server.

The following options exist for scheduling activation of the Offline Interceptor:

- The Notification Server can schedule the Offline Interceptor to run at fixed intervals, starting from the time Connector for RACF is activated. The interval is specified using parameter OFLI_INTERVAL, described below.
- The Notification Server can schedule the Offline Interceptor to run at specific times during the day. The run times are specified using parameter OFLI_RUN_TIME_LIST, described below.
- You can schedule the Offline Interceptor using an external facility.

When scheduled by the Notification Server, the Notification Server will not begin scheduling the Offline Interceptor until after the initial aggregation of managed system data to IdentityIQ has been performed.

Depending on the method used to schedule the Offline Interceptor, it may be necessary for Notification Server to be aware of the exact time the Offline Interceptor was last activated.

For this purpose, Connector for RACF maintains the `<prefix>.<version>.RSSOFLI` file. This file contains the invocation time of the Offline Interceptor and saves the time stamp of the last event that it processed.

Upon initialization, the Offline Interceptor obtains the time stamp from its previous invocation from function CTSInterceptorInit. When the Offline Interceptor completes its operation, the updated time stamp is written to the RSSOFLI file by calling function CTSInterceptorTerm.

Starting the Standard Offline Interceptor Manually

To start the Standard Offline Interceptor manually, issue the following operator command to operate the Standard Offline Interceptor as a started task (the **CTSOFLI** procedure is located in the system **PROCLIB** library):

```
S CTSOFLI
```

> **Note**
> Ensure that the Standard Offline Interceptor has been configured before operating for the first time. For more information, see Setting the Volume for Standard Offline Interceptor Working Datasets.

Customize Delayed Delete Utility

When a user or group is deleted from the Managed System via IdentityIQ, the Connector can ensure that the user’s or group’s UIDs or GIDs are deleted from other profiles in the RACF database that refer to the user or group. This cleanup is accomplished via parameters in IdentityIQ and the Connector utility CTSC100.

This section provides an overview of deleting users and groups and describes how to customize the Delayed Delete utility to perform deletion of these entities in an efficient manner.

Overview

Deleting RACF users and groups is generally not as straightforward as simply deleting a profile because other profiles in the RACF database may refer to the user or group. For example:

- Dataset profiles may exist that have the user ID (UID) or group ID (GID) as their high level qualifier.
- The group or user may be the owner of other profiles (for example, user profiles, group profiles, or resource profiles).
- The user or group may be authorized to access resources, and therefore appear in the resources' access lists.
- The group may still have subgroups or users connected to it.

Some of the above conditions prevent the deletion of a user or group. Others do not prevent the deletion, but leave the RACF database with redundant information after the user or group is deleted.

RACF cross-reference utility IRRUT100 can be used to find all occurrences of a UID or GID in the RACF database. This utility, however, does not delete the user ID or group ID occurrences from the database.

Cleanup of Deleted UIDs and GIDs

The Connector for RACF, upon receiving a request from IdentityIQ to delete a user or a user group, can perform the necessary cleanup so that all occurrences of the user's or group's UID or GID are deleted from the RACF database.

The Connector for RACF accomplishes this by activating the RACF utility IRRUT100, analyzing its output, building a dataset containing a REXX script with RACF commands to perform the cleanup, and optionally executing this dataset.

These operations are controlled by the MODE parameter in the delete request from IdentityIQ.

For a description of all delete request parameters, see **CTSDelUser** in User Connector for RACF Function Descriptions.

Delayed Delete

The execution of the RACF utility IRRUT100 can be a resource-consuming operation. In addition, if more than one user or group is deleted, the RACF database scan is executed for each delete request separately.

The Connector for RACF's delayed delete function enables you to postpone the cleanup operation (for example, until midnight), and facilitates a single database scan for several delete requests. This results in more efficient use of resources. See Renaming a Managed System for more information.

Customizing the Delayed Delete Utility

Use the procedure that follows to customize the Delayed Delete utility to satisfy the requirements in your site.

1 - Set Delayed Delete Support Parameters

Edit member RSSPARM in Connector for RACF PARM library.

If the Delay parameter is specified in a delete request from IdentityIQ, but not supported in Connector for RACF, the delete request fails and an error message is returned to IdentityIQ.

> **Note**
> These parameters have no effect on delete requests which do not specify the Delay parameter.

1. Modify the following parameters:

Parameters	Option	Description
RCF_DELAY_DELUSR	Y	The Delay parameter for delete user requests is supported.
	N	The Delay parameter for delete user requests is not supported.
RCF_DELAY_DELGRP	Y	The Delay parameter for delete group requests is supported.
	N	The Delay parameter for delete group requests is not supported.

2. Save the member.

2 - Set Up CTSC100 Utility Activation

Connector for RACF utility CTSC100 must be executed periodically to perform the delayed delete requests.

Member CTSC100 in the Connector for RACF JCL library is a sample job which activates utility CTSC100.

To schedule periodic execution of utility CTSC100:

1. Prepare a job in your scheduler’s JCL library to activate utility CTSC100.

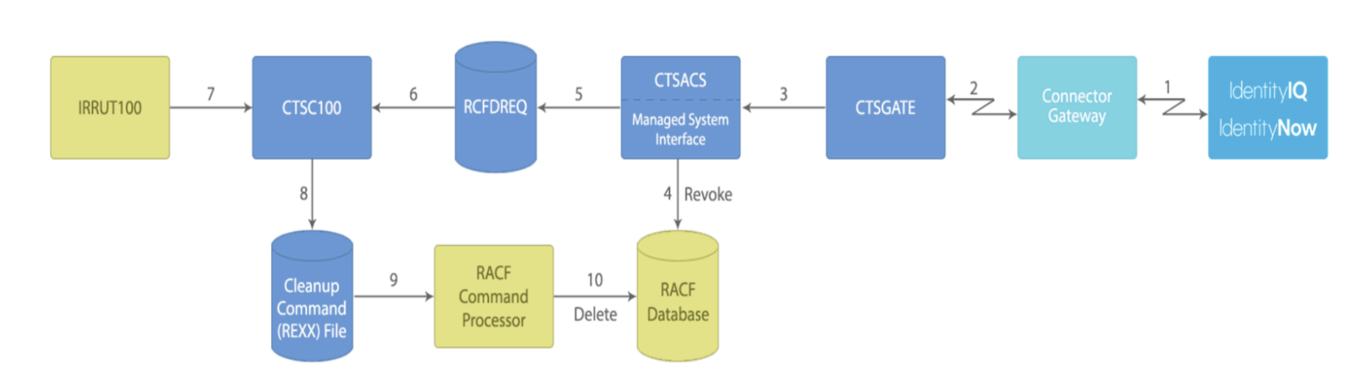
2. Set the job scheduling parameters in your scheduler environment to run job CTSC100 periodically.

> **Note**
>
> If STCJOBS are used for the product started tasks, the file allocated for EXECOUT DD Statement is a permanent file.
>
> The user submitting the job must have ALTER authority to this file.
>
> The file name is: `<olprefs>.<olvers>.EXECOUT.<procprefs>C100`

> where `olprefs`, `olvers`, and `procprefs` are the values set in Step 1 – Set the Parameter Values.

Handling Delayed Delete Requests

The Connector's utility CTSC100 and IdentityIQ's Delay parameter work together to provide the capability for delayed delete as illustrated in the following figure.



When the Delay parameter is used in a delete request, IdentityIQ instructs Connector not to perform the delete request immediately, but to queue it for delayed processing [1].

The delete request is written to Connector dataset `<prefix>.<version>.RCFDELRQ` [2] and the user is revoked [3]. This ensures that the user is denied access until actually deleted.

When utility CTSC100 is run [4], it processes the request dataset and activates IRRUT100 once [5] to process all the requests specified in the dataset. CTSC100 processes IRRUT100 output and creates a REXX exec dataset with cleanup commands for each of the requests [6]. The dataset contains RACF commands to delete all occurrences of the user or group.

CTSC100 then optionally executes all of the cleanup REXX exec datasets [7] and deletes the users or groups that were processed [8].

If the utility is not used to execute the cleanup, the administrator is responsible for manually executing the cleanup commands in the REXX exec dataset.

This procedure must be executed periodically to handle the delayed delete requests. Member CTSC100 in the Connector JCL library is a sample job which activates this procedure.

Shared RACF Database Support

A shared RACF database environment consists of two or more RACF systems which share the same RACF database.

For more information, see Interceptors in a Shared RACF Database Environment.

In a shared database configuration, one of the systems runs a full instance of Connector for RACF. This system is referred to as the primary system. Each additional RACF system sharing the database is referred to as a *secondary system*. Each secondary system only runs one component of Connector for RACF the Interceptor.

The procedure that follows describes installing the Interceptor on a secondary system.

Install Interceptor on a Secondary System

The secondary system can intercept changes and events in a system via one the following components:

- Online Interceptor (CTSAONI)
- Offline Interceptor (CTSAOFI)

After installing Connector for RACF on the primary system, you must install either the Online or Offline Interceptor on each additional secondary system which shares the RACF database.

Secondary System Online Interceptor

To install the Online Interceptor on a secondary system:

1. Define Connector QNAME to the ENQ manager.

   The Connector for RACF serializes access to its QUEUE dataset by issuing an ENQ with the SYSTEMS option.

   To allow synchronized update of the QUEUE dataset from multiple systems, you should update your external ENQ manager (for example, GRS or MIM) with the Connector QNAME to be propagated to all systems in the external ENQ manager complex.

   Another option is to let the Connector for RACF use the RNL=NO ENQ/DEQ parameter telling GRS to always propagate the ENQ to all systems in the GRS complex. This can be controlled using the ENQRNL parameter in member CTSPARM in Connector for RACF PARM library.

   The default QNAME is xxxASYNC where xxx is the value set for %PROCPREFS% in the DEFPARMS member in the INSTALL library. The default may be customized by changing the QNAME parameter in member CTSPARM in Connector PARM library.

   For explanations on how to change and activate these parameters, refer to CTSPARM – Assembler Format Parameters.

2. Share the datasets.

   Place the datasets listed in the following table on a shared DASD so that they can be shared by the primary and secondary systems.

Dataset Name	Description
prefix.version.QUEUE	Connector QUEUE dataset
prefix.version.LOAD	Connector Load library
prefix.version.CMSG	Connector Msgs library
prefix.version.PARM	Connector PARM library
prefix.version.DIAGLVL	Connector Diagnostics library
prefix.version.ENCRINT	Connector for RACF Stored Data Encryption dataset

3. Copy CTSAONI procedure and, if needed, STCJOB:

   - If STCJOBS are not used, copy the CTSAONI JCL procedure from the system PROCLIB of the primary system to the system PROCLIB of the secondary system.

   - If STCJOBS are used:

     - Copy the CTSAONI STCJOB member to the system STCJOBS library.

     - If the value LOCALCOPY is set for %PROLCIB% in the DEFPARMS member in the INSTALL library, make sure the secondary system has access to the Connector for RACF PROCLIB library. Otherwise, copy the CTSAONI JCL procedure from the system PROCLIB of the primary system to the system PROCLIB of the secondary system.

4. In the secondary system(s):

   a. Add the Connector LOAD library to the APF authorized libraries list.

   b. Define the CTSAONI started task to RACF.

      For details, refer to 10.1 – Define Connector for RACF Started Tasks in RACF.

   c. Set RACF and SMF logging options. For more information, refer to Set Logging Options.

   d. Install Online Interceptor system exits (**IEFU83**, **IEFU84** (as needed), **ICHRIX02** and **ICHPWX01**).

      For more information refer to Install and Customize Online Interceptor .

   e. Start the Online Interceptor.

   f. After verification, configure automated startup of the Online interceptor.

      For details refer to 6 – Configure Automated Startup of Online Interceptor.

> **Note**
> When we have a secondary system Online Interceptor, it needs the Stored Data Encryption key, so it can write to the QUEUE. However, it does not need the Transmitted Data Encryption key because it does not communication with IdentityIQ.

Secondary System Offline Interceptor

To install the Offline Interceptor on a secondary system:

1. Define the Connector QNAME to the ENQ manager.

   The Connector for RACF serializes access to its QUEUE dataset by issuing an ENQ with the SYSTEMS option.

   To allow synchronized updates of the QUEUE dataset from multiple systems, you may need to update your external ENQ manager (for example, GRS or MIM) with the Connector QNAME. This name must be propagated to all systems in the external ENQ manager complex.

   Another option is to let the Connector for RACF use the RNL=NO ENQ/DEQ parameter telling GRS to always propagate the ENQ to all systems in the GRS complex. This can be controlled using the ENQRNL parameter in member CTSPARM in Connector for RACF PARM library.

   The default QNAME used by Connector for RACF is xxxASYNC. The default may be customized by changing the QNAME parameter in member CTSPARM in Connector PARM library.

   For explanations on how to change and activate these parameters, see CTSPARM – Assembler Format Parameters.

2. Share the datasets.

   Place the datasets listed in the following table on a shared DASD between the primary and the secondary system.

Dataset name	Description
prefix.version.QUEUE	Connector QUEUE dataset
prefix.version.LOAD	Connector Load library
prefix.version.CMSG	Connector Msgs library
prefix.version.PARM	Connector PARM library
prefix.version.DIAGLVL	Connector Diagnostics library
prefix.version.ENCRINT	Connector for RACF Stored Data Encryption dataset

3. Copy CTSAOFI procedure and, if needed, STCJOB:

- If STCJOBS are not used, copy the CTSAOFI JCL procedure from the system PROCLIB of the primary system to the system PROLCIB of the secondary system.
- If STCJOBS are used:
  - Copy the CTSAOFI STCJOB member to the system STCJOBS library.
  - If the value LOCALCOPY is set for %PROLCIB% in the DEFPARMS member in the INSTALL library, make sure the secondary system has access to the Connector for RACF PROCLIB library. Otherwise, copy the CTSAOFI JCL procedure from the system PROCLIB of the primary system to the system PROCLIB of the secondary system.

4. In the secondary system(s):

   a. Add the Connector LOAD library to the APF authorized libraries list. For more information, see Step 15 – (Optional) Local Changes Migration.

   b. Set RACF and SMF logging options. For more information, see Set Logging Options.

   c. Install Offline Interceptor interface. For more information, see Install Offline Interceptor Interface.

RACF Remote Sharing Facility

The RACF Remote Sharing facility (referred to as RRSF) is a RACF feature that interconnects multiple RACF systems over SNA and APPC/MVS connections. The interconnection between RACF systems enables the site to synchronize RACF repositories between different z/OS images. The synchronization is achieved by RACF shipping the following over RRSF connections:

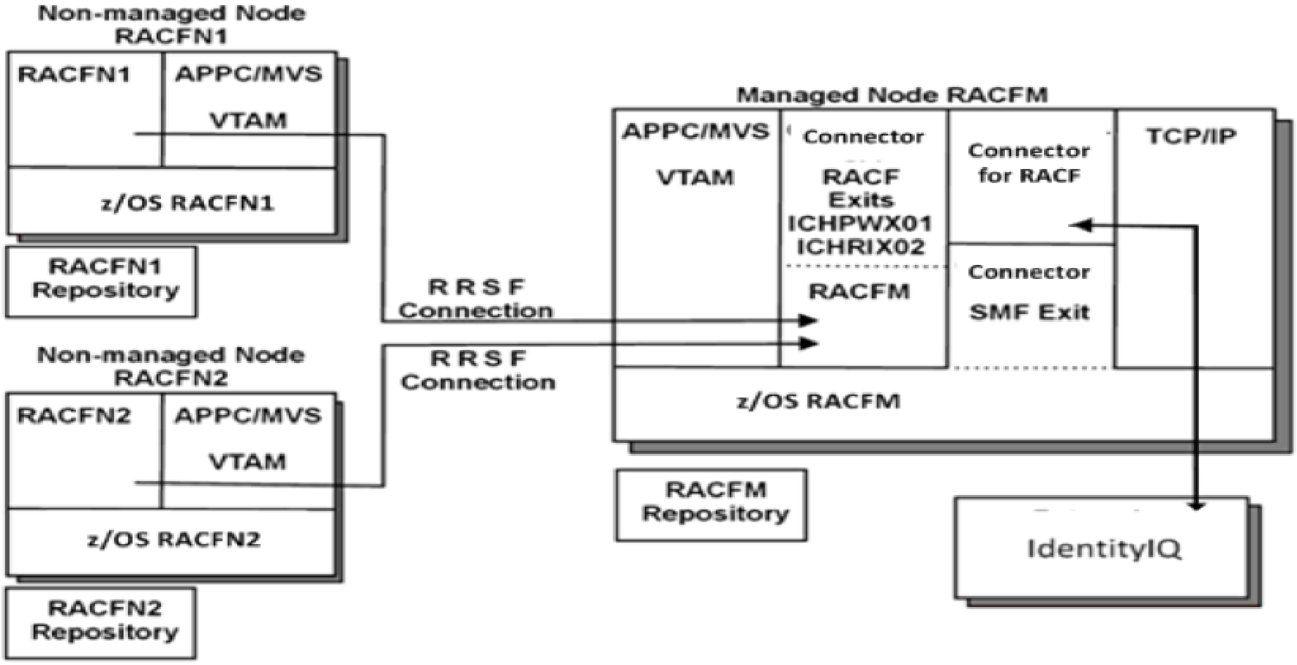
- RACF commands issued in one RACF system.
- RACF application updates performed in one RACF system.
- Password synchronization between predefined user associations between RACF systems.
- Automatic password synchronization for the same user on various RACF systems.

The following terms are used in the discussion that follows:

- **managed system (or node)** – a RACF system where full management by Connector for RACF is implemented and Connector for RACF communicates with IdentityIQ.
- **non-managed system (or node)** – a RACF system where full management by Connector for RACF is not implemented.

The following diagram depicts the setup of two RACF systems, RACFN1 and RACFN2. Each system is connected via RRSF to the third system RACFM. This setup is representative of many *non-managed* RACF systems (nodes), such as RACFN1 and RACFN2. These multiple non-managed RACF nodes have each an RRSF connection to one *managed* RACF node.

Direct RRSF connections in between non-managed RACF nodes are possible and most likely exist; however, such connections are not included in the scope of this guide.



In the previous example, some RACF events on non-managed nodes are synchronized with RACF on the managed node and are consequently intercepted by Connector for RACF on the managed node.

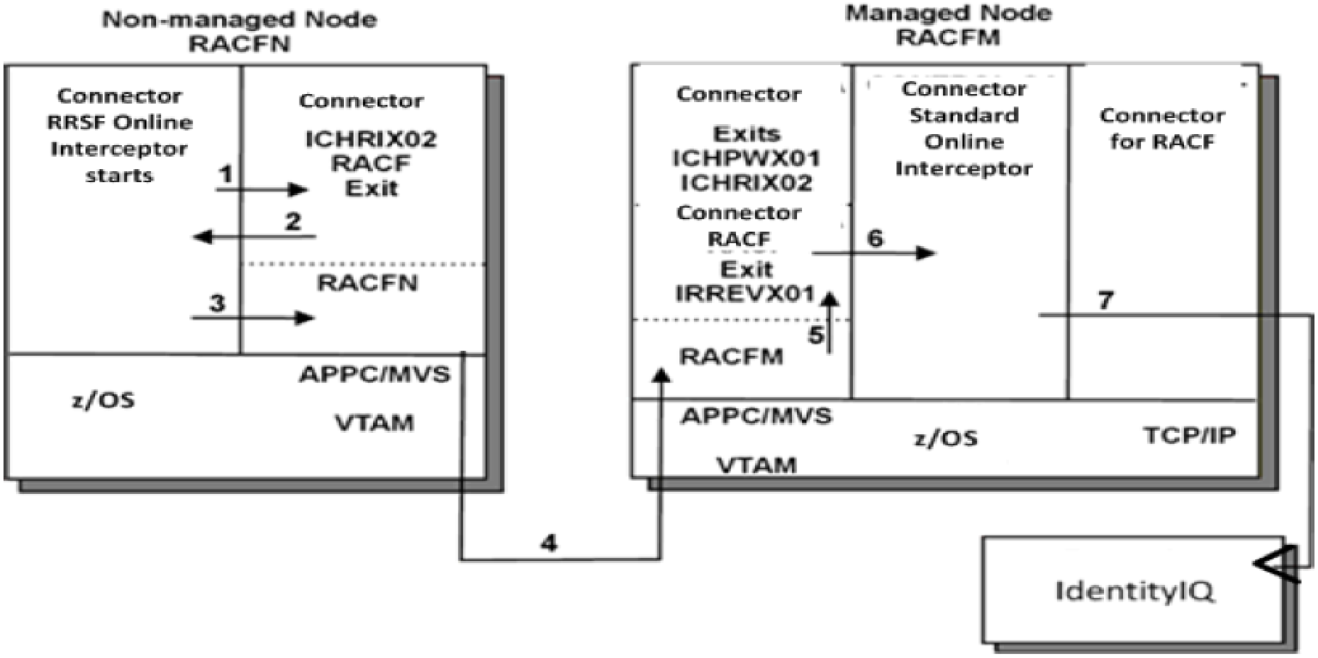
However, a problem exists with this setup, related to RRSF automatic password synchronization for a given user on various RACF systems. When a user initiates a password change on a non-managed system (called the *origin non-managed node*), RACF ships the password change event to the managed node (called the *target managed node*). This password change is applied to the RACF repository on the target managed-node. However, the password change does not drive the Connector for RACF exits which are part of Connector for RACF interception mechanism. As a result, the password change is not forwarded to IdentityIQ and the new password is not propagated to other systems managed by the Connector for RACF.

The Connector for RACF RRSF feature overcomes this problem and enables a single managed RACF node to intercept user-initiated password changes incoming over RRSF from multiple non-managed RACF nodes.

All other events automatically synchronized by RRSF drive native Connector for RACF interception exits on managed node; thus the RRSF feature for user-initiated password changes complements the ability of a single fully-operational Connector for RACF installation to manage an entire RRSF complex of multiple RACF systems.

Connector for RACF RRSF Support

The following figure depicts a non-managed node (RACFN) connected over RRSF to the managed node RACFM. Both the managed node and non-managed node contain the components of the Connector for RACF RRSF support feature supplied with Connector for RACF.



The following components and events are depicted:

1. In the non-managed node (RACFN), the Connector Online Interceptor is operated in a special mode, called RRSF mode. (The Connector Online Interceptor is run with special JCL and parameters.)

2. The Connector Online Interceptor running in the non-managed node (RACFN) starts Connector for RACF exit ICHRIX02 for RACF. ICHRIX02 handles user-initiated password changes and forwards them to the Online Interceptor which is operating in RRSF mode.

3. The Online Interceptor in the non-managed node encapsulates the password change event in a special RACF ALTUSER command that is directed only to (ONLYAT), the managed node (RACFM) over the existing RRSF connection between the two nodes.

4. The encapsulated RACF command is automatically protected by RRSF using CDMF masking, like all RRSF traffic between nodes. For more information, see Securing Connector RRSF Support.

