Configuration Guide for SAS® 9.4 Foundation for z/OS

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2025. *Configuration Guide for SAS® 9.4 Foundation for z/OS*. Cary, NC: SAS Institute Inc.

Configuration Guide for SAS® 9.4 Foundation for z/OS

Copyright © 2013 - 2025, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

For a hard-copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a Web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a) and DFAR 227.7202-4 and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

June 2026

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

Table of Contents

Chapter 1 – Setting Up SAS 9.4 Foundation 1

Implementing SAS TSO Support 2

Architectural Changes Overview 4

Remote Browsing Replaces Remote Help 4
Threaded Kernel 5
TK Options 6
TKMVSJNL DDNAME 8
Support for Environment Variables on z/OS 8

Customizing Default Options and System Configuration Files 9

Selecting a Bundled Configuration 15

z/OS non-LPA (ENTRY=SASB) 16
z/OS LPA (ENTRY=SASLPA) 16

Installing SAS 9.4 Foundation into the LPA/ELPA 16

System Configuration for Using SAS with TCP/IP 18

Planning for TCP/IP 18
Overview and Software Requirements 18
Configuring SAS to Communicate with TCP/IP 19
References 29

Customizing Your SAS CLIST, REXX Exec, and Cataloged Procedure 30

Placing SAS Modules into LINKLIST or LPA 31

Customizing Your NEWS File 34

Installing the SAS 9.4 SVC Routine 34

(Optional) Installing the SAS SMF Exit 38

Required UNIX File System Components 40

Configuring SAS Software for Use with the Java Platform 40

Locating Hot Fixes 41

Chapter 2 – Post-Installation Configuration for National Language Support (NLS) 42

Introduction 42

Languages and Installation Codes 43

Selecting LOCALE During SAS Foundation Deployment 44

Verifying the Settings for the NLS Options 45
Running SAS in a Different Encoding 45
Running SAS in a Different Locale 46
Running SAS with Localizations 46
Using NONLSCOMPATMODE 47

DBCS Support 49

Asian Font Catalogs 49
The UPPERCASE (WU) image Installation 49

Chapter 3 – Post-Installation Configuration for SAS/ACCESS 50

SAS/ACCESS Interface to ADABAS 50

Customizing the SAS CLIST, REXX exec, and Cataloged Procedure 50
(Optional) Using the NATURAL Security Interface 51
(Optional) Using the Samples 51
(Optional) SAS Options for this Interface 51
NATURAL Date and Time Support 52
Reentrancy 52