5. On the managed node (RACFM), the Connector for RACF provides a new RACF command exit, IRREVX01. This exit traps the inbound RRSF encapsulating command that contains the password change event.

6. The IRREVX01 exit forwards the password change event to the Online Interceptor operating on the managed node (RACFM) using cross-memory transfer.

7. The Online Interceptor in the managed node treats the password event as if the event originated locally. Password change event interception proceeds as usual via the QUEUE dataset and the CD component. The event is sent to IdentityIQ and then automatically distributed to other Connector platforms.

In this manner, a user-initiated password change event proceeds:

- from the user on a non-managed RACF node
- via Connector for RACF on a managed node
- to IdentityIQ
- to all other MSCSs that participate in password synchronization

Customization and Operation of Connector for RACF to Implement Support for RRSF

Support for RRSF is implemented by performing customization on the managed system and on each non-managed system.

This section describes how to perform the required customization.

Customization and Operation of Connector for RACF on Non-Managed Nodes

Use the following procedure to customize each non-managed system to be handled via the managed system.

1. Ensure that the Connector for RACF operation libraries are available on the non-managed node. (Either copy the libraries to the node or place the libraries on a shared DASD.)

2. Review the RCFRRSF member located in the Connector PROCLIB library. The member contains the JCL used to start the local Online Interceptor in RRSF mode.

   Copy this member to the PROCLIB library of each non-managed system

   Associate the Connector Online Interceptor started task with a RACF user ID (local user).

   > **Note**
   > The Online Interceptor started task allocates a one-record temporary dummy QUEUE dataset that is opened, but never used.

3. An RRSF MANAGED user ID association must be defined between the RACF user ID associated with the non-managed node(s) Online Interceptor started task (local_user) and the RACF remote user on the RACF managed system.

   This association is performed using the RACLINK command and is required for command direction of the special RACF ALTUSER command that encapsulates the user-initiated password change event.

   The syntax of the RACLINK command is:

   ```
   RACLINK ID(local_user) DEFINE(target_node.remote_user) MANAGED
   ```

   where:

   - **local_user** – RACF local user ID associated with the Connector Online Interceptor started task
   - **target_node** – Target RRSF managed system
   - **remote_user** – RACF remote user on the target RRSF managed system

   For example:

   ```
   RACLINK ID(STCUSER) DEFINE(RACFM.CTSUSER) MANAGED
   ```

   This command defines a MANAGED user ID association between the local user STCUSER and user CTSUSER on the remote RRSF node RACFM. Such MANAGED association allows one-way RRSFcommand direction from local user STCUSER to remote user CTSUSER on the remote node RACFM.

4. Enable RRSF command direction by the RACF user ID associated with the Online Interceptor started task (local_user) to the target managed RACF node.

   This is achieved in RACF by creating a class RRSFDATA resource (for example, DIRECT.RACFM) and granting permission for this resource to the RACF local user.

   Specify the following commands to perform these actions:

   ```
   RDEF RRSFDATA resource_name UACC(NONE)
   PERMIT resource_name CLASS(RRSFDATA)ID(local_user)ACCESS(READ)
   SETROPTS RACLIST(RRSFDATA)REFRESH
   ```

   where:

- **resource_name** – Name of the class RRSFDATA resource.
- **local_user** – RACF user ID associated with the Online Interceptor started task in the non-managed node.

Example

```
RDEF RRSFDATA resource_name UACC(NONE)
PERMIT resource_name CLASS(RRSFDATA) ID(STCUSER) ACCESS(READ)
SETROPTS RACLIST(RRSFDATA) REFRESH
```

5. Review and edit member RRSFPARM in the PARM library. This member contains a subset of RSSPARM parameters, as well as parameters that are unique to the Connector Online Interceptor in RRSF mode. The following parameters are of particular importance. Parameters are listed with sample values:

```
RRSFONLI RSS_TYPE            RACF
RRSFONLI TARGET_NODE         RACFM
RRSFONLI TARGET_USER         CTSUSER
RRSFONLI DUMMY_USER          $#@$$@#$
RRSFONLI ONLI_DYNAM_PWX01    N
RRSFONLI ONLI_DYNAM_RIX02    Y
```

Each parameter is explained below:

- RSSNAME must be RRSFONLI (also specified in local node Online Interceptor JCL using PARM=).
- RSS_TYPE must be RACF.
- TARGET_NODE is the RRSF ID of the target managed node.
- TARGET_USER is a RACF user on the target managed node to which the encapsulating RACF command (ALTUSER) is directed. This is the user that was assigned a RACLINK association in step 3.
- DUMMY_USER is a string expressed using the syntax of a RACF user; however, this user is never defined in any RACF node. The dummy user name is used in the special RACF ALTUSER command that encapsulates the password change event sent from the non-managed node to the managed node.
- ONLI_DYNAM_PWX01 must be set to `N` on the non-managed node to prevent the activation of the Connector for RACF exit ICHPWX01, which traps local password change events performed by the RACF administrator via the ALTUSER or password command.

> **Caution**
> Do not perform a static installation of ICHPWX01.

- ONLI_DYNAM_RIX02 must be set to `Y` to enable Connector for RACF to dynamically load the ICHRIX02 exit, which traps user-initiated password change events.

  Alternatively, ICHRIX02 can be statically installed and loaded during operating system IPL. For more information, see **Method 2–Static Installation** in 3 – Install RACF Exits ICHRIX02 and ICHPWX01.

  > **Note**
  > The parameters TARGET_USER and DUMMY_USER must exactly correspond to the contents of the RCFRRSTB table, where RCFRRSTB is used by the IRREVX01 exit on the managed node to enforce security for incoming RACF commands that encapsulate password change events. For more information, see Customizing the RCFRRSTB Table.

6. Start the Connector Online Interceptor in RRSF mode on the non-managed node by specifying the following command:

   ```
   START prefixRRSF
   ```

   where the `prefix` is the chosen three character prefix for Connector JCL procedures.

Customization and Operation of Connector for RACF on the Managed Node

Use the following procedure to customize a managed node to support the RRSF feature.

1. Prepare the RCFRRSTB security table. The RCFRRSTB security table is a data-only module that is link-edited with the IRREVX01 exit (provided on the RRSF managed node). The RCFRRSTB security table provides the following safeguards:

   - Prevents the impersonation of the source of the special RACF command that encapsulates the password change event.

   - Ensures that the special RACF command that encapsulates the password change event is accepted and interpreted by the IRREVX01 only when the command is shipped from an authorized Connector Online Interceptor on a non-managed node.

   > **Note**
   > Prior to preparing the RCFRRSTB table, review Securing Connector RRSF Support.

   It is recommended that you prepare the RCFRRSTB in a test environment using the supplied sample source, prior to actual implementation. Initial sample source is supplied in member RCFRRSTB in the INSTALL library. This sample allows the IRREVX01 exit to accept the special RACF command (containing user-initiated password event) from any combination of origin node and user. You can assemble and link-edit the supplied RCFRRSTB table as supplied to perform a fast initial end-to-end test of the RRSF feature.

To use the supplied sample source, submit the job whose JCLs are in member ASMRRSTB in the Connector INSTALL library.

This job assembles and link-edits the supplied RCFRRSTB table into module CTSEVX01, which is loaded and operated as the IRREVX01 RACF exit.

2. Specify the following console command to define module CTSEVX01 in the MVS operating system as RACF exit IRREVX01:

```
EXIT ADD EXITNAME(IRREVX01) MODNAME(CTSEVX01) STATE(ACTIVE)
DSNAME(ctsa.load.library)
```

This command can optionally be embedded in the system PARMLIB(PROGnn) member and subsequently issued by specifying the following command:

```
SET PROG=nn
```

Specify the following console command if you wish to delete the CTSEVX01 module from the RACF exit:

```
EXIT DELETE EXITNAME(IRREVX01) MODNAME(CTSEVX01)
```

> **Note**
> When module CTSEVX01 is defined as RACF exit IRREVX01, it handles password change events that are sent over RRSF from authorized Connector Online Interceptors on non-managed nodes (running in RRSF mode). The sole purpose of CTSEVX01 is to handle user-initiated password change events by forwarding the password change events to the managed Connector Online Interceptor. CTSEVX01 ignores other events sent to it by RACF.

As with any exit defined using the MVS dynamic exit facility, CTSEVX01 can be operated alongside other modules which are ADDED to the same RACF IRREVX01 exit.

Verifying Successful Customization and Operation

After completing Customization and Operation of Connector for RACF to Implement Support for RRSF, perform the verification procedures.

1. Verify that the following relevant components are active:

On the non-managed node:

- The Connector Online Interceptor running in RRSF mode.
- ICHRIX02 RACF exit, supplied by Connector for RACF.

On the managed node:

- Connector for RACF including Connector Online Interceptor.
- IRREVX01 RACF exit, supplied by Connector for RACF.

2. Verify the functionality of enhanced support for RRSF:

> **Note**
> Where review of messages is required, an example of two RRSF-interconnected nodes in a test environment is provided.

On the non-managed node:

a. LOGON to TSO or to another application that allows user- initiated password changes. Change your password during LOGON.

b. Review Connector Online Interceptor log messages in DD statement STDMSG, and confirm that the following messages appear:

```
CTS1380I Connector Online Interceptor version 3.3.01 ID U23RRSF/STC28462
started
```

```
CTS1701I Online Interceptor ready for events
```

```
CTS4567I password change for User SKOLLER propagated to P3R8.NELLY
```

```
CTS4567I indicates that the password change performed during LOGON with user
SKOLLER was sent to target managed node P3R8, using target user NELLY.
```

On the managed node:

a. Review Connector Online Interceptor log messages in DD statement STDMSG, and confirm that the following messages correspond to the above password change on the origin non-managed node:

```
CTS4554I password of User SKOLLER has changed on MSCS NERRACF
```

```
CTS4552I User SKOLLER updated on MSCS NERRACF
```

> **Note**
> NERRACF is the managed node Managed System name. The password change is treated as if the event was intercepted locally.

b. Review Connector Notification Server (CD) log messages in DD statement STDMSG, and confirm that the following messages correspond to the above password change on the non-managed node:

```
CTS1557I NERRACF: Update User SKOLLER password event sent to Provisioning
Engine
```

```
CTS1553I NERRACF: Update User SKOLLER event sent to Provisioning Engine
```

3. The Standard Online Interceptor may write the following messages to DD statement STDMSG instead of messages CTS4554I and CTS4552I (see above):

   ```
   CTS4569I password change intercepted for user SKOLLER but verification failed
   ```

   ```
   CTS4570I Event ignored
   ```

   This failure prevents further propagation of events to the Notification Server (CD) and IdentityIQ, and may occur due to one of the following reasons (external to the RRSF support enhancement, as discussed in this chapter):

   a. Automatic RRSF password synchronization did not occur; therefore, the password value intercepted by the RRSF enhancement did not match the password value in the managed RACF node.

   b. The user SKOLLER is defined as PROTECTED in managed node RACF.

   c. The user SKOLLER is REVOKED in managed node RACF.

Securing Connector RRSF Support

The RRSF feature of Connector for RACF ships password changes between non-managed and Connector managed RACF systems. To secure this information, this feature fully addresses the sensitivity built-in flows of password changes.

The Connector RRSF feature uses RRSF infrastructure and RRSF command-direction to ship password change events among its components. By doing so, the Connector for RACF solution relies on:

- APPC/MVS security as deployed by the site between RRSF nodes
- RRSF built-in confidentiality, which is achieved by using masking inter-RRSF traffic using CDMF (Commercial Data Masking facility) algorithm and secret key provided by RACF. CDMF is a form of 40-bits DEA encryption, exportable outside the United States
- RACF confidentiality, such as non-displaying password values

Security Milestones in Connector RRSF Support

This section describes security milestones along the path of a password change event shipped between components of the Connector RRSF Support feature, as illustrated in the figure in Connector for RACF RRSF Support.

Milestones on the non-managed node:

- The ICHRIX02 exit traps the user-initiated password change event and sends the event to the local Connector Online Interceptor using the regular Connector for RACF secure cross-memory channel.

- The Connector Online Interceptor encapsulates the new password in the password keyword of the special ALTUSER command; therefore the password is not displayed in the RRSFLIST logs used by RRSF. An example of a log dataset name is STCUSER.RRSFLIST, where the Connector Online Interceptor runs under the RACF user STCUSER.

- The Connector Online Interceptor started task on the non-managed node is run under a RACF user that: - Must be defined (using the RCFRRSTB table) to the IRREVX01 component on the managed node - Must be authorized to direct RRSF commands to the target managed node

Milestones on the managed node:

- The CTSEVX01 exit prevents impersonation of the source of the RRSF-inbound RACF command by using the RCFRRSTB mechanism. For more information, see Customizing the RCFRRSTB Table.

- After interpreting the RRSF-inbound RACF command, CTSEVX01 forwards the password change event to local Connector Online Interceptor, using Connector built-in cross memory channels.

- As a RACF exit, CTSEVX01 fails (does not execute) the command that contains the password change event; therefore there is no SMF recording of the RACF command with password value.

- If module CTSEVX01 is not defined as RACF exit IRREVX01, the encapsulated ALTUSER command will be executed in the managed RACF system. This command will fail, as it refers to a dummy user name that does not exist in the managed RACF system. The failed command is not logged in SMF and the password is not exposed.

Customizing the RCFRRSTB Table

The RCFRRSTB table is an assembler source that consists of a sequence of CTSRRSTB macro calls.

- The first CTSRRSTB macro call is used to define the special dummy user name (described in step 5 in Customization and Operation of Connector for RACF on Non-Managed Nodes) used by to the ALTUSER command that encapsulates the password change event. This dummy user must not be defined in any RACF system. (The dummy user name provided in the sample is: $#@$$@#$)

  The first macro call in the RCFRRSTB source is:

  ```
  RCFRRSTB CTSRRSTB START=YES, CTSDUMY=dummy_username
  ```

  where `dummy_username` is a dummy user name (up to 8 characters).

  Label RCFRRSTB is mandatory and generates the external CSECT table name.

  > **Note**
  > This dummy user name must be specified in member RRSFPARM in the PARM library on

> all non-managed nodes as the value of parameter DUMMY_USER. See step 5 in Customization and Operation of Connector for RACF on Non-Managed Nodes.

- Subsequent CTSRRSTB macro calls are used to denote the valid combinations of the following:
  - Origin non-managed node
  - RACF user ID associated with the Online Interceptor started task on the non-managed node.
  - Local RACF user (on the target managed node) to which the encapsulating RACF command is directed.

> **Note**
> User ID association between the RACF user ID on the non-managed node and the RACF user on the managed node must be defined using the RACLINK command. For more information, see step 3 in Customization and Operation of Connector for RACF on Non-Managed Nodes.

The valid combinations of origin user, origin node, and local user are listed in RCFRRSTB assembler source as a sequence of entries, using the following syntax:

```
CTSRRSTB SRCUSER=source_user,SRCNODE=source_node,LCLUSER=local_user
```

where:

- `source_user` – name of a RACF user on the non-managed node (up to 8 characters)
- `source_node` – name of a non-managed node (up to 8 characters)
- `local_user` – name of a local RACF user on the managed node (up to 8 characters)

Each such entry instructs the CTSEVX01 module to accept user-initiated password change events from the Connector Online Interceptor that are:

- running under the specified RACF user (source_user)
- on the specified non-managed node (source_node)

These should be processed by the local exit running with the specified local_user ACEE (RACF user security block).

In each CTSRRSTB entry, the granularity of the security specification is determined by the fields SRCUSER, SRCNODE and LCLUSER.

You can lower and simplify the granularity by modifying the specification to one of the following:

- SRCUSER and SRCNODE. The field LCLUSER defaults to * (any LCLUSER)
- SRCUSER. The fields SRCNODE and LCLUSER defaults to * (any SRCNODE and LCLUSER)
- No field (the entry is specified as the CTSRRSTB macro call without operands). All three fields default to `*` (any SRCUSER, any SRCNODE and any LCLUSER).

> **Note**
> This entry is contained in the RCFRRSTB table in the provided sample. See the sample table provided below. If the entire RCFRRSTB table source contains entries with multiple granularities, only those entries with the highest granularity (greatest number of fields) are considered for matching by the CTSEVX01 module at run time.

Sample RCFRRSTB Table source

This section lists member RCFRRSTB in the INSTALL library that contains Connector for RACF. This is an example of a full RCFRRSTB specification, including a leading (header) CTSRRSTB macro call, one CTSRRSRB security entry, and a trailer (last) CTSRRSTB macro call.

```
*
        PRINT GEN
RCFRRSTB CTSRRSTB START=YES,CTSDUMY=$#@$$@#$
*>>ABOVE RCFRRSTB LABEL IS MANDATORY AND GENERATES CSECT NAME
        CTSRRSTB
*>>ABOVE CTSRRSTB ENTRY DEFAULTS TO FOLLOWING:
*>>        CTSRRSTB SRCUSER=*,SRCNODE=*,LCLUSER=*         ('*' DENOTES ANY)
        CTSRRSTB END=YES
        END

```

Activating Diagnostic Data in a Non-Managed System in RRSF Environments

This describes the required steps in order to be able to get diagnostic data for the RRSF Online Interceptor.

1. These special activities are required, since the RRSF Online Interceptor does NOT include a full Connector installation and the CTSDIAG utility cannot be run there.
2. In the managed system, with the full Connector environment, keep a backup of DIAGONLI member in the DIAGLVL lib.
3. In the managed system, with the full Connector environment, update DIAGONLI member in the PARM lib with the required diagnostic values and save member.

4. In the managed system, with the full Connector environment, edit CTSDIAG member in JCL lib and set M=DIAGONLI and submit.

   The job should end with `RC=0`.

5. In the non-managed system, keep a backup of DIAGONLI member in the DIAGLVL lib.

6. Copy the DIAGONLI member from the managed system, which is created in DIAGLVL lib, to the non-managed system (RRSF machine), where only the RRSF Online Interceptor exists.

   Configure it in the DIAGLVL lib there.

7. In the managed system, with the full Connector environment, restore the backup member of DIAGONLI which was kept during step 2 above.

8. In the non-managed system, restart the RRSF Online Interceptor.

   The diagnostic messages are written to the PRTDBG file.

Once diagnostic data is not required any more, the following steps should be done in order to stop Connector from writing diagnostic data to SYSOUT:

1. Copy the DIAGONLI member in DIAGLVL library from the managed system (where diagnostic flags are set to zeros), to the non-managed system (RRSF machine), where only the RRSF Online Interceptor exists.Set it in the DIAGLVL lib there.

2. In the non-managed system, restart the RRSF Online Interceptor. The diagnostic messages should not be written to the PRTDBG file any more.

Implement Support for Universal Groups

A Universal group is a group whose RACF profile only includes information for users with authority above AUTHORITY (USE). Connection information for users with AUTHORITY (USE) can therefore only be viewed within the respective RACF user profiles.

When the Universal group functionality is implemented in Connector for RACF, the following information appears in the Group and Managed System User–Group Connection Details windows respectively in IdentityIQ:

- An indication of whether a group is of type **Universal**

- An indication of whether a Managed System user is connected to a group of type **Universal**

To implement support for Universal groups, you must define a value for the RSSPARM parameter RCF_UNIGROUP_MAX. This parameter is used for storage allocation in processing the maximum number of connections to a Universal group.

To assist you in determining the appropriate value for the RCF_UNIGROUP_MAX parameter, a utility called CTSUCNT is provided. CTSUCNT scans the RACF database and outputs (WTO console message) the number of user connections to a specified sample group. It is recommended that you run CTSUCNT for a large sample universal group.

> **Note**
> CTSUCNT can be used to count the number of user connections in any type of group.

To implement support for universal groups:

1. (*Optional*) Run sample member ASMUCNT in the Connector INSTALL library to assemble and link CTSUCNT. Review the JCL carefully and submit the job to create the CTSUCNT load module in the Connector LOAD library.

   Use the output from the CTSUCNT utility to determine the value to assign to RSSPARM parameter RCF_UNIGROUP_MAX.

2. Stop the Connector for RACF Gateway and servers by specifying the following operator command:

   ```
   P CTSGATE
   ```

3. Edit the RSSPARM member in the Connector PARM library. Insert the following parameter:

   ```
   rss_name RCF_UNIGROUP_MAX value
   ```

   where value is the *value* determined in step 1 above. This must be a non-zero number up to 999999.

   > **Note**
   > The value of RCF_UNIGROUP_MAX must be greater than the number of connections to any Universal group.

4. Save the member.

5. Restart Connector for RACF.

Suppressing CTS1120E and CTS1121E Error Messages

Depending on certain environment considerations at your site, the following error messages may be generated routinely:

- Message CTS1120E appears when a user is already connected to a group in RACF, the connection is not recorded in the IdentityIQ database, and the IdentityIQ administrator tries connecting this user to the group.
- Message CTS1121E appears when the IdentityIQ administrator successfully deletes the connection of a user to a group.

You have the option of suppressing these messages so that they do not fill up Connector log files.

To suppress CTS1120E and CTS1121E error messages:

1. Add the following line to the RSSPARM member:

   ```
   rssName MSG_CONNGRP_ERR {Y|N}
   ```

   The variables in this line are as follows:

   - **rssName** – Name of the Managed System to which this parameter applies.
   - **Y** (default) – The CTS1120E and CTS1121E messages are generated as necessary.
   - **N** – The CTS1120E and CTS1121E messages are suppressed.

2. Restart the Connector.

   If the parameter is not present in the RSSPARM member, the functionality of the Connector remains unchanged.

Supporting Security zSecure Admin RESUME/REVOKE

Customers who use a Security zSecure Admin tool; such as Tivoli zSecure Admin, Consul zAdmin RACF, Consul/zAdmin RACF, or Consul/RACF may configure the Connector for RACF to intercept RESUME, REVOKE, and SCHEDULE DISABLE commands. The Connector for RACF intercepts these commands using IEFU84 SMF exit.

For instructions to enable this feature, see 2 – (Optional) Install SMF exit IEFU84.

Supporting Mixed Case Passwords

If mixed case is used in your RACF deployment, update the ONLI_PASSWORD_CASE parameter in the PARM library in the RSSPARM member to ASIS or UPPER instead of the default LOWER value. When mixed case is used in RACF and ASIS or UPPER is set in RSSPARM, the passwords are sent to IdentityIQ without any translation made by RACF Connector.

> **Note**
> ASIS and UPPER have same effect - which means no translation is done on password.

Custom Fields in the RACF Connector

Custom fields are fields within the RACF database that you customize to store security information about the users and groups in your system. You can tailor the names and attributes of custom fields. Once you define custom fields, use RACF commands, such as the ALTUSER and ALTGROUP to add data to custom fields.