SAS/ACCESS Interface to CA-DATACOM/DB 52
Customizing the SAS CLIST, REXX exec, and Cataloged Procedure 52
(Optional) Using the Samples 53
(Optional) SAS Options for this Interface 53
SAS/ACCESS Interface to CA-IDMS 54
Customizing the SAS CLIST, REXX exec, and Cataloged Procedure 54
SAS/ACCESS DATA Step Interface Enhancement 55
SAS/ACCESS Interface to DB2 56
Customizing the SAS CLIST, REXX exec, and Cataloged Procedure 58
Creating and Loading the Sample Tables (optional) 59
(Optional) SAS Options for this Interface 60
Special Consideration for Using the RRS Attachment Facility. 60
SAS/ACCESS Interface to IMS-DL/I 61
Customizing the SAS CLIST and Cataloged Procedure 61
Verifying Installation of the SAS/ACCESS Interface to IMS 62
SAS Options for this Interface 63
SAS/ACCESS Interface to Oracle 64
Oracle Multithreading Support 64
Customizing the SAS CLIST, REXX Exec, and Cataloged Procedure (Required) 65
Creating and Loading the Sample Tables (optional) 67
SAS/ACCESS Interface to R/3 67
SAS/ACCESS Interface to SYSTEM 2000 67
Customizing the SAS CLIST, REXX Exec, and Cataloged Procedure 67
SAS/ACCESS Interface to Teradata 69
(For SAS 9.4M9 and Later) Configuring SAS/ACCESS Interface to Teradata 69
(For SAS 9.4M7 and Earlier) Defining the Interface to Teradata 71
(For SAS 9.4M7 and Earlier) Customizing the SAS CLIST, REXX exec, and Cataloged Procedure 72
Creating and Loading the Sample Tables (optional) 73
TPT or Legacy Teradata Utility (FastExport, FastLoad, and Multiload) 73
Legacy Teradata Utility Configuration 73
FastExport (optional) 73
MultiLoad (optional) 74
Chapter 4 – Post-Installation Configuration for SAS/ASSIST 75
Adding a Master Profile 75
Installing Sample DB2 Tables and a Sample Query Manager 76
Chapter 5 – Post-Installation Configuration for SAS/CONNECT 79
Configuring SAS/CONNECT 79
Communication Access Methods for SAS/CONNECT on z/OS 79
System Configuration for the TCP Access Method 79
System Configuration XMS (Cross Memory Services) Access Method 79
Setting Up TLS for SAS/CONNECT 80
Storing and Locating SAS/CONNECT Script Files 80
Types of Connections Available with SAS/CONNECT on z/OS 80
SAS/CONNECT Basic Telnet Session 80
SAS/CONNECT Spawner for z/OS 81
SAS/CONNECT to the Same Multi-Process Machine on z/OS 86
Chapter 6 – Post-Installation Configuration for SAS/IntrNet 89
Overview 89
Installing and Configuring SAS/IntrNet 90

Install Your Web Server Software 90
Install Your SAS Software 90
Test the Web Server 92
Test the Application Broker 92
Configure a Socket Service 93
Starting the Socket Service 95
Testing the Socket Service 95
Configure Additional Services 96

Chapter 7 – Post-Installation Configuration for Encryption 97

Changes to System Options 97

SAS/SECURE Client for Windows 97

SAS/SECURE Client for Java 97

FIPS Support 98

Enabling TLS 98

Pervasive Encryption 98
Configuring Pervasive Encryption 99
Troubleshooting Pervasive Encryption Configuration 99

System SSL Encryption 100
Configuring the SAS Intelligence Platform Server Tier to Use System SSL 100
Configure SAS Management Console 101
Configuration Examples for Accessing System SSL Certificates 101

Chapter 8 – Post-Installation Configuration for Workspace Server for Jobs with Large Utility File Requirements 102

Chapter 9 – Post-Installation Configuration for Security 103

SAS Security in a z/OS Environment 103
Deprecated Software 103

Multifactor Authentication 104

Chapter 10 – Post-Installation Configuration for SAS/SHARE 105

Special Files for Use with SAS/SHARE Software 105
Customizing the Started Task JCL Procedure for a Server 105
Configuration File for a Server 105

Customizing the SAS/SHARE Autocall Macros 105

Selecting Communications Access Methods to Use 106

System Configuration for the Cross-Memory Access Method 107
Installing the SASVXMS Load Module 107
Defining an Anchor Point 108

System Configuration for TCP/IP 109
Specify SAS 9.4 Foundation Option TCPSEC=_SECURE_ for the Server Execution 109
Testing the SAS/SHARE Server Startup 109

Client-Side Components 109
SAS/SHARE Data Provider 109
SAS ODBC Driver 109
SAS/SHARE Driver for JDBC 109
SAS/SHARE SQL Library for C 110

Special Consideration for the SECPROFILE System Option 110

Chapter 11 – Post-Installation Configuration for SAS/TOOLKIT 111

Chapter 12 – Post-Installation Configuration for SAS System Forms 113

(Optional) Customizing the Printer Selection List.....113

Chapter 13 – Post-Installation Configuration for System Services Scheduler.114

Chapter 14 – Licensing SAS.....115

Chapter 15 – Logging Directly on to the SAS System 116

Installing the Direct Logon Procedure 116

Example Logon Procedure 116

Using Direct Logon 116

Logging onto the SAS Display Manager System 117

Logging on to a Windowing Application..... 117

Restrictions 117

Accounting Considerations 117

Chapter 1 – Setting Up SAS 9.4 Foundation

This document describes the configuration instructions for SAS 9.4 Foundation, which is made up of server-side Base SAS and a variety of server-side SAS products (which vary by customer). It has been updated for SAS 9.4M9 (TS1M9).

***Note**: SAS 9.4M0 through 9.4M6 entered Limited Support from SAS in February 2025. SAS 9.4M7 entered Limited Support from SAS in September 2025.*

Verifying that your site meets the system requirements before you deploy SAS 9.4 Foundation is critically important. For more information, read the SAS 9.4 Foundation for z/OS System Requirements document.

Information about the configuration of mid-tier and client-side products is available from SAS Help Center.

The server-side configuration instructions contained in this document are for the configuration of a generic SAS server. If you want to configure your server for more specific functions, such as a Workspace Server or Stored Process Server, refer to the *SAS 9.4 Intelligence Platform: Installation and Configuration Guide*.

Part 1 describes how to tailor your SAS installation to suit your particular site configuration. The following topics are discussed in this section:

- ❑ Implementing SAS TSO Support (recommended)
- ❑ Architectural Changes Overview (recommended)
- ❑ Customizing Default Options and System Configuration Files (recommended)
- ❑ Selecting a Bundled Configuration (recommended)
- ❑ Installing SAS 9.4 Foundation into the LPA/ELPA (recommended)
- ❑ System Configuration for Using SAS with TCP/IP (recommended)
- ❑ Customizing Your SAS CLIST, REXX Exec, and Cataloged Procedure (recommended)
- ❑ Customizing Your NEWS File (recommended)
- ❑ Installing the SAS 9.4 SVC Routine (recommended)
- ❑ Installing the SAS SMF Exit (optional)
- ❑ Required UNIX File System Components (recommended)
- ❑ Configuring SAS Software for Use with the Java Platform (recommended)
- ❑ Locating Hot Fixes (recommended)

***Important:** Each of the "recommended" tasks should be reviewed by someone at your site who can determine whether the suggested steps are appropriate and would be consistent with your site's operating procedures, options, and parameters. Recommended tasks in this guide are designated with a note. Some of the tasks and actions might require knowledge of z/OS operating system principles.*

Implementing SAS TSO Support

This task is required if you are going to run SAS interactively under TSO.

Notes:

- If you want to run SAS interactively under TSO, you must copy the SASCP module to a TSO command or link list library.
- If you are configuring servers such as a workspace server or stored process server on z/OS, you must copy SASCP to a TSO command or link list library before invoking SAS Deployment Wizard to perform the configuration.
- If you are configuring a SAS metadata server to run on z/OS, you must copy SASCPZ64 and SASCPZAE to a link list library before invoking SAS Deployment Wizard to perform the configuration.

Note that SAS version 9.3 required you to add module SASCP64 to a link list library. If you intend to run versions 9.3 and 9.4 simultaneously, you will need all 3 modules—SASCP64, SASCPZ64, and SASCPZAE—in your link list library.

For details, see SAS Note 43697.

If you plan to run SAS under TSO, you must install the SASCP TSO command processor as outlined below, even if you previously installed it in an earlier release. If you do not install the SASCP TSO command processor, you can run SAS in batch mode only. You must also use either the REXX exec or the CLIST supplied as part of this installation to run SAS 9.4 Foundation under TSO. **Do not try to use a SAS CLIST from a previous release due to new file allocations and other changes.** The REXX exec is an alternative to the SAS CLIST for TSO users, and is the default SAS startup command for the SAS/CONNECT Spawner and the SAS/Integration Technologies Object Spawner.