For each custom field, you can customize the following attributes:

- The name of the custom field, which is used as the RACF command operand for TSO/E commands.
- The data type for the custom field. Choose character, numeric, hexadecimal, or flag (YES or NO) fields.
- The help text for each custom field.
- The output heading for LISTUSER and LISTGRP listings.
- The acceptable values for the data in each field based on data type. You can customize several options, including the following:
  - For character fields, you can customize maximum length, restrict the character contents, and allow mixed-case characters.
  - For numeric fields, you can customize maximum value and minimum value.
  - For hexadecimal fields, you can customize the maximum length.

Field attributes (names and data formats) defined as profiles in the new CFIELD general resource class. Field data for users and groups is held in a new CSDATA segments in USER and GROUP profiles.

The custom field names format is:

```
profile-type.segment-name.custom-field-name
```

where:

- **profile-type** – USER/GROUP
- **segment-name** – CSDATA
- **custom-field-name** – a name of up to 8 characters for this custom field

Custom fields are described in "Defining and using custom fields" chapter of IBM's *RACF Security Administrator Guide*.

Implement Custom Fields in Connector for RACF and SailPoint

Custom fields are supported when defined as schema attributes for the Account and Group entities. The custom fields data is passed along with the other Account or Group data when an Account or Group is synchronized, changed or added, or when aggregation is performed.

To implement support for Custom fields:

1. Stop the Connector for RACF Gateway and servers by specifying the following command:

   ```
   P CTSGATE
   ```

2. Edit the RSSPARM member in the Connector PARM library.

   Insert the following parameter:

   ```
   managedSystemName CUSTOM_FIELDS_SUPPORT Y
   ```

3. Save the member.

4. Restart the Connector for RACF Gateway and servers by specifying the following command:

   ```
   S CTSGATE
   ```

5. Add the Custom fields to the appropriate object type (account or group) in the SailPoint schema. The decision on the object type depends on the custom field type (USER or GROUP). The field name must be as follows:

   ```
   CSDATA.<custom-field-name>
   ```

6. Perform full account aggregation and group aggregation to get the custom fields data.

CTSACF Features

> **Note**
> CTSACF features can be supported with RACF V1.10 or higher.

CTSACF features are Connector features controlled by RACF custom fields. Each feature has an associated custom field default name, and an associated RSSPARM parameter that can be used to override the custom field default name.

A feature is active when the associated custom field exists in RACF and is defined with the required attributes. A feature is inactive when the associated custom field is not defined or is defined with attributes other than that is required.

The custom field associated with each feature is used for:

- Activating the feature
- Saving data at user or group level (depending on the feature)

Implementation of CTSACF features does not require implementation of Custom Fields support in the Connector for RACF. If Custom Fields support is implemented, it supports only site custom fields. The features associated custom fields are not defined as keywords to SailPoint and are not displayed in Site Specific tab in

Account or Group entity.

> **Caution**
> When CTSACF features are activated, care should be taken not to use the NOCSDATA parameter in ALTUSER or ALTGROUP RACF commands. This parameter will remove all the custom fields from the user or group record, including the CTSACF features data. There is no problem in setting the NOCSDATA checkbox in SailPoint since the Connector will do the needed actions to remove the site custom fields only without affecting the CTSACF features data.

The Connector for RACF supports the following CTSACF feature:

Feature Name	Level	Default custom field name	RSSPARM Parameter
Locked Account Support	User	CTSLKACT	LOCKED_ACCOUNT_CFNAME

Supporting the Locked Accounts Feature

> **Note**
> Supporting the Locked Accounts feature is supported only when Online Interceptor is used. In order to view the revoke_reason, specific RACF custom fields must be defined.

Connector for RACF can now be configured to support the Locked Accounts feature that was implemented in SailPoint.

The Locked Accounts feature enables SailPoint to distinguish between accounts that are revoked by an administrator and accounts that are locked due to excessive password attempts.

Handling Locked Accounts in SailPoint

When the Locked Accounts feature is implemented, SailPoint differentiates between accounts revoked because of excessive password attempts and accounts revoked because of other reasons.

- An account revoked because of excessive password attempts is considered **locked**.
- An account revoked because of another reason is considered **suspended**.

SailPoint receives the locked and suspended indications from the Connector, in the **RU_LOCKED** and **RU_SUSPENDED** keywords.

In addition, SailPoint receives from Connector the specific revoke reason. This revoke reason is received in **RACF_REVOKE_REASON** attribute. Possible values for this attribute are as follows:

- **PASSWORD** – The account is revoked due to excessive password or phrase attempts.

  This value is returned when RU_LOCKED is Y.

- **COMMAND** – The account is revoked because of an administrator command.

  This value can be returned when RU_SUSPENDED is Y.

- **INACTIVITY** – The account is revoked because of inactivity period.

  This value can be returned when RU_SUSPENDED is Y.

- **DATE** – The account is revoked because of revoke date.

  This value can be returned when RU_SUSPENDED is Y.

- **UNKNOWN** – Revoke reason is unknown.

In IdentityIQ versions **prior to 6.3** these operations must be done manually in IdentityIQ to add support in this feature:

1. Add the following attributes to Application schema:

   - RU_LOCKED

   - RU_SUSPENDED

   - RACF_REVOKE_REASON

2. In debug page in RACF Application, add to the **featuresString** the value **UNLOCK**.

3. In same debug page add the following:

```
<entry key="splAccountAttributes">
    <value>
        <Map>
            <entry key="RU_LOCKED" value="false"/>
            <entry key="RU_SUSPENDED" value="false"/>
        </Map>
</value>
```

Locked Accounts in the Connector

> **Important**
> Locked Accounts feature requires Online Interceptor to be running consecutively.

Locked Accounts is a CTSACF Feature:

Feature Name	Level	Default custom field name	RSSPARM Parameter
Locked Account Support	User	CTSLKACT	LOCKED_ACCOUNT_CFNAME

In order for the Connector for RACF to return the appropriate values in the **RU-LOCKED, RU_SUSPENDED** and **RACF_REVOKE_REASON** keywords, the connector has to know:

- The account status – revoked or resumed
- The revoke reason for revoked accounts

Users in RACF can be revoked for several reasons:

- Excessive password attempts
- Inactivity period
- ALTUSER command with the REVOKE parameter
- Revoke date

Users in RACF can be resumed for two reasons:

- ALTUSER command with the RESUME parameter
- Resume date

When a user becomes revoked, RACF does not save the reason in the user record. The revoke reason can be determined by analyzing the SMF record written by RACF when revoking a user (except for revoke because of revoke date). An SMF record is also written when a user is resumed because of an **ALUSR RESUME** command.

The RACF SMF records are seen by the Connector Online or Offline interceptors. In order to make the revoke reason available when the Connector retrieves the user data, the interceptors analyze the SMF records and determine the user status. For revoked users, they determine the revoke reason and keep it in **CTSLKACT** custom field in the user record. For resumed users, they set a special revoke reason indicating that resume was done, for further processing by the Connector server. Now the revoke reason is available for the Connector server when having to determine the account status and whether it is Locked or Suspended.

When a user is revoked or resumed as a result of a request from SailPoint, the Connector server knows the revoke reason and can set it in **CTSLKACT** custom field.

Revoke reasons set in **CTSLKACT** custom field:

- P - Excessive password attempts.
- I - Inactivity period
- C - ALTUSER command with the REVOKE parameter
- D - Revoke date
- ? - Unknown. The user was resumed by a command but it has to be checked if it is actually resumed, or revoked because of revoke date.

The **CTSLKACT** custom field does not exist in the user record when:

- The user is active.
- The user is revoked but the revoke reason is not known. This will be the case after initial activation of the feature.

When a user is retrieved by the Connector, it determines the user status and revoke reason according to the user data. There are some special cases that need special attention from the Connector server:

- When the user is revoked by date and has a revoke reason of '?', it will set the appropriate revoke reason in the **CTSLKACT** custom field.
- When the user is revoked with no revoke reason, the Connector will check the revoke date to determine if the user is revoked because of date. If it is, it will set the appropriate value in **CTSLKACT** custom field.
- When the user is active (was resumed because of resume date) but still has a revoke reason, it will remove the revoke reason by removing **CTSLKACT** custom field from the user record.

When the Connector interceptors or the Connector server set a revoke reason in **CTSLKACT** custom field, a message is issued to **STDMSG** file indicating that the user record was changed.

To implement support for Locked accounts:

1. Stop the Connector for RACF Gateway and servers by specifying the following command:

   ```
   P CTSGATE
   ```

2. Stop the Online interceptor, if active, by specifying the following command:

   ```
   P CTASONI
   ```

3. The default name of the custom field used for revoke reason is USER.CSDATA.CTSLKACT.

   If you wish to change this name:

a. Edit member RSSPARM in the Connector PARM library.

b. Insert the following parameter:

```
rss_name LOCKED_ACCOUNT_CFNAME <-custom-field-name >
```

Where `<custom-field-name>` is the last qualifier (after USER.CSDATA) of the custom field to be used for revoke reason.

c. Save the member.

4. Define the custom field to be used for revoke reason to RACF.

   The custom field should be defined with the following attributes:

   - TYPE(CHAR)
   - MAXLENGTH(1)
   - FIRST(ANY)
   - OTHER(ANY)
   - MIXED(NO)

   Custom Fields are described in "Defining and using custom fields" chapter in IBM's *RACF Security Administrator Guide*.

5. Restart the Connector for RACF Gateway and servers by specifying the following command:

```
S CTSGATE
```

6. Restart the Online Interceptor, if needed, by specifying the following command:

```
S CTASONI
```

7. Aggregate all Accounts in order to set revoke reason for users revoked because of revoke date. The initial revoke reason for users revoked for other reasons will be empty since the connector cannot determine the revoke reason.

From this point on, the revoke reason will be updated by the Connector components according to changes in users status.

AUTOPROF SMF Records

A USER or GROUP is sometimes updated automatically by RACF. For example, when FTP login is done and an OMVS segment is added to USER profile together with a proper UID. When such update occurs, an AUTOPROF SMF record is written (Event Code 88).

In order to intercept this event by Online Interceptor, IEFU84 must be used.

For more information about installation of SMF exit IEFU84, see 2 – (Optional) Install SMF exit IEFU84.

TSO LOGON Events

TSO logon events are written in SMF record type 30. Therefore, relevant SMFPRMxx member must include reference to TYPE 30 SMF record.

If **SMFPRMxx** is updated for this purpose it needs to be activated using `SET SMF=`*`xx`* command (where *`xx`* is the suffix of **SMFPRMxx** member that is updated).

For all other steps required to implement the TSO LOGON events, see 2 – (Optional) Install SMF exit IEFU84.

If TSO LOGON events are not required and must not be intercepted but IEFU84 exit is in use, parameter LOGIN_INTERCEPT may be set to **N** in RSSPARM.

> **Note**
> Using TSO LOGON events may increase dramatically the work done by Connector for RACF.
> Each user logon to TSO will cause an event to be written by Online Interceptor and a GetUser operation done by the Notification Server to read the changes from RACF DB and send updated user record to IdentityIQ to update IdentityIQ database.

Secured Communication

Secured communication has the following aspects:

- Communication security using TLS or using Transmitted Data Encryption
- IP List validation - which allows control of the IP addresses which are allow accessing the RACF Connector

The following topics are discussed in this chapter:

TLS Secured Communication

For using TLS secured communication for RACF Connector, TLS communication must be defined and configured in the following relevant components:

- TCP/IP in the system where RACF Server will be active
- Connector Gateway
- IdentityIQ or IdentityNow

System Requirements

- The following respective components for z/OS versions must be installed for TLS communication:

z/OS Version	Cryptographic Services	z/OS Security Level 3
z/OS 2.5	System SSL Base: FMID HCPT450	System SSL Security Level: FMID JCPT451
z/OS 2.4	System SSL Base: FMID HCPT440	System SSL Security Level: FMID JCPT441

- The CSF started task must be active in the system where RACF Server would be active.

Implementing Secured Communication for RACF Connector

Secured communication to RACF Connector must be implemented using **AT-TLS policy.** The TLS processing is done by TCP/IP and is transparent to the RACF Connector.

The secured communication is implemented using server authentication.

Implementation Procedure

To set up secure communication for the RACF Connector:

1. A valid server certificate with its associated server private key must be defined. This certificate must be signed by a trusted Certificate Authority's (CA).

   > **Note**
   > For testing purposes, a local CA can be defined for signing the server certificate.

2. The server certificate and the CA certificate must be connected to a key ring.

3. The CA certificate must be exported to a file, transferred (using FTP with ASCII mode) to the client (with `.cer` suffix) and installed there to be used for certificate verification by the TLS handshake process.

4. Implement an AT-TLS policy for RACF Connector communication.

   > **Note**
   > For detailed information about implementing AT-TLS policy, see "Application Transparent Transport Layer Security data protection" chapter of *z/OS Communications Server IP Configuration Guide*.

   The required policy attributes for AT-TLS policy are:

   - Local Port Range – ports defined in ECAPARM for RACF Connector
   - Direction – Inbound
   - TLS Enabled – On
   - TLS v1.1 – On
   - TLS v1.2 – On
   - Handshake Role – Server
   - Client Authorization Type – PassThru
   - Application Controlled – Off
   - Secondary Map – Off

- The name of the certificate created for the secured communication and the name of the key ring to which the server certificate and the CA certificate are connected, should be specified.

> **Note**
> TCP/IP must be granted permission to access the key ring to which the RACF server certificate and the CA certificate are connected.

Sample File for AT-TLS Policy

```
# RULE for RACF Connector CTSGATE
####################################################
TTLSRule CTSGATE
{
LocalAddr ALL
RemoteAddr ALL
LocalPortRange 2470-2471
Direction Inbound
Priority 255 # highest priority rule
Userid CTSGATE
TTLSGroupActionRef GrpAct_CTSGATE
TTLSEnvironmentActionRef GrpEnv_CTSGATE
TTLSConnectionActionRef GrpCon_CTSGATE
}
TTLSGroupAction GrpAct_CTSGATE
{
TTLSEnabled On
Trace 7
}
TTLSEnvironmentAction GrpEnv_CTSGATE
{
Trace 7
HandshakeRole Server
EnvironmentUserInstance 0
TTLSKeyringParmsRef PrmKeyRing_CTSGATE
TTLSEnvironmentAdvancedParmsRef PrmEnvAdv_CTSGATE
}
TTLSEnvironmentAdvancedParms PrmEnvAdv_CTSGATE
{
TLSv1.1 On
TLSv1.2 On
ClientAuthType PassThru
}
TTLSConnectionAction GrpCon_CTSGATE
{
HandshakeRole Server
TTLSCipherParmsRef PrmCipher_CTSGATE
TTLSConnectionAdvancedParmsRef PrmConAdv_CTSGATE
CtraceClearText Off
Trace 7
}
TTLSConnectionAdvancedParms PrmConAdv_CTSGATE
```

```
{
ApplicationControlled Off
CertificateLabel CTSGATE
SecondaryMap Off
}
TTLSCipherParms PrmCipher_CTSGATE
{
# supported cipher suites - we used a wide list, that should be decreased according
to specific needs
V3CipherSuites TLS_DH_DSS_WITH_DES_CBC_SHA
V3CipherSuites TLS_DH_RSA_WITH_DES_CBC_SHA
V3CipherSuites TLS_NULL_WITH_NULL_NULL
V3CipherSuites TLS_RSA_WITH_NULL_MD5
V3CipherSuites TLS_RSA_WITH_NULL_SHA
V3CipherSuites TLS_RSA_EXPORT_WITH_RC4_40_MD5
V3CipherSuites TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
V3CipherSuites TLS_RSA_WITH_DES_CBC_SHA
V3CipherSuites TLS_DHE_DSS_WITH_DES_CBC_SHA
V3CipherSuites TLS_DHE_RSA_WITH_DES_CBC_SHA
V3CipherSuites TLS_RSA_WITH_AES_256_CBC_SHA256
V3CipherSuites TLS_RSA_WITH_AES_256_CBC_SHA
V3CipherSuites TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
V3CipherSuites TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_DHE_RSA_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_DHE_DSS_WITH_AES_128_CBC_SHA
V3CipherSuites TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
V3CipherSuites TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
V3CipherSuites TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
}
TTLSKeyringParms PrmKeyRing_CTSGATE
{
Keyring CTSRING
}
```

Enabling TLS Between RACF Connector and Connector Gateway

For more information on the procedure to be performed on the client side (Connector Gateway), see *SailPoint Integration Guide* or *SailPoint Quick Reference Guide for Gateway Connectors* depending on the Connector Gateway release.

> **Note**
> Ensure that the Transmitted Data Encryption is not active for communication between the RACF connector and the Connector Gateway.

Enabling TLS Between SailPoint and Connector Gateway

For more information on the procedure to be performed on SailPoint, refer to the relevant documentation.

> **Note**
> Ensure that the Transmitted Data Encryption is not active for communication between the Connector Gateway and IdentityIQ.

Transmitted Data Encryption

Transmitted Data Encryption must be configured in the RACF Connector and in IdentityIQ.

> **Note**
> Transmitted Data Encryption is not supported for communication with IdentityNow.

Implement Transmitted Data Encryption for RACF Connector

Enable transmitted data encryption, then install the transmitted data encryption key dataset.

The Connector for RACF uses keys in the Encryption Key dataset to encrypt data which is transmitted between Connector for RACF and IdentityIQ. The Encryption Key file (global or platform-specific) which was generated in IdentityIQ must be copied to the Encryption Key dataset in Connector for RACF.

Enable Transmitted Data Encryption

Edit member CTSPUSR in the Connector PARM library.

Install Transmitted Data Encryption Key Dataset

To copy the Encryption Key file, use a file transfer utility available at your site or any other available transfer method. The Encryption Key file is a text dataset. Therefore, the transfer method must convert ASCII to EBCDIC and must support conversion of line breaks.

The Transmitted Data Encryption keys are stored in the following dataset:

```
<prefix>.<version>.ENCREXT
```

where:

Parameter	Description
<prefix>	Value set for the OLPREFS parameter in Connector for RACF Datasets Allocation Parameters.
<version>	Value set for the OLVERS parameter in Connector for RACF Datasets Allocation Parameters.

Implement Transmitted Data Encryption for IdentityIQ

To use Transmitted Data Encryption for IdentityIQ, an Encryption Key file must be created and transferred to the RACF Connector, and the Transmitted Data Encryption must be enabled.

For detailed implementation, see *SailPoint Quick Reference Guide for Gateway Connectors* or *SailPoint Integration Guide*.

Supporting Incoming IP Address Validation

To enable incoming IP address validation, perform the following:

1. In the ECAPARM member of the PARM library, add the following statement to the CHANNEL definition:

   ```
   IPLIST=ECAIPLSx
   ```

2. A corresponding ECAIPLSx member should be set in the library pointed by DAPARM DD card of the CTSGATE STC.

3. Member ECAIPLSx should contain the relevant Allow and Forbid statements as described in the full feature description that follows.

Configuring Incoming IP Address Validation

Syntax of IP Addresses List Syntax

The list of IP addresses is a source table that contains IP addresses of work stations allowed to communicate with CTSGATE that owns this list. An IP address is specified in the standard dotted-decimal notation. From 1 through 4 contiguous from right-to-left sections of an IP address can be specified with an asterisk. Such asterisk in a section of IP address which, actually defines a range of IP address. For example: `80.56.241.*` means in fact, a range of IP addresses from `80.56.241.0` through `80.56.241.255`.

If the above described simplified method of specifying an addresses range turns out too rough to determine a necessary range, then a standard subnet mask can be also used.

Any separate IP address or a range of addresses that should be allowed for communication, must be preceded with the keyword **ALLOW ***. IP address or a range of addresses that should be prevented from communication, must be preceded with the keyword **FORBID ***.

> **Note**
> The first statement of the list must be always [ALLOW *] or [FORBID *].

An incoming address will be checked against the table of IP addresses (in the internal format) to determine if it is to be confirmed or rejected. A given address may in principle match multiple (and even conflicting) entries in the table.

Only one entry will be used to determine whether the IP address is to be allowed or be forbidden. The entry with the most specific address is the effective entry.

The examples provided below demonstrate how a list of IP addresses can be coded:

- The following example allows any IP address, but excluding (forbidding) one specific address (`81.50.1.241`) and a range of addresses (`80.56.241.0` – `80.56.241.255`):

```
ALLOW * First statement allows any IP
Forbid 81.50.1.241
Forbid 80.56.241.*
```

- The same example may be specified with a subnet mask. For example:

```
ALLOW * First statement allows any IP
Forbid 81.50.1.241
Forbid 80.56.241.0, MASK=255.255.255.0
```

- The following example forbids all IP addresses, allowing only two specific addresses (`172.16.241.128` and `81.50.1.241`) and a range of addresses (`80.56.241.0` – `80.56.241.255`):

```
FORBID * First statement forbids any IP
ALLOW 172.16.241.128
ALLOW 81.50.1.241
ALLOW 80.56.241.*
```

- The following example defines 2 short ranges (`172.16.241.0` – `172.16.241.127`) and (`172.16.241.129` – `172.16.241.255`) of addresses that are allowed:

```
FORBID *
ALLOW 172.16.241.*
FORBID 172.16.241.128
```

- The following example shows usage of a subnet mask to allow a range of addresses (`172.16.240.0` – `172.16.255.255`):

  ```
  FORBID *
  ALLOW 172.16.240.0, MASK=255.255.240.000
  ```

- The following is an example of an ALLOW type IP list:

  ```
  FORBID *
  ALLOW 172.16.130.151
  ALLOW 172.16.110.*
  ALLOW 172.16.241.*
  ALLOW 80.56.1.*
  ```

There is no limit on the number of entries in a list of IP addresses. The order of ALLOW and FORBID statements is not important.

Source format of an IP List is processed at the initialization of CTSGATE (or when the REFRESH modify command is issued as shown below) to detect syntax errors.

Both [ALLOW *] and [FORBID *] statements create the following mask: `0.0.0.0`. The absolute value of a mask as a hexadecimal number defines the degree of specificity.

Location of IP Addresses List

The list of IP addresses must reside in a library allocated by a DAPARM DD statement under ECAIPLS fixed name. A one-character suffix is supported in a member name; for example: ECAIPLSx

List name of IP Addresses List

The name of the required IP list should be specified by the new Channel parameter of CTSGATE as follows:

```
IPLIST=ECAIPLSx
```

The presence of ECAIPLS source member in a library allocated by a DAPARM DD statement is required as soon as the IPLIST channel parameter is specified in ECAPARM.

Modifying the IP Addresses List

ECAIPLS source member is available to a user for changes. ECAIPLSx source member can be refreshed dynamically by means of the modify command, `REFRESH=ECAIPLSx`, without need to restart CTSGATE.

Administration of IP Addresses Validation

- **Operations** – To refresh ECAIPLSx source member dynamically, the following modify command should be issued:

  ```
  F <CTSGATE>, REFRESH=ECAIPLSx
  ```

  > **Note**
  > The IPLIST only blocks establishing connections. Refreshing the IPLIST does not affect existing connections.