SAS TSO support includes two different facilities. The SAS TSO command processor allows you to invoke SAS from a TSO session. The SAS TSO command support feature provides a SAS statement for executing TSO commands from a SAS session and allows SAS DATA step programs to execute TSO commands conditionally.

SAS 9.4 Foundation supports these features through installation-modifiable modules. These modules contain all of the TSO service routine dependent functions and make no reference to SAS service routines or data areas. These modules are:

- **SASCP**: the TSO command processor for invoking 31-bit SAS 9.4 Foundation.

  Note: *SASCP is backward-compatible and replaces the modules from previous releases. You can continue to execute previous releases of SAS with the Release 9.4 SASCP installed on your system.*

- **SASTSO**: the TSO command executor for executing TSO commands.
- **SASCALL**: the TSO `CALL` command processor used by `SASTSO`.
- **SASTSMAC**: the macros necessary to assemble these modules.

These source modules are created during installation in the `&prefix.BAMISC` library. The default load modules `SASCP`, `SASTSO`, and `SASCALL` are unloaded into your `&prefix.LIBRARY`. These modules work correctly under all levels of TSO.

Most sites will not need to modify the modules. If you do not need to customize the modules, proceed to STEP 3. However, should your site have special requirements, STEPs 1 and 2 describe how to modify and reassemble these modules.

STEP 1: Determine whether the distributed TSO support modules require installation customization for your site. Perform customization if needed. (Optional)

The Assembler source code for the SASCP, SASTSO, and SASCALL modules is available in your &prefix.BAMISC library for modification. Assembly requires that the SYSLIB concatenation contain the general use system macro library SYS1.MACLIB, the product-sensitive system macro library SYS1.AMODGEN or SYS1.MODGEN, and the BAMISC library. Examine the source code for the TSO support modules for further details. Sample JCL for assembling and linking these modules is included with the comments of the modules.

If your site has modified the source code and re-linked the SASCP module, you can use the TSO TEST command to test SASCP directly from the SAS library by specifying the CP option of TEST. Create a test version of the SAS9 CLIST and insert the TEST command with the CP option immediately before the SASCP invocation at the end of the CLIST.

STEP 2: Modify system tables as needed. (Optional)

This installation might also require that you modify certain system tables, such as the PCF and/or ISPF Command Authorization tables. These modifications might require the assistance of local systems or technical support personnel.

The TSO command executor, SASTSO, contains support for TSO command validation by both PCF and ACF2. The use of ACF2 command validation must be explicitly enabled either by modifying and reassembling SASTSO (following the instructions given in the program header), or by using the z/OS service aid, AMASPZAP. Contact the Technical Support division at SAS if you have any questions concerning these procedures.

The functionality that will allow you to interface SAS with IBM's ISPF is included in this release. For this reason, you should examine the ISPF Command Table, ISPTCM. For more information about this functionality, refer to "SAS Interfaces to ISPF and REXX" in the *SAS 9.4 Companion for z/OS*.

If the SAS command processor, SASCP, is defined within your installation's ISPTCM, the flag bit that allows a function pool to be created for the command must be on x'40'. If SASCP is not in your ISPTCM, either the default flag value must include the preceding bit, or you should do one of the following:

- change the default flag value
- add SASCP to the table with the flag on

In the sample ISPTCM that is distributed by IBM, the bit is ON in the default flag value.

STEP 3: Copy the SASCP module to your TSO command load library.

Note: *This step is required if you are running under TSO.*

Installation of SAS TSO support requires copying the module SASCP from the &prefix.LIBRARY to a load library that contains TSO commands. This can be a STEPLIB library defined in a LOGON procedure, a system link list library, or a link pack area library. (The SASTSO and SASCALL modules should remain in &prefix.LIBRARY.)

Use job BASASCP in the CNTL data set to copy SASCP to your TSO command load library. The CMDDSN, CMDUNIT, and CMDVOL JCL procedure parameters specify the user command load library into which you copy the SASCP module. Modify these values to specify your TSO command library. If

you have customized the `SASCP` module and are storing it in a library other than `&prefix.LIBRARY`, you also need to specify its new location in the `//SASLIB DD` statement.

Check the JCL and run the `BASASCP` job.

Note: *If you are running SAS Metadata Server on z/OS, the Z64 CNTL data set also has a job named BASASCP. You must run both BASASCP batch jobs. The Z64 version will put SASCPZ64 and SASCPZAE (as well as its alias SASCPZAM) into your TSO command load library. For configuration, this must be in a link list library rather than a TSO command load library.*

You can follow the instructions in SAS Note 43697 before installing SAS 9.4, or you can use SAS Deployment Wizard to install your software without configuring your servers; use BASASCP jobs to move these modules to a link list library; and then use SAS Deployment Wizard to configure your servers.

Architectural Changes Overview

For more information about changes to SAS 9.4 Foundation on z/OS, see the "What's New" section of *SAS 9.4 Companion for z/OS*.

Remote Browsing Replaces Remote Help

A documentation server is integrated into SAS 9.4 Foundation. The documentation files are contained in the UNIX File System Components. The Remote Browser Server component is still used to display HTML content on a web browser.

Remote browsing also enables users to view ODS HTML output on their desktop PC. With the UNIX File System Components installed and the `HELPLOC` system option properly set, ODS HTML files will display in a browser either by clicking on the files in the **Results** window or as they are generated by ODS.

SAS 9.4 Foundation also includes an updated Remote Browser Server for Windows, with an enhanced user interface. SAS highly recommends that users upgrade to the new version, which can be downloaded directly from their SAS session.

Exit the RBS running on your system and run SAS, and then select **SAS Help and Documentation** from the Help menu. The installation URL will be printed to the SAS log.

Additional configuration of your system might be required in order to use remote browsing. If your TCP/IP networking is configured to use a hosts table instead of DNS (see "Using a Host Table" for details), and one or more SAS desktop clients resides outside the DNS domain of the SAS host, change the hosts table to specify the system's fully qualified domain name (FQDN) as the primary host name for local network interfaces.

For example, SAS is installed on the host `sashost.<company_name>.com`, two client computers exist in the same domain (`client1.<company_name>.com` and `client2.<company_name>.com`), and two other clients exist in another domain (`client3.domain2.<company_name>.com` and `client4.domain2.<company_name>.com`). This relationship is illustrated in Figure 1.

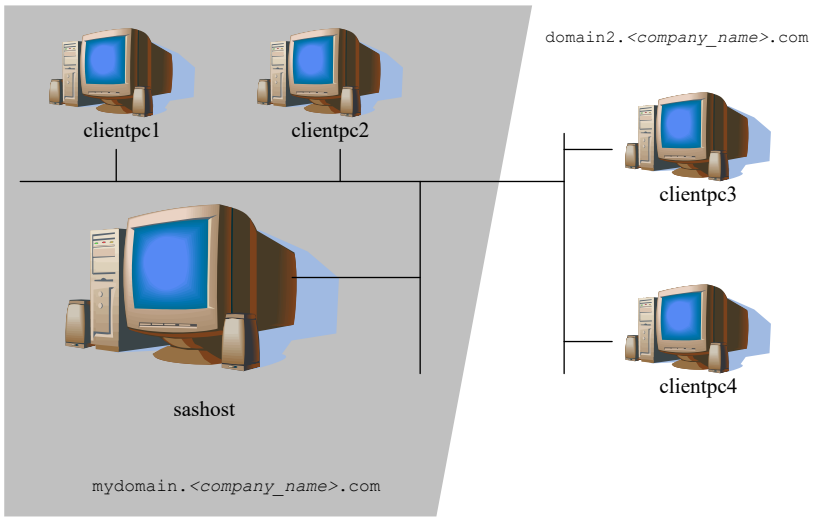


Figure 1. Desktop Clients Inside and Outside the Host's DNS Domain.