- **Administrative features** – IP address validation becomes available as soon as:
  - Proper PTF has been applied.
  - A list of IP addresses resides in the library allocated by a DAPARM statement
  - The PLIST channel parameter is specified in ECAPARM.

    TRACE=199 should be set on in order to track processing of the IP list.

- **Security requirements** – The feature is not mandatory. To enable the feature, the IPLIST=ECAIPLSx channel parameter should be specified in the ECAPARM parameters member. When the feature is enabled, a list of IP addresses must exist in a library allocated by a DAPARM statement.

- **Internal diagnostics** – TRACE=199 should be set to ON in order to track processing of the IP list. The ECAIPLSx member will be printed in DAPRENV.

  If the feature is enabled, information about specific IP list will be printed in DAIGLOG output.

  If the CTSGATE channel is disabled due to invalid ECAIPLSx, the detected invalid lines in ECAIPLS will be displayed.

BIND

IP address that IOAGATE must use to listen for incoming connections. If you want IOAGATE to listen on a specific IP address, such as a DVIPA assigned for IOAGATE, use this parameter to identify that IP address.

Use the following syntax:

```
BIND=INADDR_ANY | IP_address | hostname
```

where:

- `INADDR_ANY` instructs IOAGATE to listen for incoming connections from any IP address (adapter) on the system.

- `IP_address` or `hostname` indicates that IOAGATE BINDs to either the given IP_address or the IP_address after hostname resolution.
- Default: `INADDR_ANY`

Operations

The Connector for RACF does not include a local user interface since almost all the functions it performs are activated via SailPoint. However, certain Connector for RACF operations (generally used for maintenance) are available. These operations are described in this chapter.

The following topics are discussed in this chapter:

Starting the Connector for RACF

To start the Connector for RACF:

1. Issue the following operator command:

```
S CTSGATE
```

The Connector for RACF Gateway starts and, in turn, automatically starts the Connector servers: Transaction Server (s) and Notification Server. As each of these components of Connector for RACF is successfully started, an appropriate message is displayed.

- If the Connector Gateway is active when you start the Connector for RACF, communication is automatically established between the gateways.

- If the Connector Gateway is not active when you start the Connector for RACF, the Connector for RACF Gateway waits. When the Connector Gateway is started, it establishes communication with the Connector for RACF Gateway.

> **Note**
> If the Connector for RACF will be started using an automatic startup tool during system ini-

> ialization, the Connector for RACF must be started after TCP/IP and the managed RACF subsystem have been started.

Shutting Down the Connector for RACF

To shut down the Connector for RACF:

1. Issue the following operator command:

   ```
   P CTSGATE
   ```

The Connector for RACF Gateway shuts down the Connector Transaction Server and Notification Server, and then shuts itself down.

Starting and Stopping the Online Interceptor

The Connector Online Interceptor is started independently of the Connector for RACF Gateway and the Connector Transaction Server and Notification Server.

> **Note**
> It is recommended that you keep the Online Interceptor active at all times (even if Connector for RACF Gateway is down), to ensure that all RACF changes are recorded.

Configuring the automated startup of Online Interceptor is described in procedure Step 9 – Customize Communication Settings.

Start the Online Interceptor

1. Issue the following operator command:

   ```
   S CTSAONI
   ```

Shut Down the Online Interceptor

1. Issue the following operator command:

   ```
   P CTSAONI
   ```

> **Note**
>
> If two or more instances of Connector for RACF on the same computer use the dynamic installation of the new password exit, restrictions apply to the order in which the Online Interceptors for each instance of Connector for RACF are shut down. The Online Interceptors must be shut down in the reverse order to which they were started.

> For example, if the Online Interceptor for Connector instance **A** was started, followed by the Online Interceptor for **B** and then for **C**, the Online Interceptors should be shut down in the order **C**, then **B**, and finally **A**.If two or more instances of Connector for RACF on the same computer use the dynamic installation of the new password exit, restrictions apply to the order in which the Online Interceptors for each instance of Connector for RACF are shut down. The Online Interceptors should be shut down in the reverse order to which they were started.

Starting the Offline Interceptor Manually

To start the Offline Interceptor manually:

1. Issue the following operator command to operate the Offline Interceptor as started task (the procedure is located in the system PROCLIB library):

   ```
   S CTSAOFI
   ```

   The task terminates upon processing of the incremental SMF input.

Scheduling the Offline Interceptor

The Offline Interceptor can be scheduled independently of Connector for RACF, either manually or automatically via the Notification Server.

The following options exist for scheduling activation of the Offline Interceptor:

- The Notification Server can schedule the Offline Interceptor to run at fixed intervals, starting from the time Connector for RACF is activated. The interval is specified using parameter OFLI_INTERVAL, described below.

- The Notification Server can schedule the Offline Interceptor to run at specific times during the day. The run times are specified using parameter OFLI_RUN_TIME_LIST, described below.

- You can schedule the Offline Interceptor using an external facility.

Configuring Offline Interceptor Parameters

Offline Interceptor scheduling is controlled using parameters in file RSSPARM. This file contains both common and Managed System-specific parameters of the different managed system by Connector for RACF. Unless indicated otherwise, the parameters (as mentioned below) that control Offline Interceptor scheduling are specified for each Managed System:

- OFLI_INTERCEPT

- OFLI_WAIT_INTERVAL

- OFLI_INTERVAL
- OFLI_RUN_TIME_LIST
- OFLI_RUN_INTERVAL

OFLI_INTERCEPT

Offline Interceptor scheduler activation. Possible values are:

- **Y:** The Offline Interceptor is started periodically by the Notification Server based on the value of parameter OFLI_INTERVAL or OFLI_RUN_TIME_LIST (described below). Default.
- **N:** The Offline Interceptor is not started by the Notification server. You must provide another means of scheduling the Offline Interceptor.

For information on starting the Offline Interceptor manually, see Starting the Offline Interceptor Manually.

OFLI_WAIT_INTERVAL

An interval, starting from the time of the initial communication between Connector for RACF and SailPoint, during which the Notification Server will not activate the Offline Interceptor, even if it is scheduled to run during that time. This provides a period at Connector for RACF startup during which the administrator can perform functions in Connector for RACF without interference from the Offline Interceptor. Format: **hhmmss**. Default: 001000 (10 minutes).

> **Note**
> Unlike the other Offline Interceptor scheduling parameters, OFLI_WAIT_INTERVAL applies to all managed system by the Connector for RACF installation.

OFLI_INTERVAL

The interval that must pass between consecutive activations of the Offline Interceptor (format: **hhmmss**). The Offline Interceptor is first activated at the earliest opportunity following the activation of Connector for RACF, and then after the specified interval. If Connector for RACF is stopped and restarted, the Offline Interceptor is not activated until the specified interval has passed from the previous activation. Default: 010000 (1 hour).

OFLI_RUN_TIME_LIST

A list of times at which the Offline Interceptor should be activated. The times are stated in 24-hour format **hh:mm**, separated by commas. 00:00 represents midnight.

For example:

```
00:00,03:00,09:30,12:00,15:30,21:25
```

The Offline Interceptor will be activated at the times specified in the list. If, for any reason (for example, Connector for RACF was not active), the Offline Interceptor missed a scheduled activation, it will not be activated until the next time specified in the list. See also OFLI_RUN_INTERVAL parameter.

If OFLI_RUN_TIME_LIST and OFLI_INTERVAL parameters are specified, Connector for RACF ignores OFLI_RUN_TIME_LIST parameter.

If neither of the parameters are specified, Connector for RACF assigns the default value **02:30** to the OFLI_RUN_TIME_LIST parameter.

OFLI_RUN_INTERVAL

The maximum deviation from scheduled activation time (in parameter OFLI_RUN_TIME_LIST) that the Notification Server will tolerate for activation of the Offline Interceptor. If the Offline Interceptor cannot be activated within this interval from the scheduled time (for example, Connector for RACF was not active), the scheduled activation is skipped. Format: **hhmmss**. Default: 001000 (10 minutes).

> **Note**
> Parameters OFLI_INTERVAL, OFLI_WAIT_INTERVAL and OFLI_RUN_INTERVAL must be expressed as valid time representations in the format **hhmmss**, where **hh** is 00 through 23, **mm** is 00 through 59, and **ss** is 00 through 59. For example: 010000 translates to 01:00:00 (1 hour) and is a valid value; 006000 translates to 00:60:00 which is not a valid time and thus will be ignored.

Example:

Given the following parameter values:

- OFLI_RUN_TIME_LIST: 00:00,12:00
- OFLI_RUN_INTERVAL: 020000 (2 hours)
- OFLI_WAIT_INTERVAL: 001000 (10 minutes)

Scheduling of the Offline Interceptor by the Notification Server would be performed as follows:

- If Connector for RACF was shut down at 23:00 and then restarted at 01:30, the Offline Interceptor would be activated at 01:40 (Connector for RACF start time + OFLI_WAIT_INTERVAL), as this is within the 2-hour deviation permitted by parameter OFLI_RUN_INTERVAL.
- If Connector for RACF was shut down at 23:00 and then restarted at 01:55, the Offline Interceptor would not be activated for its scheduled run time of 00:00. This is because the earliest possible activation time for the Offline Interceptor is 2:05 (Connector for RACF start time + OFLI_WAIT_INTERVAL). This exceeds the 2-

hour deviation permitted by parameter OFLI_RUN_INTERVAL by 5 minutes. The next activation of the Off-line Interceptor would occur at 12:00.

Stopping the Notification and Transaction Servers

The Notification Server and/or Transaction Server can be stopped without having to shut down the Connector for RACF Gateway.

To stop the Notification Server or Transaction Server:

1. Issue the following command:

   ```
   F CTSGATE,STOPASID=<nn>
   ```

   where <*nn*> is the ID of the server to be stopped.

Example

The Notification Server is typically server 01 and the Transaction Server is server 02. Given this situation, you use the commands that follow:

- To stop the Notification Server: `F CTSGATE,STOPASID=01`
- To stop the Transaction Server: `F CTSGATE,STOPASID=02`

Restarting the Notification and Transaction Servers

If either the Notification Server or a Transaction Server terminates for any reason, it can be restarted without having to shut down the Connector for RACF Gateway.

To restart the Notification Server or Transaction Server:

1. Issue the following command:

   ```
   F CTSGATE,STARTASID=<nn>
   ```

   where <*nn*> is the ID of the server to be restarted.

Example:

The Notification Server is typically server 01 and the Transaction Server is server 02, 03 or 04. Given this situation, you would use the commands that follow.

- To restart the Notification Server: `F CTSGATE,STARTASID=01`
- To restart a Transaction Server: `F CTSGATE,STARTASID=02, 03 or 04`

Viewing System Status

This section describes how to display the list of servers and the status of each. If the server is currently handling a specific service, the service is also displayed.

To view the system status:

1. Issue the following command:

   ```
   F CTSGATE,STATUS
   ```

Maintenance

The Connector for RACF provides general maintenance procedures for the administrator’s use. These procedures are run by submitting batch jobs.

The following procedures are discussed in this section:

Formatting the Diagnostic Level Dataset

This procedure formats the diagnostic level dataset and should only be used at the request of Technical Support.

Member CTSDIAG in the Connector JCL library is a sample job which activates this procedure. The M= parameter refers to the relevant diagnostic flags member, which can be CTSACS, CTSACD or CTSAONI. These members are in the PARM library and should be updated with relevant flags based on requests from Technical Support.

Displaying Local Connector for RACF Data

This procedure is activated by member STATUS1 in the Connector JCL library. When STATUS1 job is submitted, the following Connector for RACF information is written to the job’s sysout:

> **Caution**
> This procedure creates a report containing information about the local Connector for RACF instance. This job should only be used at the request of Technical Support.

- SMP’s PTFs
- RSSPARM
- CTSPUSR
- RSSAPI
- RSSTABL
- RSSKWDS
- CTSPARM

Setting Transmitted Data Encryption

> **Note**
> Transmitted Data Encryption is not supported for communication with IdentityNow

All data transmitted between Connector for RACF and IdentityIQ is (optionally) encrypted using an encryption key from the Encryption Key file.

Since the transmitted data must be understood by IdentityIQ and by Connector for RACF, the Encryption Key file in both systems must match.

If encryption was activated during Connector for RACF installation, a synchronized Encryption Key file already exists. If you wish to change the Encryption Key file, it must be changed from IdentityIQ and then synchronized with Connector for RACF.

Change the Transmitted Data Encryption Key

This is a two-part procedure in which part of the procedure is performed in IdentityIQ and part in Connector for RACF. Follow the procedures for encryption of security data in the *IdentityIQ Administration Guide*. At the appropriate point in that procedure, perform the supplemental steps listed below.

1. Stop the Connector for RACF Gateway and servers by specifying the following operator command:

   ```
   P CTSGATE
   ```

2. Transfer a copy of the Encryption Key file from IdentityIQ to Connector for RACF as described in 9.4 – Set Up Secured Communication procedure.

3. Start the Connector for RACF Gateway (which automatically starts the Connector for RACF servers) by specifying the following operator command:

   ```
   S CTSGATE
   ```

Setting Stored Data Encryption

All data which is stored temporarily in Connector for RACF is encrypted using a Stored Data Encryption key (which differs from the Transmitted Data Encryption key). For example, sensitive security information that is written by the Interceptors to the Connector queue file is encrypted using the Stored Data Encryption key.

As part of the Connector for RACF installation procedure, an encryption key for stored data is created. However, the Stored Data Encryption key can be changed periodically to strengthen security.

> **Note**
> Before changing the encryption key, or before enabling or disabling Stored Data Encryption, verify that the Connector queue file does not contain data. This is because the Notification Server cannot process data in the Connector queue file which was encrypted by a previous key.

The Stored Data Encryption key is used internally by Connector for RACF; there is no need to synchronize this key with SailPoint or any other Connector for RACF installation.

The following procedures are described below:

- Generating a new Stored Data Encryption key
- Disabling (or enabling) the encryption of stored data

Generate a New Stored Data Encryption Key

To generate a new stored data encryption key:

1. Verify that the Connector queue dataset doesn't contain data (using the procedure Printing the Connector Queue).

2. Stop the Connector Online Interceptor by specifying the following operator command:

   ```
   P CTSAONI
   ```

3. Stop the Connector for RACF Gateway and servers by specifying the following operator command:

   ```
   P CTSGATE
   ```

4. Edit member CTSKGEN in the JCL library.

5. Submit the job and a new key is generated.

   All job steps must end with a condition code of `0`.

6. Start the Connector for RACF Gateway (which automatically starts the Connector for RACF servers) by specifying the following operator command:

   ```
   S CTSGATE
   ```

7. Start the Connector Online Interceptor by specifying the following operator command:

   ```
   S CTSAONI
   ```

Disable or Enable Stored Data Encryption

To disable (or enable) the encryption of stored data:

1. Verify that the Connector queue dataset does not contain data. (using procedure Printing the Connector Queue).

2. Stop the Connector Online Interceptor by specifying the following operator command:

   ```
   P CTSAONI
   ```

3. Stop the Connector for RACF Gateway and servers by specifying the following operator command:

   ```
   P CTSGATE
   ```

4. Edit member CTSPRSV in the Connector PARM library.

5. Set the ENCR_INT_ACT parameter to one of the following settings:

   - `N` – Disables Stored Data Encryption

   - `Y` – Enable Stored Data Encryption

6. Save the member.

7. Start the Connector for RACF Gateway (which automatically starts the Connector for RACF servers) by specifying the following operator command:

   ```
   S CTSGATE
   ```

8. Start the Connector Online Interceptor by specifying the following operator command:

   ```
   S CTSAONI
   ```

Formatting the Offline Interceptor Dataset

This procedure formats the Offline Interceptor dataset. This procedure is used during installation and to initialize the Offline Interceptor dataset if it was deleted and re-allocated.

Member FORMOFLI in the Connector JCL library is a sample job which activates this procedure. See Starting the Offline Interceptor Manually

Recovering the Offline Interceptor After Failure

The Offline Interceptor operates in one of the following modes:

- **Init Mode** – The Offline Interceptor typically runs in Init mode the first time it is activated. In this mode, Offline Interceptor IMG files are created. These files contain an image of the security objects that are currently defined in RACF. When operating in Init mode, the Offline Interceptor does not send any events to IdentityIQ.
- **Standard Mode** – The Offline Interceptor typically runs in Standard mode each time it is activated after its initial run. In this mode, the Offline Interceptor compares the current snapshot of RACF security objects to that in the IMG files. The differences represent security events and are sent to IdentityIQ.

By default, the Offline Interceptor operates in Init mode if the IMG files do not exist, and operates in Standard mode if the IMG files exist.

In the event an Offline Interceptor failure occurs, (for example due to RACF shutdown or manual cancellation), the IMG files may be incomplete or corrupt. To rectify this situation, the Offline Interceptor must be forced to run again in Init mode.

Force the Offline Interceptor to Run in Init Mode

Use one of the following methods set the Offline Interceptor to run in Init Mode.

> **Caution**
> SailPoint recommends that you consult with Technical Support before using this procedure.

Method A

1. Delete Offline Interceptor working and IMG files.
2. Delete all datasets with the following prefixes:

    - %OLPREFS%
    - %OLVERS%
    - %RSSNAME%
    - OFL*

3. Run the Offline Interceptor.

   This may be done manually or automatically. The Offline Interceptor will run in Init mode and creates new IMG files. (See Starting the Offline Interceptor Manually)

Method B

1. Do one of the following:

   - Use sample job CTSOFLMI in the Connector JCL library. This job executes the Standard Offline Interceptor while setting the mode to Init. Make sure the user submitting the job has permissions to all files used by the job.

   - Specify the following command:

     ```
     S CTSOFLI,PARM=‘-I rssName’
     ```

     where *rssName* is the name set in parameter RSS in the CTSOFLI procedure.

Any future run of the Offline Interceptor will automatically be performed in Standard mode.

Initializing the Connector Queue

Use this procedure to initialize the Connector queue during the installation process. You can also use this procedure to initialize the Connector queue dataset if the dataset was deleted and re-allocated.

1. Stop all the Connector for RACF and Online Interceptor processes. Ensure that all the Connector for RACF and Interceptor processes have shut down.

2. Submit FORMQUE member in the Connector JCL library (a sample job which activates this procedure).

3. Restart the Connector for RACF and the Online Interceptor.

Changing the Size of the Connector Queue

Depending on the level of activity in the Managed System managed by Connector for RACF, it may become necessary to change the size of the Connector queue. This section describes how to change the queue size.

> **Note**
> You can change the size of the Connector queue dataset regardless of whether or not it contains data. Data contained in the queue is preserved during this procedure.

To change the size of the Connector queue:

1. Stop the Connector Online Interceptor by specifying the following operator command:

   ```
   P CTSAONI
   ```

2. Stop the Connector for RACF Gateway and servers by specifying the following operator command:

   ```
   P CTSGATE
   ```

3. Ensure that all the Connector for RACF and Interceptor processes have shut down.

4. Allocate a new Queue dataset with increased DASD space, using a dataset name suffix other than ".QUEUE".

   For example if your existing Queue dataset is "CTSA.V400.QUEUE", allocate the new Queue dataset as "CTSA.V400.QUEUEN2".

   > **Note**
   > For sample allocation of a Queue dataset, see job FORMCTS in the Connector INSTALL library.

5. Edit sample job CTSQCR in the Connector JCL library, and specify the dataset name suffix for the new Queue (example "QUEUEN2").

   This job consists of two steps:

   - Step 1 calls the CTSAQCR JCL procedure which calls the CTSQCR utility with appropriate parameters. The utility expects the new Queue dataset to be pre-defined and allocated. The utility formats the new Queue dataset and then copies the contents of the active Queue dataset to the new Queue dataset.

   - Step 2 calls the IDCAMS utility to change the suffix of the active Queue dataset (for example, to ".OLDQUEUE") so that it is no longer active, and to change the suffix of the new Queue dataset so that it becomes the active Queue dataset. Renaming of datasets can also be done using other methods available under OS/390, such as TSO or ISPF commands.

6. Submit the job.

7. Upon the successful termination of job CTSQCR and dataset rename commands, restart the Connector for RACF and the Online Interceptor. The new Queue dataset is now active.

Printing the Connector Queue

This procedure prints the contents of the Connector queue dataset.

Member PRTQUE in the Connector JCL library is a sample job which activates this procedure.

Renaming a Managed System

You can change the name assigned to the Managed System during Connector for RACF installation.

Once the Managed System has been defined in SailPoint, renaming the Managed System involves changes both on the SailPoint workstation and in Connector for RACF. The procedure described below changes the name of the Managed System only in Connector for RACF.

> **Note**
> This procedure should only be performed within the framework of the procedure for changing the name of the Managed System in SailPoint.

The name of an Managed System is changed in Connector for RACF using utility CTSAADPT.

Member CTSADAPT in the JCL library contains a sample job to invoke the JCL procedure for the CTSAADPT utility (procedure `<prefix>AADPT`, where *<prefix>* is the prefix of your Connector JCL procedures). The utility should be run when Connector for RACF and the Interceptors are inactive.

Specify the old and new Managed System names as values of parameters FROMRSS and TORSS respectively when invoking the `<prefix>AADPT` procedure to execute the CTSAADPT utility.

This will change all occurrences of the old Managed System name in Connector for RACF datasets to new Managed System name. Note that the RSSPARM member itself is also modified.

The utility reports which datasets (and how many records in each) were modified. Most datasets considered for modification are allocated via JCL (see the relevant DD statements in the `<prefix>AADPT` JCL procedure).

One exception to this is the OFLRIMG dataset, used by Offline Interceptor utility. The DSNAME for this dataset is determined during CTSAADPT execution by the value of the RSS_WORK_DIR parameter for TORSS name in the RSSPARM member. The DSNAME dynamically allocated is the concatenation of the value of RSS_WORK_DIR with the suffix OFLRIMG (for example: CTSA.V400.MYRSS.OFLRIMG). If this DSNAME does not exist, no dynamic allocation (and thus, no modification) is done by CTSAADPT for this dataset.

Before running the CTSAADPT utility, verify that the RSS_WORK_DIR parameter value conforms to MVS dataset naming conventions.

> **Note**
> If the new Managed System name is longer than 8 characters, additional adjustments must be performed. For more information, see Step 11 – Adjust for Longer Managed System Names.

The CTSAADPT utility does not modify the Connector for RACF started task procedures. After running the utility, the following additional changes must be done manually.

Changes to Online Interceptor

Change the name of the Managed System in the CTSAONI (Online Interceptor) started task procedure as follows:

1. Stop all the Connector for RACF and Online Interceptor processes. Ensure that all the Connector for RACF and Interceptor processes have shut down.
2. Edit the CTSAONI member in the PROCLIB library containing Connector for RACF procedures.

   The name of the PROCLIB library is defined in variable **%PROCLIB%** of member DEFPARMS in the INSTALL library.
3. Locate the following line:

   ```
   Managed System=<rss_name>
   ```
4. Modify the value for `<rss_name>` to the new Managed System name.
5. Save the member.
6. Restart the Connector for RACF and the Online Interceptor.