If the system `sas.company_name.com` is not configured with a host name that is the FQDN for the system, client1 and client2 will be able to view HTML content from SAS, but client3 and client4 will not. URLs generated for the SAS host will not include the domain, as in the following example:

```
http://sashost:12345/output.html
```

Because client1 and client2 are in the same domain as the Base SAS host, their browser will build valid host names from their domain, `sashost.company_name.com`.

But client3 and client4, which are outside the domain of the SAS host, will use their domain names to construct a complete hostname, which results in the invalid name `sashost.domain2.company_name.com`.

When you configure the SAS host with the system's FQDN, URLs for HTML display are valid from all of clients. From the example, the valid URL for all clients resembles the following example:

```
http://sashost.<company_name>.com:12345/output.html
```

Instructions for changing the host table are described in "Using an FQDN with a Host Table" on page 2626.

Threaded Kernel

The SAS threaded kernel (TK) is an independent internal interface to low-level operating-system interfaces such as memory, events, task creation, etc. The TK interface is booted one time in an address space and its services are available to any task in that address space. SAS makes use of TK services. Moreover, SAS is itself initiated as a TK-created task rather than being entered directly as the job step task or via `ATTACH`. SAS runs, in effect, as a TK application. The introduction of TK into the SAS environment enables SAS to invoke operating-system services in a portable manner on multiple platforms, thereby enabling concurrent operations on multiple processors from multiple operating-system tasks.

Over time more SAS procedures will exploit the TK interface. As a reflection of these architectural changes, there are some changes in the SAS initialization and termination details in the following areas:

- new entry points
- return codes
- TK options

- TK use of HFS
- TKMVSJNL DDNAME

TK Options

A mechanism in SAS 9.4 to supply MVS platform-specific options to the TK interface, TK options are independent and separate from SAS application options. They are not specified with the SAS application options on the JCL EXEC card or the SAS CLIST or REXX exec. TK options are instead provided by the standalone program TKMVSENV (see "Support for Environment Variables on z/OS" on page 88).

The TKMVSENV program maintains name/value pairs, similar to UNIX environment variables. These name/value pairs have a lifetime of the job step task. Thus, names defined in a batch job step will be undefined when the batch job step ends. Names defined during a TSO session, on the other hand, will be defined for the life of the TSO session (or until redefined or cleared by another invocation of the TKMVSENV program). The TK boot process makes TKMVSENV name/value pairs available to any application using TK services, including SAS.

The TKMVSENV program will obtain values from one of two sources. If the TKMVSENV program receives standard parms from the `JCL EXEC` statement or the `TSO CALL` command line, the program processes the command found in that parms image. If, however, the standard parms length is zero, the program will look for a data set allocated with the DDNAME of `TKMVSENV`. If the data set is found, the program processes each record in the data set as a command. If the TK boot process finds a TKMVSENV DDNAME allocated, TK boot invokes the TKMVSENV program with a zero length parms specification. Because a TKMVSENV data set is included in the SAS proc, CLIST, and REXX exec it is, in effect, a TK options file.

The TKMVSENV file created by the installation process has valid settings for the options necessary for most SAS processing. Many of the remaining supported options are required only in special debug circumstances, primarily when SAS technical support needs information for problem determination purposes. The data set is included in the SAS PROC, CLIST and REXX exec so that it will be available when needed. In addition, certain SAS applications (such as the JAVA interface) might make use of the TKMVSENV data set to provide options specific to the JAVA interface. The following is a list of the TKMVSENV name/value pairs supported by the TK interface. Note that some of these are used as Boolean-style options and do not require a value after the = sign on the `set` command, although a value can be provided if desired.

TKMVSENV Command	Meaning
`set SASROOT=`	Set to the UNIX file system location where additional SAS Foundation content is stored.