   > **Note**
   > If you use the Delayed Delete utility, repeat steps 2 and 3 for procedure CTSC100. For more information, see Renaming a Managed System and Shared RACF Database Support.

Changes to Offline Interceptor

Change the name of the Managed System in the CTSAOFI (Offline Interceptor) procedure as follows:

1. Edit member CTSAOFI in the PROCLIB library containing Connector for RACF procedures.

   The name of the PROCLIB library is defined in variable %PROCLIB% of member DEFPARMS in the INSTALL library.
2. Locate the following line:

   ```
   Managed System=<rssName>
   ```
3. Modify the value for `<rssName>` to the new Managed System name.
4. Save the member.

Filtering Interception Messages

Events affecting the Managed System database that are intercepted by Connector for RACF are recorded in Connector for RACF log files. However, not all intercepted events are relevant for Connector for RACF. You can set RSSPARM parameters in the member RSSPARM to filter the interception messages that are recorded in the log files.

The following Managed System parameters from the member RSSPARM are used to control filtering of interception messages:

LOG_INTERCEPT_MSG

Specifies the types of interception messages to be recorded in the CD log file.

Possible values for this parameter are described in the following table:

Value	Description
ALL	All messages are recorded, indicating in each case whether the intercepted event was sent to SailPoint. This is the default value.
ACCEPTED	Only Managed System data included in aggregation (and therefore sent to SailPoint) is recorded.
IGNORED	Only Managed System data not included in aggregation (and therefore not sent to SailPoint) is recorded.
NONE	No messages are recorded.

LOG_GET_MSG

Specifies the filtering option for intercepted messages generated by the synchronization action (Managed System Retrieval Transaction). The messages are recorded in the CS log file.

Possible values for this parameter are described in the following table:

Value	Description
ALL	All Sync messages are recorded. This is the default value.
NONE	No Sync messages are recorded.

To filter interception messages:

1. Stop the Connector for RACF as described in Shutting Down the Connector for RACF.

2. Edit the RSSPARM member in the Connector PARM library.

3. Insert or modify either or both of the following parameters as necessary:

```
rss_name LOG_INTERCEPT_MSG    ALL <== Modify as required
rss_name LOG_GET_MSG          ALL <== Modify as required
```

4. Save the member and exit.

5. Restart the Connector for RACF.

Interception Acknowledgment

The Connector for RACF sends intercepted Managed System events to IdentityIQ. When the *Interception Acknowledgment* function is active, IdentityIQ sends acknowledgment for events received to the Connector for RACF. This process is managed by the **INTERCEPT_SEND_MAX** parameter in the RSSPARM file. This parameter determines whether or not the Connector for RACF waits for acknowledgment from IdentityIQ for each intercepted Managed System event sent before sending the next event.

- When the **INTERCEPT_SEND_MAX** parameter is not present in the RSSPARM file, or is set to `0`, the interception acknowledgment mechanism is disabled. The Connector for RACF sends events without waiting for acknowledgment.

- When **INTERCEPT_SEND_MAX** is set to `1`, Connector for RACF waits for acknowledgment after each event is sent.

To set Interception Acknowledgment:

1. Stop the Connector for RACF as described in Starting and Stopping the Online Interceptor.

2. Edit member RSSPARM in the Connector PARM library.

```
ALL_RSS    INTERCEPT_SEND_MAX     1
```

3. Save the member and exit.

4. Restart the Connector for RACF.

Maintain Custom Field Related Keywords

When changes are made to custom field definitions in RACF, the custom fields-related keywords in the Account or Group entities should be re-defined according to the new custom field definitions. Example changes can include adding, altering, or deleting custom fields.

To re-define the Custom Fields related keywords:

1. Stop the Connector Gateway and Connector servers by specifying the following command:

   ```
   P CTSGATE
   ```

2. Restart the Connector Gateway and Connector servers by specifying the following command:

   ```
   S CTSGATE
   ```

3. Update Application schema in SailPoint with the new custom fields.

4. Aggregate all Accounts and Groups in order to retrieve the new Custom Fields data.

Remove Custom Fields Support

If there is a need to stop handling custom fields data via Connector and SailPoint, the custom fields support can be removed. The support removal process includes removing the custom fields-related keywords defined for Account and/or Group entities and removing all the custom fields data from accounts and/or groups.

To remove support for custom fields:

1. Stop the Connector gateway and Connector servers by specifying the following command:

   ```
   P CTSGATE
   ```

2. Edit the RSSPARM member in the Connector PARM library.

3. Change the value in the CUSTOM_FIELDS_SUPPORT parameter to `N`:

   ```
   managedSystemName CUSTOM_FIELDS_SUPPORT N
   ```

4. Save the member.

5. Restart the Connector gateway and Connector servers by specifying the following command:

   ```
   S CTSGATE
   ```

6. Delete all custom fields from application schema in SailPoint.

7. Aggregate all Accounts and Groups in order to remove the custom fields data.

Scripts

The Connector for RACF functions (described in this book) are designed to meet fundamental Managed System requirements.

In addition to the basic requirements, site-specific requirements may exist in established Managed System, and Managed System may have changing needs. Therefore, Connector for RACF enables customization of the functions. You can customize any Connector for RACF function by writing scripts.

A script is a group of statements that perform one or more actions and that manipulate standard SailPoint and Managed System specific fields. A script may include conditions that determine when actions should be performed. For example, cleanup activities should be run if the Connector for RACF function does not execute successfully.

Scripts can also be used to automate any required actions that are currently performed manually.

The Connector for RACF has the ability to call a script before and after executing a Connector for RACF function. Parameters may be received from and returned to Connector for RACF functions by including script commands in the scripts.

Writing a Script

Scripts are written in REXX language and must adhere to REXX syntax rules. Use a text editor to write the scripts and store them in the following library:

```
<prefix>.<version>.USER.CLIST
```

where:

- `<prefix>` – Value set in parameter OLPREFS in member LOADCTS in the Connector INSTALL library.
- `<version>` – Value set in parameter OLVERS in member LOADCTS in the Connector INSTALL library.

> **Note**
> This library name is defined in parameter SCRIPT_DIR in member RSSPARM stored in the Connector PARM library.

Scripts may be executed before and/or after a Connector for RACF function is executed. All scripts are activated under the Connector for RACF address space. For more information, see Executing a Script.

A script should consist of the following sequence of actions:

1. Read the current Connector for RACF variables into REXX variables.
2. Examine the REXX variables, modify them (if required) and execute additional actions as required.
3. Update the Connector for RACF variables with the resultant REXX variable values.

To enable the script to manipulate Connector for RACF variables, Connector for RACF provides the script command CTSAVAR which reads the current Connector for RACF variables into REXX variables. After the REXX variables have been examined and modified, the script command CTSAVAR is used to update the Connector for RACF variables with the resultant REXX variable values. Script commands are described later in this chapter.

Since the script is executed in the TSO-REXX environment, Connector for RACF scripts can also issue TSO and REXX commands.

When writing a script, the following must be taken into account:

- Certain variables are unmodifiable. For more information, see Script Variables.
- The return code of a script must be set by the script before it terminates for more information see Setting the Return Code.
- Several TSO commands can only be issued indirectly from within a Connector for RACF script. For more information, see TSO Considerations.
- Due to REXX limitations, field names for keywords must be all uppercase. This applies both when defining keywords in SailPoint and when specifying the field name of the keyword in a script.

Executing a Script

The Connector for RACF determines which scripts to execute and whether or not to run the scripts by examining the values of the following parameters in member RSSAPI in the Connector PARM library.

Pre/Post-Scripts

One script can be called before a RACF function executes and another script (or the same one) can be called after the RACF function executes. The scripts are referred to as:

Name	Description
Pre-script	This script is run immediately before the RACF function executes.
Post-script	This script is run immediately after the RACF function executes.

> **Note**
> When using GET transactions, only post-scripts are called.

The RACF execution of the functions is as follows:

1. If a pre-script is specified and should be executed, the Connector for RACF invokes it. Otherwise, execution of the RACF function begins with step 3.
2. The return code of the pre-script is checked. If the script has returned **SKIP** or **FATAL**, the Connector for RACF does not invoke the functions and execution skips to step 4.
3. If the RACF function is to be invoked, it is called.
4. If a post-script is specified and should be executed, the Connector for RACF invokes it, regardless of the return codes from the pre-script and the actual RACF function.
5. The final return code is the return code of the last item (pre/post-script or actual RACF) that was executed.
6. If both of the following conditions are satisfied:
   - The function is an ADD or UPDATE function.
   - A GET post-script is specified for the object and should be executed.

   The Connector for RACF invokes the GET post-script after invoking any ADD or UPDATE pre/post-scripts required by the function. This is done to synchronize the Managed System database with the IdentityIQ database.

Structure of the RSSAPI Member

Each line in the RSSAPI member (excluding comment lines) represents a Connector for RACF call which is followed by various parameters. The parameters influence the behavior of the Connector for RACF. Each line conforms to the following format:

```
Managed System-type Managed System-name API-name NUM PRF ACF POF PRN PON
```

The following table describes the parameters appearing in each line of the RSSAPI member.

Parameter	Description
Managed System-type	Managed System type
Managed System-name	Managed System name
API-name	Connector for RACF functional name
NUM	Maximum number of keywords that may be added to a transaction by the Pre-script for use by the Connector for RACF function or Post-script
PRF	Y/N flag indicating whether or not to invoke the Pre-script
ACF	Y/N flag indicating whether or not to invoke the actual Connector for RACF routine
POF	Y/N flag indicating whether or not to invoke the Post-script
PRN	Pre-script name
PON	Post-script name

Example

```
- MVS ADDUSER 05 N Y Y ACFADDU1 ACFADDU2
```

This line indicates that the Connector for RACF function ADDUSER in Managed System MVS operates as follows:

- Does not call the Pre-script ACFADDU1.
- Performs the actual Connector for RACF routine.
- Calls the post-script ACFADDU2.

A script can be called as a pre-script and/or post-script for any Connector for RACF function. For example, assume that you wish to perform the same action before every Connector for RACF function. First write a script which performs the desired action. Next, specify the script name in parameter *PRN* for all the Connector for RACF functions and set the parameter *PRF* to Y.

Script Variables

When executed within the Connector for RACF environment, a script has access to certain predefined variables. These variables are used for a variety of purposes, including:

- Receiving and modifying values of Managed System-specific fields
- Receiving information regarding the Connector for RACF environment
- Controlling actions performed by the Connector for RACF function
- Controlling execution of subsequent scripts

Types of Variables

The Connector for RACF distinguishes between the different kinds of information that pass to and from the scripts by using different categories of variables; each category of variable is assigned a different prefix.

The categories of variables are:

- **CTSA0.<field_name>** (Read-only) – Predefined variables representing SailPoint structure fields (common to all Managed System types) or Connector for RACF environment information.

  For a full list of CTSA0 variables, see CTSA0 Variables.

- **CTSA1.<field_name>** (Read/write) – Predefined Managed System-specific variables containing values sent from SailPoint. If the value of a field was modified in SailPoint, the corresponding CTSA1 variable contains the modified value. The value of the CTSA1 variable can be further modified by the script.

  For more information, see Appendix E: Managed System-Specific Fields.

  In addition to the variables described above, the following special CTSA1 variables are available:

  - **CTSA1.MSG** (Write-only): Used to send a message from the script to SailPoint. This variable is defined by the script; use of the variable is optional. A string value assigned to this variable is sent to SailPoint and is displayed in the Transaction Properties window.

  - **CTSA1.RC** (Write-only) – Return code of the pre- or post-script. Before Connector for RACF executes a function, it checks the pre-script return code to determine whether or not to execute the function. Therefore, the return code of a pre-script must be set by the script before it terminates.

    For more information, see Setting the Return Code.

- **CTSA2.<field_name>** (Read-only) – Predefined Managed System-specific variables containing the original values of fields. The values are retrieved from the Managed System.

  These variables are similar to CTSA1 variables. The same field can be referenced with both the CTSA1 and CTSA2 prefixes. However, the CTSA1 variable contains the new value of the field as sent from SailPoint while the CTSA2 variable contains the original "old" field value.

After the Managed System has been updated, each CTSA2 variable passed to the script contains the new, updated value of the corresponding field.

- **CTSA9.<rss_parameter>** (Read-only) – Variables containing values of parameters in the RSSPARM file.

  The script only receives CTSA9 variable values when the parameter SEND_RSSPRM_TO_SCRIPT in the RSSPARM file is assigned the value `Y`.

List Fields

Managed System-specific information may include List (table) fields. A List field is passed to a script as a string of entries. This string includes specific characters that are designated to separate entries and sub-fields in the list.

The default entry separator value is a comma (,). To specify a value other than the default, assign any printable character to the Managed System parameter SCRIPT_ SEP_ENTRY in the RSSPARM file.

The default sub-field separator value is a semicolon (;). To specify a value other than the default, assign any printable character to the Managed System parameter in SCRIPT_SEP_FIELD in the RSSPARM file.

Example

Given the following:

- SCRIPT_ SEP_ENTRY is set to **^**
- SCRIPT_SEP_FIELD is set to **$**

A List field contains the following data:

A1	A2	A3
B1	B2	B3
C1	C2	C3
D1	D2	D3

The above table of data values will be passed to the variable in a script as:

```
A1$A2$A3^B1$B2$B3^C1$C2$C3^D1$D2$D3
```

CTSA0 Variables

CTSA0 variables and their values are listed below:

Connector for RACF Environment Variables

The following table lists variables containing information that relates to the environment in which the script is executed. These variables are common to all RACF function types.

Variable	Value	Description
CTSA0.ACTION	SCRIPTPRE	Call as a Pre-script
	SCRIPTPOST	Call as a Post-script

Variable	Value	Description
CTSA0.FUNC_NAME	ADDU2UG	Connect a user to a group
	ADDUG	Create a new group
	ADDUSER	Create a new user
	DELU2UG	Disconnect a user from a group
	DELUG	Delete a group
	DELUSER	Delete a user
	GETUGS	Obtain group details
	GTRSPRM	Obtain Managed System details
	GTUG2UC	Obtain user to group connection details
	GTUSERS	Obtain user details
	REVUSER	Revoke/Restore a user
	UPD_PASS	Update password of a user
	UPDUSER	Update user details. **Note** When a transaction is issued from SailPoint for any of the following actions: • changing the user’s password • revoking the user • restoring the user With no added Managed System-specific or user-defined fields, the transaction type can be either UPDUSER, or it can be UPD_PASS / REVUSER, depending on how the action was initiated. However, when the transaction includes added Managed System-specific or user-specific fields, the transaction type is always UPDUSER.
	UPDU2UG	Update user to group connection details
	UPDUG	Update group details
CTSA0.ACT_RC	OK	Return code of the actual Connector for RACF. Available in the Post-script only
	ERROR	

Variable	Value	Description
	FATAL	
	<UNDEFINED>	
CTSA0.ADM_G		Group of administrator performing the operation
CTSA0.ADM_VER	4.0.01	Current version of Connector for RACF.
CTSA0.ADM_MOD	1	Reserved
CTSA0.ADM_ID		User ID of administrator performing the operation.
CTSA0.ADM_PASSWD		Managed System Administrator password or phrase. This parameter is passed to the scripts, only when the parameter SEND_PWD_TO_SCRIPT in the file RSSPARM is set to Y.
CTSA0.PRE_RC	OK	Return code of the Pre-script. Available in the Post-script only
	WARN	
	SKIP	
	ERROR	
	FATAL	
	<UNDEFINED>	

Managed System User Connector for RACF Function Variables

The following table describes variables available for any Managed System user operation.

Variable	Value	Description
CTSA0.USER_ID		User ID
CTSA0.USER_PWD	<password or phrase>	Managed System user new password or phrase. This parameter is passed only when parameter PASS_PASSWORD in file RSSPARM is set to Y.
CTSA0.UG_DEF		Default group of the user
CTSA0.USER_ADMIN	1	User is a regular user
	2	User is an auditor
	3	User is an administrator
	4	User is an auditor and administrator
	5	Ignore this field
CTSA0.USER_STA	1	User is revoked
	2	User is restored
	3	Ignore this field

Variable	Value	Description
CTSA0.PWD_LIFE	1	Permanent
	2	Temporary
	3	Ignore this field
CTSA0.RSS_NAME		Name of the Managed System
CTSA0.RSS_TYPE		Type of Managed System

Group Connector for RACF Functions

The following table describes variables available for any Group Connector for RACF operation.

Variables	Value	Description
CTSA0.GROUP_ID		Group ID
CTSA0.GROUP_PR		Parent Group
CTSA0.RSS_NAME		Name of the Managed System
CTSA0.RSS_TYPE		Type of Managed System

Managed System User–Group Connector for RACF Functions

The following table describes variables available for any Managed System User—Group Connector for RACF operation.

Variable	Value	Description
CTSA0.GROUP_ID		Group ID
CTSA0.USER_ID		User ID
CTSA0.U2UG_ATR	1	Regular connection between a user and a group
	2	Connection is of user to its default group
	3	Ignore this field
CTSA0.U2UG_MSC	1	User is a regular member of the group
	2	User is an administrator of the group
	3	User is an auditor of the group
	4	User is administrator and auditor of the group
	5	Ignore this field
CTSA0.RSS_NAME		Name of the Managed System
CTSA0.RSS_TYPE		Type of Managed System

Setting the Return Code

Before Connector for RACF executes a Connector function, it checks the Pre-script’s return code to determine whether or not to execute the function. Therefore, the return code of a Pre-script must be set by the script before it terminates.

To set the return code in a Pre-script:

In the script, set variable **CTSA1.RCODE** to one of the following values:

Value	Description
OK	Pre-script processing completed successfully.
SKIP	Do not call the Connector for RACF function, but do call the Post-script.
WARN	Continue and call both the Connector for RACF function and the Post-script.
ERROR	Continue and call both the Connector for RACF function and the Post-script.
FATAL	Do not call the Connector for RACF function, but do call the Post-script.

To set the return code in a Post-script:

In the script, set variable **CTSA1.RCODE** to one of the following values:

Value	Description
OK	Post-script processing completed successfully.
WARN	Post-script processing completed with warning.
SKIP	(GET Post-script only) Do not send the retrieved object (for example, Managed System user, resource) to SailPoint. Using this value enables a Post-script to filter the objects to be aggregated to SailPoint, regardless of the aggregation.
ERROR	Post-script processing completed with error.
FATAL	Post-script processing completed with fatal error.

TSO Considerations

The TSO environment is available to scripts and therefore most TSO commands can be issued by the script. However, the following TSO commands can't be issued directly from within a Connector for RACF script.

- Authorized TSO commands
- SUBMIT command

Authorized TSO Command

Authorized TSO commands cannot be issued directly from a script.

To execute an authorized TSO command:

Use the CTSAEXC command processor.

For example, to activate command IDCAMS DEFINE (which is an authorized command processor) enter the following statement in the script:

```
CTSAEXC DEFINE ....define command arguments...
```

SUBMIT Command

The TSO SUBMIT command cannot be issued directly from a script.

To submit a job:

Use the CTSASUB command processor. CTSASUB receives the name of a DD statement which contains the job JCL image.

Example

Include the following statement in the script:

```
CTSASUB TEST
```

This activates command CTSASUB which reads DD statement TEST and writes its contents to the JES internal reader. An exclusive JES internal reader is allocated to the Connector for RACF started task via DD statement INTRDR in the task’s JCL. If no DD statement is specified for command CTSASUB, a default DD statement CTSJOBIN is used and its contents are written to the JES internal reader.

Script Commands

The following table provides a brief description of the available script commands.

Command	Description
CTSAEXC	Activates authorized TSO command processors.
CTSASUB	Submits a job.

Command	Description
CTSAVAR	Copies Connector for RACF variables to REXX variables. *OR* Updates Connector for RACF variables based on the REXX variables.
CTSASYNC	Generates a synchronization event.

CTSAEXC

Purpose	Activates authorized TSO command processors in the Connector for RACF script
Syntax	`CTSAEXC <TSO_command>` where `<TSO_command>` is any authorized TSO command.
Description	Activates authorized TSO command processors in a Connector for RACF script.
Example	The following statement activates the IDCAMS DEFINE command: `CTSAEXC DEFINE ALIAS NAME(USER1) RELATE(CATALOG.USERS)`

CTSASUB

Purpose	Submits jobs from a Connector for RACF script
Syntax	`CTSASUB [<ddname>]` where `<ddname>` is the name of the DD statement which contains the JCL cards.
Description	Submits jobs from a Connector for RACF script.
Example	The following statement submits the job contained in the dataset allocated to DD statement MYJOB: `CTSASUB MYJOB`

CTSAVAR

Purpose	Copies Connector for RACF variables to REXX variables or updates Connector for RACF variables based on the REXX variables
Syntax	`CTSAVAR {GET | PUT} <token>` where: • `GET`—Sets the REXX variables to the values in the Connector variables. • `PUT`—Updates the Connector variables to the values in the REXX variables. • `<token>`—Token passed to the script as a parameter which is used to identify the relevant Connector for RACF parameters by the CTSAVAR command.
Description	Copies Connector for RACF variables to REXX variables or updates Connector for RACF variables based on the REXX variables.

Examples	The following statement sets the REXX variables to the values in the Connector for RACF variables:
	`CTSAVAR GET <token>`
	The following statement updates the Connector for RACF variables to the values in the REXX variables:
	`CTSAVAR PUT <token>`

CTSASYNC

Purpose	Triggers synchronization of Managed System details with SailPoint

Syntax	One of the following:
	`CTSASYNC <token> USER <user>`
	`CTSASYNC <token> GROUP <group>`
	`CTSASYNC <token> CONN <conn_user>     <conn_group>`
	where:
	• `<token>` – Token passed to the script as a parameter which is used to identify the relevant Connector parameters by the CTSASYNC command.
	• `<user>` – User name of the user to be synchronized.
	• `<group>` – Group name of the group to be synchronized.

Purpose	Triggers synchronization of Managed System details with SailPoint
	• `<conn_user>` – User name of the user in the connection to be synchronized. • `<conn_group>` – Group name of the group in the connection to be synchronized.
Description	Triggers synchronization of the Managed System with SailPoint.
Example	The following statement triggers synchronization of the USER details between SailPoint and Managed System. `CTSASYNC <token> USER USER1`

Appendix A: Maintaining Connector for RACF using SMP/E

The design of the Connector for RACF SMP/E implementation accomplishes the following:

- Having as few modifications to the existing installation process as possible.

  This is achieved by enhancing the installation process to load the SMP/E dataset from tape into datasets with customizable names, and establish a complete SMP/E environment.

- Providing the user with an SMP/E environment that represents, as closely as possible, the target system, and that requires very few local modifications and tailoring during the Connector for RACF installation.

  This is achieved by shipping a complete set of SMP/E datasets, representing the environment being installed by the Connector for RACF installation process. These datasets are pre-loaded with information representing the environment installed. There is no need to perform any of SMP/E's traditional installation steps (that is, RECEIVE/APPLY/ACCEPT) to perform the Connector for RACF installation. Upon completion of the installation process, the SMP/E datasets loaded represent the installed Connector for RACF product environment.

Packaging of Connector for RACF using SMP/E

The Connector for RACF product is shipped using a pre-installed SMP/E environment. The environment shipped includes SMP/E's CSI, service datasets, and distribution and target libraries. All these datasets are unloaded during the Connector for RACF installation jobs.

SMP/E parameters that are environment dependent are modified by the Connector for RACF installation process to contain the values specified by the user. When the installation process is complete, the SMP/E CSI and related datasets are customized to reflect the site environment.

Following the customization step, you will find a complete SMP/E environment, containing the Global zone, a target zone and a distribution zone.

The DDDEF names for the target and distribution libraries may be seen in Appendix C: Connector for RACF Datasets and JCL Procedures.

Zone Structure

The Connector for RACF product environment supplied contains three zones in a single CSI.

Zone	Description
GLOBAL	The SMP/E global zone
CTSATZN	Connector for RACF product target zone
CTSADZN	Connector for RACF product distribution zone

All three zones are contained in a single VSAM KSDS cluster, in a single CSI structure. After the CSI has been loaded and customized by the Connector for RACF installation process, you will be able to move the loaded zones to other CSIs or rename these CSIs to suit their local standards.

Functions Installed

In the SMP/E environment supplied, the functions installed are:

Function	Description
CACF400 CRCF400 CTSA400 CTSS400	Connector for RACF elements
ECA7000 IOA700E IOA700E	Connector for RACF Gateway elements (Level 7.0.0)

Maintenance

Fixes supplied for Connector for RACF are module or other element replacement fixes and are shipped in PTF format.

Possible Changes in RACF Commands Usage Introduced by Maintenance

- The Connector for RACF performs changes by issuing RACF commands (for example: ALTUSER, ADDGROUP, PASSWORD or PHRASE).
- Often a patch (fix or enhancement) to Connector for RACF adds usage of new RACF commands, or usage of new operands to the RACF commands currently used.
- Since RACF enables a RACF administrator to control who is permitted to issue various RACF commands and their operands, it is important for z/OS sites using the connector to be aware of and prepare or adapt the capabilities of the RACF users under which the connector issues the above RACF commands.
- The Connector for RACF issues commands to RACF using Managed System administrator. This is the RACF user defined to handle provisioning operations performed from SailPoint. This administrator typically has the RACF SPECIAL attribute.

Connector for RACF Maintenance Procedures

This section describes the Connector for RACF maintenance procedures in detail.

Running SMP/E Jobs

The Connector for RACF installation supplies a JCL procedure (CTSASMP) designed to run SMP/E, and allocate its CSI, auxiliary datasets, and target and distribution libraries. This procedure should be used for all SMP/E operations involving Connector for RACF.

> **Note**
> The samples below are only intended to demonstrate the most basic use of each SMP/E command; they are not intended to provide the complete format of each command.

For a full description of SMP/E commands, refer to the appropriate SMP/E documentation, especially the SMP/E User’s Guide and SMP/E Reference. Full publication names and IBM Form Numbers are provided in the References section at the end of this appendix.

SMP/E requires the user to define, for each operation, the zone for which the operation will take effect. This definition is done via the SET BOUNDARY SMP/E command. This command must be the first in SMP/E’s command stream, and is effective for all subsequent commands up to the next SET command.

Receiving Maintenance

The SMP/E RECEIVE command loads SYSMODs into SMP/E’s Global zone. The RECEIVE command does not update any element, and its major purpose is to store the SYSMOD in the Global zone for subsequent processing. The following is a sample RECEIVE job:

```
//jobname JOB jobparms
//RECEIVE EXEC CTSASMP
  SET BOUNDARY (GLOBAL).
  RECEIVE [SYSMODS] [SELECT(sysmod1,sysmod2,...)].
/*
//SMPPTFIN DD ...
```

The SMPPTFIN DD statement should point to a sequential dataset (or PDS member) holding the SYSMODs to be processed.

The SELECT parameter of the RECEIVE command can be used to limit its scope to the SYSMODs specified in it. Omitting this parameter will make SMP/E receive all the SYSMODs contained in the dataset pointed to by the SMPPTFIN DD statement. Omitting the SYSMODS parameter will cause SMP/E to attempt and receive HOLD information contained in the SMPHOLD dataset.

Member SMPRECIV in the Connector JCL library, contains JCL tailored to RECEIVE PTFs for Connector for RACF.

Applying Maintenance

After the maintenance is received, it must be installed into the software product to take effect. The installation is done via the APPLY command. The following is a sample APPLY job:

```
//jobname JOB jobparms//APPLY  EXEC CTSASMP
  SET BOUNDARY(CTSATZN).
  APPLY [SELECT(sysmod1,sysmod2,....)].
/*
//
```

Member SMPAPPLY in Connector JCL library, contains JCL tailored to APPLY PTFs for Connector for RACF.

Accepting Maintenance

Following the SYSMODs application into the target zone, and after sufficient time has passed, the SYSMOD should be ACCEPTed into the distribution zone. The following is a sample ACCEPT job:

```
//jobname JOB jobparms
//ACCEPT EXEC CTSASMP
  SET BOUNDARY(CTSADZN).
  ACCEPT [SELECT(sysmod1,sysmod2,....)].
/*
//
```

Member SMPACCPT in Connector JCL library, contains JCL tailored to ACCEPT PTFs for Connector for RACF.

Producing SMP/E Reports

It is recommended that you ACCEPT the maintenance after a certain period of time has elapsed since the maintenance was applied. To verify which SYSMODs have been applied and are not ACCEPTed yet, run the following sample job:

```
//jobname JOB jobparms
//REPORT EXEC CTSASMP
  SET BOUNDARY(GLOBAL).
  REPORT SYSMODS INZONE(CTSATZN) COMPAREDTO(CTSADZN).
/*
//
```

Appendix B: Connector for RACF Configuration Parameters

This appendix describes configuration parameters used by Connector for RACF. Many of these parameters can be modified to suit user requirements.

The Connector for RACF configuration parameters are stored in the following members in the Connector PARM library.

Member	Description
CTSPUSR	Parameters in this member are common to all Connector for RACF platforms. For example, this member includes the parameter that determines whether or not Transmitted Data Encryption is enabled.
RSSPARM	Parameters in this member are specific to RACF. For example, this member includes the parameter that determines the interval between runs of the Offline Interceptor.
RSSAPI	This member contains all Connector for RACF calls and the corresponding script-related parameters.
CTSPARM	Parameters in assembler format which require compile in case they are updated.

The parameters in these members are set during the Connector for RACF installation and customization process and should not be modified.

CTSPUSR – Connector for RACF Parameter

The following table lists the CTSPUSR parameter.

Parameter	Description
WRITE_TO_QUEUE	Whether the messages of account and group aggregations are written to Queue file. Default: `N`. **Note** MAIN_CS MUST be specified in column 1 of CTSPUSR with WRITE_TO_QUEUE parameter.

RSSPARM – Managed System Parameters

This section contains a description of parameters in the RSSPARM member, followed by a listing of the member as it appears after installing the Connector for RACF.

Description of Parameters

Each parameter in the RSSPARM member is applicable either for all MSCSs managed by the Connector for RACF installation or for a specific Managed System.

Each parameter in the RSSPARM member has the following syntax:

```
mscs parameterName value
```

where:

- **mscs** - Name of the Managed System to which the parameter applies. If the parameter applies to all Managed System, contains ALL_RSS.
- **parameterName** - Name of the RSSPARM parameter.
- **value** - Value assigned to the parameter.

The tables that follow describe the parameters that can appear in the RSSPARM member.

> **Note**
> Many of the parameters in the tables are not automatically present in the RSSPARM file after Connector for RACF installation. If you wish to assign a value to a specific parameter, it may be necessary to add the parameter to the file. The value labeled as *Default* appearing in the Values column of the tables that follow indicates the value assigned if the parameter is *not* present in the member or if the parameter is assigned an invalid value. To see the default value for the parameters that *are* present in the member, see General Parameters.

Each table contains the following columns:

Column name	Description
Parameter	Name of the RSSPARM parameter.
	An asterisk (*) in this column indicates that if the parameter is assigned an invalid value, Connector for RACF automatically assigns the parameter the "default" value specified (see the description of the **Values** column below).
	The presence of "(ALL_RSS)" in this column indicates that the parameter is applicable to all MSCSs managed by the Connector for RACF installation. If the parameter is specific to a certain type of Managed System, the parameter is applicable to all MSCSs of that type.
Description	Description of the parameter.
Values	Possible parameter values, or limitations. Where specified, the **Default** value in this column indicates the value assigned if the parameter is not present in the RSSPARM member or if the parameter is assigned an invalid value. To see the default value (if any) for a parameter, locate the parameter in General Parameters.

General Parameters

The following table contains descriptions of RSSPARM parameters which are specified once for all MSCSs managed by the Connector for RACF installation. Each parameter name is preceded by ALL_RSS.

Parameter	Description	Value
CHECK_SYNC_OBJS	During aggregation: Number of entity/connection operations handled by Connector for RACF, after which an "active" confirmation message is sent to SailPoint.	Default: 100
INTERCEPT_SEND_MAX	When INTERCEPT_SEND_MAX is set to a positive numeric value, the Connector for RACF waits for acknowledgment after the amount of events specified are sent. For example, when INTERCEPT_SEND_MAX is set to `10`, Connector for RACF sends 10 event messages to IdentityIQ before waiting for an acknowledgment message.	0: Do not wait (Default) 1 or higher: Number of events which are sent to IdentityIQ before waiting for an acknowledgment from IdentityIQ.

Parameter	Description	Value
OFLI_VERBOSE	Whether the Log Message of the Online and Offline Interceptor is sent to Managed System console.	Y, N Default: N
OCCUPIED_QUEREU_DATA	For future use. Do not change the default value.	Y, N Default: N
STAT_CHKSUM_INTRVL	During aggregation: Number of entity/connection checksums received by Connector for RACF from SailPoint, after which a message is sent to the Connector for RACF log and an event is sent to SailPoint.	Default: 5000
STATIST_INTRVL	During aggregation: Number of entities or connections received by Connector for RACF from SailPoint, after which a message is sent to the Connector for RACF log and an event is sent to SailPoint.	Default: 20
STATUS_INTERVAL	When a Managed System is not active, interval at which Connector for RACF checks the status of the Managed System. When the Managed System is active, an event is sent to SailPoint.	Format: `hhmmss` Default: 000500 (5 minutes)
STOP_REQ_MSGS	During aggregation: Number of entity/connection operations handled by Connector for RACF, after which an "active" confirmation message is sent to SailPoint.	Default: 10
WAIT_LOCK	Wait Lock (seconds)	Default: 60
WAIT_QUEUE	Wait Queue (seconds)	Default: 60

Managed System-Specific Parameters

The following table contain descriptions of parameters which are specified separately for each individual Managed System managed by the Connector for RACF installation. The name of the Managed System must appear before the parameter name in the record.

Parameters	Description	Values
ADMIN_CASE_SENS*	Whether the Administrator name is case-sensitive.	Y, N Default: Y
ADMIN_USER_REQ*	Whether a default administrator is used.	Y, N Default: N
CUSTOM_FIELDS_SUPPORT	Whether product supports RACF cus-	Y, N

Parameters	Description	Values
	tom fields feature.	Default: N
DEFAULT_ADMIN	Name of Connector for RACF default administrator account, which is used for GET operations. Only applicable when ADMIN_USER_REQ=Y.	
DEFAULT_CD_ADMIN	Name of default administrator for the CD process. If special administrator for CD process is not required, then DEFAULT _ADMIN is used. Only applicable when ADMIN_USER_REQ=Y.	
DEFAULT_CS_ADMIN	Name of default administrator for the CS process. If special administrator for CS process is not required, then DEFAULT _ADMIN is used. Only applicable when ADMIN_USER_REQ=Y.	
DEFAULT_OFLI_ADMIN	Default administrator for the Offline Interceptor process. If special administrator for Offline Interceptor process is not required, then DEFAULT _ADMIN is used. Only applicable when ADMIN_USER_REQ=Y.	
DELETE_INTERCEPT_CHECK	When a delete event is detected, whether the Notification server should call a `get` function to check that an object was actually deleted on the Connector for RACF platform. If the call determines that the object	Y, N Default: N

Parameters	Description	Values
	exists, an update event is sent to IdentityIQ instead of a delete event.	
LOG_GET_MSG	Filtering option for messages generated by the synchronization action-Managed System Retrieval Transaction). The messages are recorded in the Transaction Server (CS) log file.	ALL: All Sync messages are written to the CS log file. NONE: No Sync messages are written to the CS log file. Default: ALL
LOGIN_INTERCEPT	Determines whether TSO LOGON events are intercepted by Online and Offline Interceptors.	Y, N Default: Y
LOCKED_ACCOUNT_CFNAME	Alternative custom field name to be used for Locked Accounts support, instead of CTSLKACT.	Suffix of the RACF custom field name.
MAX_Q_TRY	When the Queue file is full, the Offline or the Online Interceptor may retry to write to the Queue file according to the value set in this parameter.	Value for number of retries. If this parameter is not set the default is to keep retrying without any limit.
LOG_INTERCEPT_MSG	Types of interception messages to be recorded in the Notification Server (CD) log file.	ALL: All messages are recorded, indicating in each case whether the event was sent to IdentityIQ coded. ACCEPTED: Only Managed System data included in the aggregation (and therefore sent to IdentityIQ) is recorded. IGNORED: Only Managed System data not included in the aggregation (and therefore not sent to IdentityIQ) is recorded. NONE: No messages are recorded. Default: ALL
OFLI_INTERCEPT	Whether the Offline Interceptor is star-	Y: The Offline Interceptor is

Parameters	Description	Values
	ted automatically by the Notification Server.	started periodically by the Notification server. N: The Offline Interceptor is not started by the Notification server. You must provide another means of scheduling the Offline Interceptor. Default: Y
OFLI_INTERVAL	Minimum interval between consecutive activations of the Offline Interceptor.	Value in the format `hhmmss` Default: 010000
ONLI_EVENT_USER_PWD_ONLY	This parameter controls whether user and group events are intercepted and sent by Online Interceptor. By default it is set to N, meaning that all users, groups, connections and password events are sent to SailPoint. When set to Y, only the password events are sent to SailPoint. In this case the password violation events controlled by SEND_PASS_VIOLATION parameter are not sent to SailPoint.	Y, N
PASSWORD_EVENT_FILTER	This parameter makes it possible to filter all password_change events and user_update events. Filtering is done based on jobname or prefix of jobname which we want to filter its commands.	To filter specific jobname specify full jobname. To filter multiple jobnames with same prefix, specify jobname prefix with '*' adjacent to it. For example, if you want to filter all commands issued by jobnames starting with ABC, specify: ABC* .
RSS_TYPE	Type of Managed System.	RACF
RSS_WORK_DIR	The prefix used to dynamically allocate	Must conform to MVS dataset

Parameters	Description	Values
	working datasets.	naming conventions.
SCRIPT_DIR	The name of a dataset containing customer scripts.	
SCRIPT_SEP_ENTRY	Separator entry value for list fields passed to scripts.	Default: comma (,)
SCRIPT_SEP_FIELD	Separator field value for list fields passed to the scripts.	Default: semicolon (;)
SEND_PWD_TO_SCRIPT	Whether the Managed System Administrator password or phrase is sent to scripts.	Y, N Default: N
SEND_RSSPRM_TO_SCRIPT	Whether to send RSSPARM parameters to scripts.	Y, N Default: Y
SYNC_SEMAPHORE	Name of lock obtained while the Offline Interceptor or aggregation is running (in order to avoid concurrent execution).	
ONLI_EVENT_GROUP	Whether intercepted group events are sent to SailPoint	Y, N Default: Y
ONLI_EVENT_USER	Whether intercepted user events are sent to SailPoint.	Y, N Default: Y
ONLI_EVENT_USER_PWD_ONLY	(Only relevant when ONLI_EVENT_USER = Y) Type of intercepted user events sent to SailPoint.	Y: Only user password change events are sent. N: All user events are sent. Default: N
ONLI_MAX_EVENTS	Number of events which may be active in memory, when Queue file is full. Each entry holds 2,560 bytes in memory, depending on the event type (user, group, connection, password) and length of userid, group, password (above 16M line). A queue full situation might occur when Online Interceptor is active and CD component of the Connector for RACF	ValuesDefault: 20000

Parameters	Description	Values
	is down or reads events from the Queue too slowly comparing to the written events by the Online Interceptor. When queue full situation occurs, Online Interceptor continues accepting events from the SMF exit and from the password exits and these events are accumulated in memory, until queue full situation is relieved. The number of events which can be accumulated in memory is determined by this parameter. ONLI_MIN_NOTIFY_EVENT% Percent number of residual place for new events left in memory that below it, CTS4509W message will be sent by Online Interceptor, each time it handles a new event. The 100% is ONLI_MAX_EVENTS which its default is 20,000. When CTS4509W message is sent, Online Interceptor is able to handle only ONLI_MIN_NOTIFY_EVENT% more events from SMF exit and password exits. If this situation is not relieved fast enough, new events may get lost and	
ONLI_SEMAPHORE	Name of lock obtained while the Online Interceptor is running (in order to avoid concurrent execution). Since this is the ENQ's RNAME (the QNAME is taken from CTSPUSR), this field does not have to be unique.	Default value as set in RSSPARM: `CTSAONLI`
USAAPI_LIB_NAME	The name of an optional DD statement	

Parameters	Description	Values
	in the CS and CD STC procedure. This parameter is used for customized Connector for RACF that are called from a non-APF load library.	
WAIT_WHEN_Q_EOF	When the Queue file is full, the Offline or the Online Interceptor may retry to write to the Queue file according to the value set in the MAX_Q_TRY parameter. The WAIT_WHEN_Q_EOF parameter sets the number of seconds to wait between retries.	Number of seconds to wait.

The following table contains descriptions of RSSPARM parameters which only appear in Connector for RACF.

Parameter	Description	Values
CTSA_ID (ALL_RSS)	An ID that uniquely identifies the Connector for RACF installation.	Default: CTSA
HANDLE_ABENDS (ALL_RSS)	Determines the behavior of Connector for RACF function CTSAPITerm if Connector for RACF abends. Relevant for custom Connector for RACFs developed using the SDK for OS/390. The ability to call the CTSAPITerm function if Connector for RACF abends is useful when the Connector for RACF includes databases and files that should be properly disconnected.	Y: The recovery routines from the function **CTSAPITerm** are invoked. N: The standard SAS-C abend handler is invoked. The SAS-C abend handler only provides DUMP and diagnostic information and does not perform application-specific recovery. Default: N
MAX_SCRIPT_NOTIFY (ALL_RSS)	Number of entries in the Script Notify Buffer.	Default: 250
OFLI_STCNAME	Offline Interceptor started task name.	Default: %PROCPREFS%AOFI

Parameter	Description	Values
ONLI_ACSJBN	Name of the Transaction Server reported to the Online Interceptor.	Default: %PROCPREFS%ACS
ONLI_DETAIL_MSGS	Whether to retrieve detailed messages from the Online Interceptor.	Y, N Default: N
ONLI_DYNAM_PWX01	Whether the Online Interceptor will dynamically install ICHPWX01 when it is started.	Y, N Default: N
ONLI_DYNAM_RIX02	Whether the Online Interceptor will dynamically install ICHRIX02 when it is started.	Y, N Default: N
VERIFY_PASSWORD_BY_LOGIN	Whether to perform password or phrase verification using the RACROUTE macro, by logging on to the RACF Security system.	Y,N Default: N
EXTERNAL_RESOURCE_P REFIX	Prefix of the external resource type. If the Resource type prefix is identical to the value of the RSSPARM parameter EXTERNAL_RESOURCE_PREFIX, the RACF Connector calls user-defined external GET scripts. For more information, see Writing a Script.	Maximum Characters: 5
ONLI_PASSWORD_CASE	Case in which the Online Interceptor sends the password of the administrator (who has performed a password update) to IdentityIQ.	LOWER: Send in lowercase ASIS: Send as received (with no translation) UPPER: Same as ASIS Default: LOWER
ONLI_PASSWORD_FILTER	Whether to send password updates to IdentityIQ.	SUPPRESS: password updates are not sent to IdentityIQ. FORWARD: password updates are sent to IdentityIQ.

Parameter	Description	Values
		Default: FORWARD
RACF_DATASET_QUOTES	Whether the Connector for RACF will add quotes around dataset names for Managed System mscs_name.	Y, N
RCF_DELAY_DELGRP	Whether the SailPoint Delay parameter for delete group requests is supported.	Y, N
RCF_DELAY_DELUSR	Whether the SailPoint **Delay** parameter for delete user requests is supported. If the **Delay** parameter is specified in a delete request from SailPoint, but not supported in Connector for RACF, the delete request fails and an error message is returned to SailPoint.	Y, N
RCF_EXPIRE_PASSCHG	Whether an Managed System user defined by SailPoint is assigned a one-time password or phrase.	Y: The user is assigned a one-time password or phrase and must change the password or phrase upon logging in to the Managed System for the first time. N: The user is assigned a regular password or phrase which can be changed at any time. Default: N
RCF_RESUME_BEFORE_P ASSCHG	Whether an ALTUSER <userid> RESUME command is issued before a user password or phrase update command is sent. When the RACF INACTIVE option is set, inactive users will be reset by this command (last active field is updated).	Y, N Default: N

Parameter	Description	Values
	Note This option will resume revoked userids whenever SailPoint issues update password or phrase transactions.	
RCF_UNIGROUP_MAX	Maximum number of users that can be connected to a Universal group.	Number up to 999999. Default: 0
USERS_TO_FILTER	For Online Interceptor, userid filtering is enabled based on first letter of the userid. Set this parameter to filter users with first letters. For example, set as follows to filter all events of all users which their first letter P, Q, or R: `<rssname> USERS_TO_FILTER PQR`	

CTSPARM – Assembler Format Parameters

This CTSPARM member in PARM library includes few parameters in an assembler source format. This means that, once a parameter is updated, the member must be saved and then it must be compiled and linked. This is done with the CTSPARMJ member located in the INSTALL library. So once CTSPARM is updated, CTSPARMJ must be submitted.

Parameter	Description
ENQRNL	When global resource serialization (GRS), (for example, ENQ or DEQ) encounters a request for a resource with a scope of SYSTEMS, it scans the SYSTEMS exclusion resource name list (RNL) to determine the scope of the requested resource. However, if the request specifies RNL=NO, GRS would not scan the SYSTEMS exclusion RNL and bypass the RNL search. By default, the CTSPARM ENQRNL parameter is set to **Y**. If GRS environment requires that resource requests bypass the RNL search, set ENQRNL to **N**.
QNAME	Unique name used to protect access to the Queue file by Interceptors and Notification

Parameter	Description
	Server. The default QNAME is xxxASYNC where xxx is the value set for %PROCPREFS% in the DEFPARMS member in the INSTALL library.