`set TKJNI_OPT_LIBPATH=`	Specifies the paths of Java shared libraries that TKJNI uses on MVS. The most important of these libraries is libjvm.so. That library uses other libraries, such as libjitc.so. The paths to these libraries are specified using colons as delimiters, as they would be in UNIX System Services (USS).
	
	TKJNI will not be able to start the Java Virtual Machine if `TKJNI_OPT_LIBPATH` is not provided or is incorrectly specified in the TKMVSENV dataset. Here is an example:
	
	`set TKJNI_OPT_LIBPATH=/usr/lpp/java/IBM/J8.0/bin/j9vm:/usr/lpp/java/IBM/J8.0/bin`
	
	For SAS 9.4M9, the command might resemble the following example:
	
	`set TKJNI_OPT_LIBPATH=/usr/lpp/java/IBM/J21.0/lib/j9vm:/usr/lpp/java/IBM/J21.0/lib`

TKMVSENV Command	Meaning
`set TKJNI_OPT_PROXYPATH=`	Instructs SAS where the JPROXY module can be found. JPROXY runs in USS and is needed when Java applications are started from within SAS.
`set TKOPT_ENV_LOCALE=`	Specifies a locale that is used by the TK subsystems. This locale is mainly used when an MVA locale is not available. It is used for TK journal files and other TK applications. When SAS is installed, the value of `TKOPT_ENV_LOCALE` matches the installed value of the `MVA LOCALE` option setting. Valid values are the 5-byte locale names documented in the NLS Reference Guide for the `LOCALE` option.
`set TKOPT_ENV_ENCODING_PATH=`	Specifies the encoding used for USS pathnames. The pathname encoding defaults to open edition 1047. If you want to use National Characters in pathnames, you must set the `TKOPT_ENV_ENCODING_PATH` in TKMVSENV. If you set this variable and MVA is running, the encoding must be set to the MVA session encoding. This variable is not set in the TKMVSENV when SAS is installed.
`set TKOPT_ENV_ENCODING=`	Sets the encoding that TK uses for data in a file. This encoding is used for TK logging and other purposes. When SAS is installed, the value of `TKOPT_ENV_ENCODING` matches the MVA session encoding that is set during MVA initialization. Valid encoding names are documented in the NLS Reference Guide.
`set TKOPT_NOSTAE=`	Boolean option. Instructs the TK interface NOT to provide an MVS ESTAE around TK created tasks. The MVS ESTAE that is normally provided to protect these tasks enables TK applications to potentially recover from abends. This option is analogous to the SAS NOSTAE option.
`set TKOPT_NOSTAEX=`	Requests that when a TK created task ABENDs, recovery routines are not to be run before percolating the ABEND to SAS as a whole. (See U996 TK abend description above.) This option should not generally be used except at the direction of SAS technical support for diagnosis of a problem.
`set TKOPT_DUMPPROL=`	Boolean option that provides tasks created by TK with additional debug information in the save areas provided by the function prolog and epilog code, easing the dump reading process. Analogous to the `SAS DUMPPROL` option.
`set TKOPT_TKABORT=`	Set to `TRUE` or `FALSE` (which may be abbreviated to a single character). The default is `FALSE`. If set to `TRUE`, SAS attempts to continue execution after abend of TK-created tasks. Depending on the abend, this may cause instability or further abends. `TKOPT_TKABORT` is ignored if `TKOPT_NOSTAE` or `TKOPT_NOSTAEX` is specified. This option is not recommended unless requested by SAS Technical Support to resolve or diagnose a problem.
`set TKOPT_MEMFILL=`	Boolean option that fills memory provided by the TK memory management interface with special characters at memory get and memory free time. This option is intended to enhance the debug process and is analogous to the SAS `$VMMFILL` option.
`set TKOPT_SVCNO=nnn` `set TKOPT_SVCR15=nn`	Instruct the TK interface how the SAS SVC is installed at the user site. Required because the TK interface might need to use some of the SVC services INDEPENDENT of the SAS application. These options should be specified the same as the SAS options of the same name.
`set TKOPT_LPANAME=xxxxxxxx`	Specifies the name of the SAS application entry point invoked by the SAS SASLPA main entry point. If the installation placed the LPA resident module in the LPA with a name other than `SASXAL`, you must specify the same name for the `TKOPT_LPANAME` option value.