RSSAPI – Connector Entries and Scripts

Each line in this member (excluding the comment lines) represents a Connector call which is followed by various parameters. The parameters influence the behavior of the Connector.

For a description of the parameters in each line, see the **Standard Managed System User Fields** table in User Data Translation Tables.

Pre-scripts and Post-scripts must be stored in the PDS library which was set in parameter SCRIPT_DIR of member RSSPARM. The default value for this parameter is:

```
prefix.version.USER.CLIST
```

where:

- *prefix* - Value set for parameter OLPREFS in member INSTALLD in the Connector INSTALL library
- *version* -Value set for parameter OLVERS in member INSTALLD in the Connector INSTALL library.

Appendix C: Connector for RACF Datasets and JCL Procedures

This appendix lists details for RACF datasets and JCL procedures as they pertain to your connector.

Connector for RACF Dataset List

Connector for RACF Installation Datasets

The following datasets are allocated during the Connector for RACF installation procedure. In the names of these datasets, **<prefix>** is the value specified for parameter **ILPREFS**, and **<version>** is the value specified for parameter **ILVERS**. Both parameters are located in LOADCTS member in the Connector INSTALL library. These parameters are set in the installation step 4.1 – Tailor the LOADCTS Member using values set in Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs).

Dataset	Description
<prefix>.<version>.CLIST	REXX library
<prefix>.<version>.CTRANS	SAS/C Runtime Load library
<prefix>.<version>.CMSG	Message text library
<prefix>.<version>.INSTALL	Installation library
<prefix>.<version>. ISMSGENG	ISPF English messages of CTSGATE ISMSGENG
<prefix>.<version>.JCL	Sample jobs library
<prefix>.<version>.LOAD	Load library
<prefix>.<version>.LOADE	SSL load modules LOADE
<prefix>.<version>.MAC	Macros library
<prefix>.<version>.MSG	Message text library
<prefix>.<version>. MSGENG	English messages of CTSGATE MSGENG
<prefix>.<version>.PARM	Parameters library
<prefix>.<version>.PANELENG	English panels of CTSGATE PANELENG
<prefix>.<version>.PROCLIB	JCL procedures library
<prefix>.<version>.SAMPLE	Sample REXX and REXX library
<prefix>.<version>.SECSRC	Sample security exits for CTSGATE (not used)
<prefix>.<version>.UPGRADE	UPGRADE jobs

Operation Datasets

The following datasets are allocated during the Connector for RACF installation procedure. In the names of these datasets, **<prefix>** is the value specified for parameter **OLPREFS**, and **<version>** is the value specified for parameter **OLVERS**. Both parameters are located in member LOADCTS in the Connector INSTALL library. These parameters are set in the installation step 4.1 – Tailor the LOADCTS Member using values set in Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs).

Dataset	Description
<prefix>.<version>.CARECNN	Managed user to group connections
<prefix>.<version>.CAREGRP	Managed groups
<prefix>.<version>.CAREOE	Managed organization elements
<prefix>.<version>.CAREUSR	Managed users
<prefix>.<version>.DIAGLVL	Diagnostics setup
<prefix>.<version>.ENCREXT	Transmitted Data Encryption
<prefix>.<version>.ENCRINT	Stored Data Encryption
<prefix>.<version>.QUEUE	Interception Queue dataset
<prefix>.<version>.RCFDELRQ	Delayed delete request
<prefix>.<version>.RSSKWDS	Managed System specific keywords table
<prefix>.<version>.RSSOFLI	Offline interceptor table
<prefix>.<version>.USER.CLIST	Scripts dataset

SMP/E Distribution Datasets

The following datasets are allocated during the Connector for RACF installation procedure. In the names of these datasets, **<prefix>** is the value specified for parameter **SPDPREF**, and **<version>** is the value specified for parameter **SPDVER**. Both parameters are located in member LOADCTS in the Connector INSTALL library. These parameters are set in the installation step 4.1 – Tailor the LOADCTS Member using values set in Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs).

Dataset	Description
<prefix>.<version>.ACMSG	CMSG Dlib
<prefix>.<version>.ACLIST	REXX Dlib ACLIST
<prefix>.<version>.AINSTALL	INSTALL Dlib
<prefix>.<version>.AISMSGEN	ISMSGENG Dlib AISMSGEN
<prefix>.<version>.AIOALOAD	GATEWAY LOAD Dlib
<prefix>.<version>.AJCL	JCL Dlib

Dataset	Description
<prefix>.<version>.ALOADE	SSL Load modules Dlib ALOADE
<prefix>.<version>.AMAC	Macro Dlib
<prefix>.<version>.AMSG	MSG Dlib
<prefix>.<version>.AMSGENG	English messages of CTSGATE Dlib AMSGENG
<prefix>.<version>.APARM	PARM Dlib
<prefix>.<version>.APROCLIB	PROCLIB Dlib
<prefix>.<version>.APANELEN	PANELENG Dlib APANELEN
<prefix>.<version>.ASAMPLE	SAMPLE Dlib
<prefix>.<version>.ASECSRC	SECSRC Dlib ASECSRC
<prefix>.<version>.AUPGRADE	UPGRADE Dlib AUPGRADE

SMP/E Datasets

The following datasets are allocated during the Connector for RACF installation procedure. In the names of these datasets, **<prefix>** is the value specified for parameter **SPAPREF**, and **<version>** is the value specified for parameter **SPAVER**. Both parameters are located in member LOADCTS in the Connector INSTALL library. These parameters are set in the installation step 4.1 – Tailor the LOADCTS Member using values set in Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs).

Dataset	Description
<prefix>.<version>.SMPLOG	SMP/E Work dataset
<prefix>.<version>.SMPLOGA	SMP/E Work dataset
<prefix>.<version>.SMPLTS	SMP/E Work dataset
<prefix>.<version>.SMPMTS	SMP/E Work dataset
<prefix>.<version>.SMPPTS	SMP/E Work dataset
<prefix>.<version>.SMPSCDS	SMP/E Work dataset
<prefix>.<version>.SMPSTS	SMP/E Work dataset

SMP/E CSI

The following dataset is allocated during the Connector for RACF installation procedure. In the names of this dataset, **<prefix>** is the value specified for parameter **SPCPREF**, and **<version>** is the value specified for parameter **SPCVER**. Both parameters are located in member LOADCTS in the Connector INSTALL library. These parameters are set in the installation step 4.1 – Tailor the LOADCTS Member using values set in Allocate and Load Connector for RACF Datasets ($LOADINS and LOADCTS Jobs).

Dataset	Description
<prefix>.<version>.CSI	SMP/E CSI

Connector for RACF JCL procedures

The following JCL procedures are copied to your system PROCLIB library during the Connector for RACF installation procedure.

JCL procedure	Description
CTSAADPT	Connector for RACF Managed System rename utility
CTSACD	Connector Notification Server (CD)
CTSACS	Connector Transaction Server (CS)
CTSADFR	Connector Offline Interceptor file formatting utility
CTSALERT	Connector for RACF alert data utility
CTSAONI	RACF Online Interceptor
CTSAQCR	Connector for RACF copy and format Queue dataset utility
CTSAQFR	Connector for RACF Queue formatting utility
CTSAQPR	Connector for RACF Queue printing utility
CTSASMP	Connector for RACF SMP/E procedure
CTSCDIAG	Connector for RACF diagnostics file formatting utility
CTSDIAG1	Connector for RACF diagnostic initialization utility
CTSGATE	Connector for RACF Gateway
CTSKGEN	Connector for RACF Stored Data Encryption utility
CTSAOFI	Offline Interceptor
CTSUPMSG	Connector for RACF message update table
CTSRRSF	Connector for RACF Online Interceptor for RRSF
CTSOFLI	Connector for RACF Common Offline Interceptor
CTSC100	Delayed Delete procedure
CTSACDJ	STCJOB of Connector Notification Server (CD)
CTSACSJ	STCJOB of Connector Transaction Server (CS)
CTSAOFIJ	STCJOB of RACF Offline Interceptor
CTSAONIJ	STCJOB of RACF Online Interceptor
CTSGATEJ	STCJOB of RACF Gateway Connector Monitor
CTSC100J	STCJOB of RACF Delayed Delete procedure

Appendix D: Copying a Connector for RACF installation

This appendix describes the procedure to copy an existing installation of Connector for RACF from one MVS system to another.

The following terms are used throughout this appendix:

- **source system** – MVS system where Connector for RACF was originally installed.
- **target system** – MVS system to which Connector for RACF installation is copied.

Installation Copy Procedure

Use the procedure that follows to copy an existing installation of Connector for RACF from one MVS system to another.

> **Note**
> If LOCALCOPY value is set for the %PROCLIB% parameter in the DEFPARMS member in the Connector INSTALL library, the same procedures will be used by the source and new Connector for RACF environments. If the Managed System name has to be changed (see Installation Copy Procedure below), contact SailPoint Customer Support for instructions.

1 – Copy Connector for RACF JCL Procedures

During installation of Connector for RACF, various JCL procedures were copied to the system procedures library. The JCL procedures are used by Connector for RACF started tasks and by jobs activating the Connector for RACF utilities.

These JCL procedures must be copied to the target system.

The procedures may be copied manually or using the instructions specified in installation Step 5 – Tailor Connector for RACF Members with Site Parameters.

1A – Connector for RACF Started Tasks

The following JCL procedures are Connector for RACF started tasks and must be copied to the target system JCL procedures library:

- CTSACD
- CTSAOFI

- CTSACS
- CTSAOFS
- CTSGATE
- CTSAONI
- CTSC100

When STCJOBs are used:

- The corresponding STCJOB members must be copied to the target system STCJOBS library.
- If LOCALCOPY value is set for the %PROCLIB% parameter in DEFPARMS member in the Connector INSTALL library, make sure the target system has access to the Connector for RACF PROCLIB library.

1B – Connector for RACF Utilities

The following JCL procedures are used by Connector for RACF utilities and maintenance jobs. They are not required for the daily operation of Connector for RACF. However, they are required for customization and installation operations performed in subsequent steps described in this procedure.

- CTSADFR
- CTSDIAG
- CTSAQFR
- CTSDIAGI
- CTSAQCR
- CTSAQPR
- CTSKGEN
- CTSALERT
- CTSUPMSG
- CTSAADPT

2 – Copy Connector for RACF Datasets

The following parameters are used in this step:

- **<i_prefix>** – Value set for parameter ILPREFS in member LOADCTS in the Connector INSTALL library. Default: CTSA.
- **<i_version>** – Value set for parameter ILVERS in member LOADCTS in the Connector INSTALL library. Default: V400.
- **<o_prefix>** – Value set for parameter OLPREFS in member LOADCTS in the Connector INSTALL library. Default: CTSA.
- **<o_version>** – Value set for parameter OLVERS in member LOADCTS in the Connector INSTALL library. Default: V400.

For more information regarding parameters in member LOADCTS, see step 4.1 – Tailor the LOADCTS Member.

2A – Operations datasets

The following datasets are used during the operation of Connector for RACF and must be copied from the source system to the target system:

```
<i_prefix>.<i_version>.CLIST
<i_prefix>.<i_version>.CMSG
<i_prefix>.<i_version>.LOAD
<i_prefix>.<i_version>.MSG
<i_prefix>.<i_version>.PARM
<i-prefix>.<i-version>.CTRANS
```

2B – Installation datasets

The following datasets are not used by Connector for RACF started tasks and utilities but are required for customization and installation operations performed later in this procedure:

```
<i_prefix>.<i_version>.INSTALL
<i_prefix>.<i_version>.MAC
<i_prefix>.<i_version>.JCL
<i_prefix>.<i_version>.SAMPLE
<o_prefix>.<o_version>.USER.CLIST
<i_prefix>.<i_version>.PROCLIB
```

3 – Adjust Connector for RACF parameters

Perform the customization steps described below on the target system.

> **Note**
> As an alternative to steps 3A and 3B below, you can use utility CTSAADPT to rename the Managed System on the target system. For more information, see Renaming a Managed System.

3A – RSSPARM Parameters

Edit member RSSPARM member in the Connector PARM library.

The RSSPARM member contains the Managed System parameters for the installed Managed System.

Each line is in the format:

```
<Managed System-name> <parameter_name> <parameter_value>
```

where `<Managed System-name>` is either ALL_RSS or the Managed System name specified during installation.

If the Managed System name defined in SailPoint for the target system is different from the one specified in the member RSSPARM, you must update member RSSPARM to reflect that difference.

To perform the required update in member RSSPARM, change the `<Managed System-name>` value on all lines that specify the source system Managed System name to the new Managed System name that is correct for the target system.

In addition, the Managed System name qualifier in the value specified for parameter RSS_WORK_DIR must be modified.

> **Note**
> The Managed System name specified in the RSSPARM parameters file must match the name defined in SailPoint for the target system. If the names do not match, SailPoint will not be able to connect to Connector for RACF.

3B – RSSAPI Parameters

Edit member RSSAPI member in Connector PARM library.

RSSAPI member contains the script activation definitions for the Managed System.

Each line is in the format:

```
RSS_type RSS_name additional_parameters
```

where:

- RSS_type – hyphen (-)

- RSS_name – Managed System name specified during installation.
  Default: MVSRACF

The complete syntax of this member is described under Structure of the RSSAPI Member.

If the Managed System name defined in SailPoint for the target system is different from the one specified in member RSSAPI, you must update member RSSAPI to reflect that difference.

To perform the required update in member RSSAPI, change the **RSS_name** parameter on all the lines containing the source system Managed System name to the new Managed System name that is correct for the target system.

3C – Adjust Procedures

If the Managed System name was changed, modify the `RSS=` parameter in each of the following procedures:

- CTSAADPT

- CTSADFR

- CTSAOFI

- CTSOFLI

- CTSAOFS

- CTSAONI

- CTSC100

3D – Allocate Connector for RACF Work Datasets

The datasets listed below are created during Connector for RACF installation. These datasets should *not* be copied from the source system to the target system. Instead, they should be allocated and formatted directly on the target system.

Originally, allocation of these datasets was performed by job FORMCTS, which is run in installation step Step 8 – Format Connector for RACF Datasets.

You may run job FORMCTS in the Connector INSTALL Library to allocate the datasets on the target system.

```
<o_prefix>.<o_version>.CARECNN
```

```
<o_prefix>.<o_version>.CAREGRP
```

```
<o_prefix>.<o_version>.CAREUSR
```

```
<o_prefix>.<o_version>.DIAGLVL
```

```
<o_prefix>.<o_version>.QUEUE
<o_prefix>.<o_version>.RCFDELRQ
<o_prefix>.<o_version>.RSSKWDS
<o_prefix>.<o_version>.RSSOFLI
<o_prefix>.<o_version>.ENCRINT
<o_prefix>.<o_version>.ENCREXT
```

3E – Encryption Datasets

The following datasets are used during the operation of Connector for RACF and must be copied from the source system to the target system:

```
<o_prefix>.<o_version>.ENCREXT
<o_prefix>.<o_version>.ENCRINT
```

These datasets are allocated in step 3D – Allocate Connector for RACF work datasets and must be overwritten in this step.

4 – Adjust MVS System parameters

4A – APF Authorized Library

The Connector LOAD and CTRANS libraries must be defined as APF authorized libraries in the target system. For more information, see procedure Step 9 – Customize Communication Settings.

4B – Security Rules Definition

As part of Connector for RACF installation, several RACF definitions were setup for Connector for RACF datasets and started tasks. For more information, see Step 10 – Define Connector for RACF in RACF.

Apply the same definitions to the target system.

5 – Customize RACF Support

The customization process required for completing Connector for RACF installation is described in RACF Support Customization.

The process includes the installation of the SMF IEFU83 exit that is required for interception of racf database security administration events.

To install support for this capability for the target system, perform the necessary instructions as described in the above chapter.

Appendix E: Managed System-Specific Fields

This appendix provides reference tables for Managed System-specific fields.

Description of Table Column Titles

Due to the many columns of information contained in the tables in this appendix, abbreviated column names are used. This section describes the meaning of the column titles for the Managed System-specific field tables later in the appendix.

The columns described in the following table appear in all the Managed System-specific Field tables in this appendix.

Column Title	Description
Field	Field name (as it appears in the Details window of SailPoint). For list fields, the subfields are indented. By default, field labels are displayed in a Details window. To view field names, click the right mouse button anywhere in the Details window in SailPoint (except on a field) and choose the option **Show Field Names** from the pop-up menu. The field names are displayed instead of the field labels.
L	Whether or not the field accepts a list of values. A list consists of values separated by commas. Possible values in this column are: • L – Identifies a list field. • S – Identifies a subfield of a list field (names of subfields are indented in the **Field** column).
T	Type of input accepted in the field. Possible values in this column are: • C – Character. All input is treated as characters even if all are digits.

Column Title	Description
	- F – Flag. Input must be **Y** or **N**. - N – Integer. Input must be numeric. - T – Time. Input must be in the time format specified in the column **Restrictions** - D – Date/Time. Input must be in the format specified in the column **Restrictions**. This format generally requires that the value be specified as a string consisting of the date or date/time. - S – Selection from a list of predefined values.
Len	Maximum number of characters in a character field. This field length only applies if the type (column **T**) is **C**. (Field length limitations for other data types are determined by information in columns **L** and **Restrictions**.)
Restrictions	Validation restrictions such as numeric ranges or list of possible values. Underlined values (if any) are the default values.

The column titles for each type of entity differ slightly. The following table describes the meaning of the single-letter column titles used to indicate the type of function for which each Managed System-specific field is relevant.

An **X** appearing in a column for a given field indicates that the field is relevant to that function.

Function Type	Column Title	Description
User	A	Add user
	U	Update user
	G	Get user
	D	Delete user
	R	Revoke/restore user
	P	Update password
Group	A	Add group
	U	Update group
	G	Get group
	D	Delete group

Function Type	Column Title	Description
User–Group Connection	C	Connect user to group
	U	Update user to group connection
	G	Get user to group connection
	D	Disconnect user from group

Managed System User Fields

The following table lists managed system user fields.

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
ADSP		F	1			X	X	X			
CATEGORY	L	C	230-3			X	X	X			
CLAUTH	L	C	230-3			X	X	X			
DATA		C	255			X	X	X			
GRPACC		F	1			X	X	X			
MODEL		C	44	dsname		X	X	X			
NAME		C	20			X	X	X			
OIDCARD		F	1			X	X	X			
OPERATIONS		F	1			X	X	X			
OWNER		C	8		X	X	X	X			
RACF_REVOKE_REASON		C	20	password or phrase, COMMAND, INACTIVITY, DATE, UNKNOWN				X			
RESUME_DATE		N		YYYYMMDD		X	X	X	X		
REVOKE_DATE		N		YYYYMMDD		X	X	X	X		
REVOKED		F	1		X	X	X	X			
RU_LOCKED		F	1		X	X	X	X			
RU_SUSPENDED		F	1		X	X	X	X			

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
SECLABEL		C	8			X	X	X			
SECLEVEL		C	8			X	X	X			
UAUDIT		F	1			X	X	X			
WHEN.DAYS	L	S	80	ANYDAY, WEEKDAYS, SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY	X	X	X	X			
WHEN.TIME		N	4	ANYTIME | hhmm:hhmm	X	X	X	X			
INFO.INTERVAL		N	3	0-255				X			
INFO.CREATE_DATE		N		YYYYMMDD				X			
INFO.LAST_LOGIN_DATE		N		YYYYMMDD				X			
INFO.LAST_LOGIN_TIME		N		HHMMSS				X			
INFO.PASSCHG_DATE		N		YYYYMMDD				X			
INFO.UNKNOWNCAT		N						X			
CICS.OPCLASS	L	N	256	1-24		X	X	X			
CICS.OPIDENT		C	3			X	X	X			
CICS.OPPRTY		N		0-255		X	X	X			
CICS.TIMEOUT		N		0-60		X	X	X			
CICS.XRFSOFF		S	7	FORCE | NOFORCE		X	X	X			
DFP.DATAAPPL		C	8			X	X	X			
DFP.DATACLAS		C	8			X	X	X			

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
DFP.MGMTCLAS		C	8			X	X	X			
DFP.STORCLAS		C	8			X	X	X			
LANGUAGE.PRIMARY		C	24			X	X	X			
LANGUAGE.SECONDARY		C	24			X	X	X			
OPERPARM.ALTGRP		C	8			X	X	X			
OPERPARM.AUTH	L	S	30	MASTER, ALL, INFO, CONS, IO, SYS		X	X	X			
OPERPARM.AUTO		S	3	YES | NO		X	X	X			
OPERPARM.CMDSYS		C	8			X	X	X			
OPERPARM.DOM		S	6	NORMAL | ALL | NONE		X	X	X			
OPERPARM.KEY		C	8			X	X	X			
OPERPARM.LEVEL	L	S	20	NB, R, I, CE, E, IN, ALL		X	X	X			
OPERPARM.LOGCMDRESP		S	6	SYSTEM | NO		X	X	X			
OPERPARM.MFORM	L	S	10	J,M,S,T,X		X	X	X			
OPERPARM.MIGID		F	3			X	X	X			
OPERPARM.MONITOR	L		30	JOBNAMES | JOBNAMEST, SESS | SESST, STATUS		X	X	X			
OPERPARM.MSCOPE	L		50	ALL, *		X	X	X			
OPERPARM.ROUTCODE	L		30	ALL, NONE, 1-128 | xxx:yyy		X	X	X			
OPERPARM.STORAGE		N		1 - 2000		X	X	X			
OPERPARM.UD		S	3	YES | NO		X	X	X			

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
TSO.ACCTNUM		C	39			X	X	X			
TSO.COMMAND		C	80			X	X	X			
TSO.DEST		C	8			X	X	X			
TSO.HOLDCLASS		C	1			X	X	X			
TSO.JOBCLASS		C	1			X	X	X			
TSO.MAXSIZE		N		0-2096128		X	X	X			
TSO.MSGCLASS		C	1			X	X	X			
TSO.PROC		C	8			X	X	X			
TSO.SECLABEL		C	8			X	X	X			
TSO.SIZE		N		0-2096128		X	X	X			
TSO.SYSOUTCLASS		C	1			X	X	X			
TSO.UNIT		C	8			X	X	X			
TSO.USERDATA		C	4			X	X	X			
WORKATTR.WAACNT		C	255			X	X	X			
WORKATTR.WAADDR-1		C	60			X	X	X			
WORKATTR.WAADDR-2		C	60			X	X	X			
WORKATTR.WAADDR-3		C	60			X	X	X			
WORKATTR.WAADDR-4		C	60			X	X	X			
WORKATTR.WABLDG		C	60			X	X	X			
WORKATTR.WADEPT		C	60			X	X	X			
WORKATTR.WANAME		C	60			X	X	X			
WORKATTR.WAROO-M		C	60			X	X	X			
MODE		S	8	DELETE | SCAN | FULL | FORCE	X					X	
DELAY		F	1							X	
REPLACE		C	8							X	
CMDFILE		C	44							X	

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
INTERVAL		C	3	YES | NO | 1-254							X
OMVS.UID		N	7	0-2096128		X	X	X			
OMVS.HOME		C	1023			X	X	X			
OMVS.PROGRAM		C	1023			X	X	X			
NETVIEW.CONSNAME		C	8			X	X	X			
NETVIEW.CTL		S	8	GENERAL, GLOBAL, SPECIFIC		X	X	X			
NETVIEW.DOMAINS	L	C	2305			X	X	X			
NETVIEW.IC		C	255			X	X	X			
NETVIEW.MSGRECVR		S	3	YES | NO		X	X	X			
NETVIEW.NGMFADMN		S	3	YES | NO		X	X	X			
NETVIEW.OPCLASS	L	N	256	1-2040		X	X	X			
NETVIEW.NGMFVSPN		C	255			X	X	X			
DCE.AUTOLOGIN		S	3	YES | NO				X	X	X	
DCE.DCENAME		C	1023					X	X	X	
DCE.HOMECELL		C	1023					X	X	X	
DCE.HOMEUUID		C	36					X	X	X	
DCE.UUID		C	36					X	X	X	
OVM.FSROOT		C	1023					X	X	X	
OVM.HOME		C	1023					X	X	X	
OVM.PROGRAM		C	1023					X	X	X	
OVM.UID		N	10	0-2147483647				X	X	X	

Field	L	T	Len	Restrictions	M	A	U	G	R	D	P
targetPermissions[2]		C	4096					X			
DCERT.PROFILE.#[1]		C						X			
DCERT.CREATE_DATE.#[1]		C						X			
DCERT.OWNER.#[1]		C						X			
DCERT.TRUST.#[1]		C						X			
DCERT.APPLDATA.#[1]		C						X			
DCERT.VALID_NOT_BEFORE.#[1]		C						X			
DCERT.VALID_NOT_AFTER.#[1]		C						X			
DCERT.SUBJECT_NAME.#[1]		C						X			
DCERT.SERIAL_NUMBER.#[1]		C						X			
DCERT.ISSUER_NAME.#[1]		C						X			
DCERTMAP.LABEL.#[1]		C						X			
DCERTMAP.OWNER.#1		C						X			
DCERTMAP.TRUST.#1		C						X			
DCERTMAP.OWNER.#1		C						X			
DCERTMAP.ISSUER_NAME.#[1]		C						X			
DCERTMAP.SUBJECT_NAME.#[1]		C						X			

1. # in DCERT. and DCERTMAP. fields represents numbers starting from 1.

2. All permissions are included within the **targetPermissions** attribute. Full description of targetPermissions content appears in Resource ACL Data Translation Tables.

Group Fields

The following table lists group fields.

Field	L	T	Len	Restrictions	M	A	U	G	D
DATA		C	255			X	X	X	
MODEL		C	44	dsname		X	X	X	
OWNER		C	8		X	X	X	X	
TERMUACC		F	1			X	X	X	
INFO.CREATE_DATE		N		YYYYMMDD				X	
DFP.DATAAPPL		C	8			X	X	X	
DFP.DATACLAS		C	8			X	X	X	
DFP.MGMTCLAS		C	8			X	X	X	
DFP.STORCLAS		C	8			X	X	X	
MODE		S	8	DELETE | SCAN | FULL | FORCE	X				X
DELAY		F	1						X
REPLACE		C	8						X
SUPGRP		C	8						X
CMDFILE		C	44						X
OMVS.GID		N	7	0-2096128		X	X	X	
OVM.GID		N	10	0-2147483647		X	X	X	
UNIVERSAL		F	1			X		X	
targetPermissions[1]		C	4096					X	

[1] All permissions are included within the targetPermissions attribute. Full description of targetPermissions content appears in the resource ACL data translation table in Resource ACL Data Translation Tables.

User-Group Connection Fields

The following table lists user-group connection fields.

Field	L	T	Len	Restrictions	M	A	U	G	D
ADSP		F	1			X	X	X	
AUTHORITY		S	7	USE | CREATE | CONNECT | JOIN	X	X	X	X	
GRPACC		F	1			X	X	X	
OPERATIONS		F	1			X	X	X	
OWNER		C	8		X	X	X	X	
REVOKED		F	1			X	X	X	
RESUME_DATE		N		YYYYMMDD		X	X	X	
REVOKE_DATE		N		YYYYMMDD		X	X	X	
UACC		S	7	ALTER | CONTROL| EXECUTE | UPDATE | READ | NONE	X	X	X	X	
INFO.CREATE_DATE		N		YYYYMMDD				X	
UNIVERSAL		F							

Appendix F: Connector for RACF Batch Utility

The batch utility enables a user to execute a batch job containing provisioning or list requests directly on the RACF Connector without the requirement of a partner (such as SailPoint). The provisioning or list requests are processed by the requested Managed System Interface together with the requested security product, RACF.

The batch utility would be helpful in the following scenarios:

- When a new installation is performed and a connection has not yet been established with SailPoint.

  Provisioning or list transactions can be issued locally within the Mainframe using the batch utility to ensure that the RACF Connector is installed and working properly.

- For testing or debugging purposes as instructed by SailPoint Support.

- When multiple provisioning transactions are required to be performed quickly and easily from the Mainframe than from SailPoint.

Whenever provisioning transactions are issued by the batch utility and if the Online Interceptor is not active, it is required to issue full aggregation in order to update SailPoint with the changes done by these provisioning transactions.

This appendix describes the syntax rules, provisioning and list requests, and invocation JCL required to execute the utility and an example of utility control statements with the sample job output.

Security Requirements

Before running the utility, ensure that the following security requirements are met:

- When invoking the batch utility with provisioning requests, the user specified as ADMIN_UNAME must have sufficient authority in order to execute the requests.

- If **STCJOBS** are used for the product started tasks, the file allocated for EXECOUT DD statement is a permanent file.

- The user submitting the job must have ALTER authority to this file.
- The user specified as ADMIN_UNAME must have UPDATE authority to this file. The file name is:

```
<olprefs>.<olvers>.EXECOUT.<procprefs>BATCH
```

where olprefs, olvers and procprefs are the values set in Step 1 – Set the Parameter Values.

Input Control Statements Syntax Rules

The following rules must be followed for the utility control statements:

- Must contain environment and provisioning / list request lines
- Data must be written in columns 1 to 72.
- The file must not have sequence numbers, that is, number mode off or unnum.
- A comment line must begin with an asterisk in column 1. The Input Control Statements can have many comment lines. Comments must not be inserted in a non-comment line.
- The input for the utility must begin with environment lines as defined in Environment Definition Syntax.
- The input for the utility may contain multiple provisioning / list requests. Each request begins with a request line which defines the request and is followed by one or more parameter lines which provides request details.
- The input control statements – keywords and values – should not be converted to upper or lowercase. All data must be specified in the correct case.
- If a value is longer than 1 line, it must be surrounded by parentheses and continued in column 1 of the next line.

  For example:

```
Keyword=(value1,value2,value3,value4,value5,value6,
value7,value8,value9,value10,value11,value12,value13)
```

Environment Definition Syntax

The utility input must begin with environment definition lines.

:ENV

Parameter	Mandatory/Optional	Description
RSS_TYPE	Mandatory	RACF
RSS_NAME	Mandatory	The name defined for RSSNAME in the DEFPARMS member

Parameter	Mandatory/Optional	Description
ADMIN_UNAME	Mandatory	The administrator User ID
ADMIN_GROUP	Optional	The group to be used for the administrator. **Note** Used to override the default group used during logon processing.

Batch Provisioning and List Requests

This section describes the following provisioning and list requests:

- **ADDUSER** – Add a User
- **UPDUSER** – Update a User
- **DELUSER** – Delete a User
- **DISABLEUSER** – Disable a User
- **ENABLEUSER** – Enable a User
- **LISTUSER** – List User Information
- **CHGPWD** – Change a User password
- **ADDGROUP** – Add a Group
- **UPDGROUP** – Update a Group
- **DELGROUP** – Delete a Group
- **LISTGROUP** – List Group Information
- **ADDCONN** – Add a User-Group Connection
- **UPDCONN** – Update a User-Group Connection
- **DELCONN** – Delete a User-Group Connection

- **LISTCONN** – List User-Group Connection information
- **LISTACL** – List ACL permission information

ADDUSER – Add a User

Use the following request and parameter lines to add a user. None of the parameters lines are mandatory under RACF.

```
:ADDUSER=userid
```

Parameter	Mandatory/Optional	Description/Value
USER.DFLTGRP	Optional	Default group name. `USER.DFLTGRP=default-group-name`
USER.PASSWORD	Optional	User's password `USER.PASSWORD=[(]password[,PERM / TEMP)]`
USER.DISABLE	Optional	`USER.DISABLE=YES`
USER.AUTH	Optional	`USER.AUTH=REG / ADMIN / AUDIT / ADMINAUDIT`

kwd	Optional	The kwd lines represent attributes which are specified in the SailPoint schema.
		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`

UPDUSER – Update a User

Use the following request and parameter lines to update a user.

```
:UPDUSER=userid
```

Parameter	Mandatory/Optional	Description
USER.AUTH	Optional	`USER.AUTH=REG / ADMIN / AUDIT / ADMINAUDIT`
USER.DFLTGRP	Optional	`USER.DFLTGRP= (default-group-name, DROPOLD / KEEPOLD)`
USER.PASSWORD	Optional	`USER.PASSWORD=[(]password[,PERM / TEMP)]`
USER.DISABLE	Optional	`USER.DISABLE=YES`
USER.ENABLE	Optional	`USER.ENABLE=YES`
kwd	Optional	The `kwd` lines represent attributes which are specified in the SailPoint schema.

Parameter	Mandatory/Optional	Description

		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`
		To nullify a keyword, the keyword must be specified with a null value.
		To delete one value from a multi-value keyword field, specify the keyword with all remaining values.

DELUSER – Delete a User

Use the following request and parameter lines to delete a user.

```
:DELUSER=userid
```

Parameter	Mandatory/Optional	Description
MODE	Mandatory	`MODE=REVOKE / DELETE / SCAN / FULL / FORCE`
kwd	Optional	The relevant `kwd` lines are those which affect the delete process. For example, DELAY `kwd=value`

DISABLEUSER – Disable a User

Use this request line to disable a user.

```
:DISABLEUSER=userid
```

ENABLEUSER – Enable a User

Use this request line to enable a user.

```
:ENABLEUSER=userid
```

LISTUSER – List User Information

Use the following request and parameter lines to list user information.

```
:LISTUSER
```

Default parameter setting:

```
FILTER.USERID=*
```

Parameter	Mandatory/Optional	Description
FILTER.USERID	Optional	`FILTER.USERID={* | [(]userid[,userid…][)] | userid-mask}`
USER.GETCONN	Optional	Indicates that the groups keyword would contain all the groups associated with the user. `USER.GETCONN=NO/YES`
kwd	Optional	The specified `kwd` lines are attributes which are retrieved and displayed. If no `kwd` is specified all user related attributes are retrieved and displayed. `kwd`

CHGPWD – Change a User Password

Use the following request and parameter lines to change a user's password.

```
:CHGPWD=userid
```

Parameter	Mandatory/Optional	Description
USER.PASSWORD	Optional	User password `USER.PASSWORD =[(]password [,PERM / TEMP)]`
VERIFY_PWD	Optional	Indicates that the password is to be verified, not changed. `VERIFY_PWD=N / Y`

ADDGROUP – Add a Group

Use the following request and parameter lines to add a group.

```
:ADDGROUP=groupid
```

Parameter	Mandatory/Optional	Description
GROUP.PARENT	Optional	`GROUP.PARENT =parent-group-name`

kwd	Optional	The kwd lines represent attributes which are specified in the SailPoint schema.
		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`

UPDGROUP – Update a Group

Use the following request and parameter lines to update a group.

```
:UPDGROUP=groupid
```

Parameter	Mandatory/Optional	Description
GROUP.PARENT	Optional	`GROUP.PARENT =parent-group-name`

kwd	Optional	The kwd lines represent attributes which are specified in the SailPoint schema.
		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`
		To nullify a keyword, the keyword must be specified with a null value.
		To delete one value from a multi-value keyword field, specify the keyword with all remaining values.

DELGROUP – Delete a Group

Use the following request and parameter lines to delete a group.

```
:DELGROUP=groupid
```

Parameter	Mandatory/Optional	Description
MODE	Mandatory	`MODE=REVOKE / DELETE / SCAN / FULL / FORCE`
kwd	Optional	The relevant `kwd` lines are those which affect the delete process. For example: DELAY `kwd=value`

LISTGROUP – List Group Information

Use the following request and parameter lines to list group of information.

```
:LISTGROUP
```

Default parameter setting:

```
FILTER.GROUPID=*
```

Parameter	Mandatory/Optional	Description
FILTER.GROUPID	Optional	`FILTER.GROUPID={*|[(]groupid[,groupid…][)]}`
kwd	Optional	The specified `kwd` lines are attributes which are retrieved and displayed. If no `kwd` is specified all group-related attributes are retrieved and displayed. `kwd`

ADDCONN – Add a User-Group Connection

Use the following request and parameter lines to add a user-group connection.

```
:ADDCONN=(userid, groupid)
```

Parameter	Mandatory/Optional	Description
CONN.AUTH	Optional	`CONN.AUTH=REG/ADMIN/AUDIT/ADMINAUDIT`

kwd	Optional	The `kwd` lines represent attributes which are specified in the SailPoint schema.
		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`

UPDCONN – Update a User-Group Connection

Use the following request and parameter lines to update a user-group connection.

```
:UPDCONN=(userid, groupid)
```

Parameter	Mandatory/Optional	Description
CONN.AUTH	Optional	`CONN.AUTH=REG/ADMIN/AUDIT/ADMINAUDIT`

kwd	Optional	The kwd lines represent attributes which are specified in the SailPoint schema.
		`kwd=value`
		or
		`kwd=(value1,value2,value3,value4...)`
		To nullify a keyword, the keyword must be specified with a null value.
		To delete one value from a multi-value keyword field, specify the keyword with all remaining values.

DELCONN – Delete a User-Group Connection

Use the following request and parameter lines to delete a user-group connection.

```
:DELCONN=(userid, groupid)
```

Parameter	Mandatory/Optional	Description
CONN.AUTH	Optional	`CONN.AUTH=REG/ADMIN/AUDIT/ADMINAUDIT`
OWNER	Optional	`OWNER=connection-owner`

LISTCONN – List User-Group Connection Information

Use the following request and parameter lines to list user-group connection information.

```
:LISTCONN
```

Default parameter setting:

```
FILTER.CONN=*
```

All the parameters listed below are mutually exclusive.

Parameter	Mandatory/Optional	Description
FILTER.CONN	Optional	All user-group connections or a specific user-group connection `FILTER.CONN={* | (userid,groupid)}`
FILTER.GROUP	Optional	All user connections with the specific group(s) `FILTER.GROUP=[(]groupid[,groupid…)]`
FILTER.USER	Optional	All group connections with the specific user(s) `FILTER.USER=[(](userid[,userid…)]`

LISTACL – List ACL (Permission) Information

Use the following request and parameter lines to list ACL (permission) information.

```
:LISTACL
```

Parameter	Mandatory/Optional	Description
FILTER.RESTYPE	Mandatory	`FILTER.RESTYPE=resource-type`
FILTER.RESNAME	Mandatory	`FILTER.RESNAME=resource-name`
kwd	Optional	The specified kwd lines are attributes which are retrieved and displayed. If no kwd is specified, all permission-related attributes are retrieved and displayed. `kwd=`

Invocation JCL

> **Note**
> Before running the utility, refer to the Security Requirements.

Use the CTSBATCH JCL member to invoke the batch utility. This job calls the batch utility from the CS Server procedure but uses a different program name (EXEC CTSACS,PROG=CTSCBAT). The input control statements for the utility are provided in DD statement SYSIN.

To use a different input file name, specify the TOKEN parameter and add the DD statement for the input file as follows:

```
//CALLCBAT EXEC CTSACS,PROG=CTSCBAT,TOKEN=(DDIN(BATINPUT))
```

```
//CTSACS.BATINPUT DD DSN=SPIIQ.V4000.INPUT(BATCHFAC),DISP=SHR
```

SYSIN File Example

The following is an example of a SYSIN file specifying several provisioning /list requests:

```
:ENV
RSS_TYPE=RACF
RSS_NAME=MVSRACF
ADMIN_UNAME=SECAL
*****************
:DELUSER=BLABLA1
MODE=DELETE
*****************
:ADDUSER=BLABLA01
USER.DFLTGRP=CC106410
USER.AUTH=ADMINAUDIT
OWNER=SECNY
NAME=SELIG
*****************
:UPDUSER=BLABLA01
WHEN.DAYS=SUN,MON
USER.AUTH=ADMIN
*****************
:LISTUSER
FILTER.USERID=BLABLA01
OWNER
NAME
WHEN.DAYS
****************
:ADDGROUP=SELIG01
GROUP.PARENT=CC106410
OWNER=SECST
****************
:ADDGROUP=SELIG02
GROUP.PARENT=CC106410
OWNER=SECST
****************
```

```
:LISTGROUP
FILTER.GROUPID=(SELIG01,SELIG02)
****************
:ADDCONN = (BLABLA01, SELIG01)
CONN.AUTH=AUDIT
REVOKE_DATE=20171231
*****************
:LISTCONN
FILTER.CONN=(BLABLA01,SELIG01)
```

Sample Batch Job Output

When executing the batch utility with the provisioning/ list requests as displayed in the SYSIN input file above, the following output is printed in the SYSPRINT file:

```
================================
*** Control-SA Batch Program ***
Control-SA 4.0.00 - BASE
================================
:DELUSER:
    User : BLABLA1
    ADDINFO: TYPE KEYWORD/VALUE
             1A: MODE = DELETE
API call: DELUSER OK
API msg(s):
2017/04/02 2:52:38 CTS1380I R SA-Agent Batch Utility version 4.0.00 ID
SECSTBAT/JOB00931 started
2017/04/02 2:52:38 CTS3084I R Feature LOCKED ACCOUNT (USER.CSDATA.CTSLKACT) is ini-
tialized successfully.
2017/04/02 2:52:38 CTS3081I R Custom Fields loaded successfully.
2017/04/02 2:52:40 CTS3016I R >>> "CTSAEXC DELUSER BLABLA1"
:ADDUSER:
    User: BLABLA01
    Group: CC106410
    password:
    Status: Ignored
    Authority: Administrator & Auditor
    password
        life   : Ignored
        ADDINFO: TYPE KEYWORD/VALUE
                  1A: OWNER = SECNY
                  1A: NAME = SELIG
API call: ADDUSER OK
API msg(s):
2017/04/02 2:52:40 CTS3016I R >>> "CTSAEXC ADDUSER BLABLA01 DFLTGRP(CC106410)
SPECIAL AUDITOR OWNER…
2017/04/02 2:52:40 CTS3016I R ICH01024I User BLABLA01 is defined as PROTECTED.
:UPDUSER:
    User : BLABLA01
    Group : \
    password : \
    Status : Ignored
    Authority: Administrator
    password
        life : Ignored
    Old Def UG Action: Ignored
        ADDINFO: TYPE KEYWORD/VALUE
                   1B: WHEN.DAYS = SUN,MON
API call :UPDUSER OK
API msg(s):
2017/04/02 2:52:41 CTS3016I R >>> "CTSAEXC ALTUSER BLABLA01 SPECIAL NOAUDITOR
WHEN(DAYS(SUN,MON))"
:LISTUSER:
User List
=========
User: BLABLA01
    Group : CC106410
    password :
    Status : Normal
    Authority: Administrator
    password
        status: Ignored
    ADDINFO: TYPE KEYWORD/VALUE
             1A: OWNER = SECNY
             1A: NAME = SELIG
             1B: WHEN.DAYS = SUNDAY,MONDAY
*** Total number of users found: 1 ***
:ADDGROUP:
    Group : SELIG01
    Parent Group: CC106410
    ADDINFO: TYPE KEYWORD/VALUE
             1A: OWNER = SECST
API call :ADDGROUP OK
API msg(s):
2017/04/02 2:52:41 CTS3016I R    >>> "CTSAEXC ADDGROUP SELIG01 SUPGROUP(CC106410)
OWNER(SECST)"
:ADDGROUP:
    Group : SELIG02
    Parent Group: CC106410
    ADDINFO: TYPE KEYWORD/VALUE
             1A: OWNER = SECST
API call :ADDGROUP OK
API msg(s):
2017/04/02 2:52:41 CTS3016I R    >>> "CTSAEXC ADDGROUP SELIG02 SUPGROUP(CC106410)
OWNER(SECST)"
:LISTGROUP:
Group List
==========
Group: SELIG01
Parent group: CC106410
    ADDINFO: TYPE KEYWORD/VALUE
             1A: DFP.DATACLAS =
             1A: UNIVERSAL = N
             1A: INFO.CREATE_DATE = 20170402
             1A: DFP.STORCLAS =
             1A: OMVS.GID =
             1A: MODEL =
             1A: DFP.MGMTCLAS =
             1A: OWNER = SECST
             1A: DFP.DATAAPPL =
             1A: DATA =
             1A: TERMUACC = Y
             1A: OVM.GID =
             1B: SUBGROUP =
             1B: TME.ROLES =
Group: SELIG02
    Parent group: CC106410
    ADDINFO: TYPE KEYWORD/VALUE
             1A: DFP.DATACLAS =
             1A: UNIVERSAL = N
             1A: INFO.CREATE_DATE = 20170402
             1A: DFP.STORCLAS =
             1A: OMVS.GID =
             1A: MODEL =
             1A: DFP.MGMTCLAS =
             1A: OWNER = SECST
             1A: DFP.DATAAPPL =
             1A: DATA =
             1A: TERMUACC = Y
             1A: OVM.GID =
             1B: SUBGROUP =
             1B: TME.ROLES =
:ADDCONN:
User : BLABLA01
Group : SELIG01
    Connection details : Auditor
    Connection attributes: None
    Default group :
        ADDINFO: TYPE KEYWORD/VALUE
                 1A: REVOKE_DATE = 20171231
API call :ADDCONN OK
API msg(s):
2017/04/02 2:52:42 CTS3016I R >>> "CTSAEXC CONNECT BLABLA01 GROUP(SELIG01)
AUDITOR REVOKE(12/31/17)"
:LISTCONN:
User to User-Group List
=======================
Group: SELIG01 -- User: BLABLA01
    Connection details : Auditor
    Connection attributes: None
    Default group : CC106410
*** Total number of connections found: 1 ***
```