TKMVSENV Command	Meaning
`set TKOPT_MEMLEAVE=nnnnnnn`	Specifies the amount of memory in bytes that the TK memory subsystem will attempt to leave free in the address space for use by the system and other users.

TKMVSJNL DDNAME

TK includes the concepts of a global journal and a global IO. Sometimes, error information is placed in the buffer of the global journal and written to the global IO. The default IO is opened first to the SASCLOG DDNAME, if it is present, and then to the TKMVSJNL DDNAME, if it is present. Otherwise, any information normally directed to the global IO will be lost. The SASCLOG DDNAME is part of the production SAS proc, CLIST and REXX exec, so you should look there to find any messages written to the global journal. If you have trouble initializing some non-SAS programs like the V9 object spawner, you might receive error information by allocating a TKMVSJNL DDNAME. However, if the spawner has trouble initializing a spawned SAS process, you should also look to the SASCLOG of the spawned process for initialization error messages. This SASCLOG is most likely a JES SYSOUT file whose owner is the user ID of the spawned process. You can find these files using a JES Spool browser such as SDSF, IOF APPC, or EJES.

Support for Environment Variables on z/OS

TKMVSENV is a stand-alone program that maintains a set of name/value string pairs or environment variables. The name/value pairs only exist during the MVS jobstep task in which the program is invoked.

TKMVSENV takes a standard JCL EXEC style parm list. General register 1 is expected to point to a single parm, which is the address of a half word length followed by a parm string of that length. The parm string is composed of a single command. If the half word length field is zero, the program looks for an allocated DDNAME of TKMVSENV. If the DDNAME is allocated, the file is read and processed one record at a time. Each record is expected to contain a single command. The file must be a sequential data set or a PDS member with a record format of fixed blocked and a logical record length up to 32760. The record length of the sample TKMVSENV file provided with the install is 560, which is the recommended minimum record length.

There are five valid commands. String values are case-sensitive. If you invoke TKMVSENV from a TSO CALL command or from batch JCL and you need to ensure mixed-case values are passed, specify the ASIS parm on the TSO CALL command image or set CAPS OFF in an ISPF edit session before submitting the batch JCL. Valid commands must begin in column 1, followed by one or more blanks. No blanks are allowed on either side of the '=' sign.

Valid commands are as follows:

Command	Function
`set name=value`	Set a name/value pair
`get name`	Get a name value
`clear name`	Clear a name from the list
`reset`	Reset the list to zero members
`disp`	Display all name/values in the list (TSO only)

For the `get` command, R15 returns the address of a half word length of the value, followed by the value string. If the requested name was not found, R15 will contain 0 (zero).

There are two interfaces to the TKMVSENV utility from MVS TK:

1. TK NAMED/REPOSITORY calls with the special prefix of "OSENV." will be mapped to TKMVSENV calls as described in the TK documentation.
2. TKMVSENV will be invoked by MVS TK initialization processing with a zero-length parm. Thus you can allocate a TKMVSENV DDNAME to your TK invocation environment and define name/value pairs which can be queried by the TK application via the OSENV NAMED REPOSITORY interface.

Customizing Default Options and System Configuration Files

In the process of installing SAS 9.4 Foundation under z/OS, you establish default SAS system options for all SAS invocations at your site. You can set these global defaults in any of the following places:

- Default Options Table
- System configuration file
- Restricted Options Table

Sample system configuration files are shipped with SAS 9.4 Foundation in the `&prefix.CONFIG` data set. The CLIST, REXX exec, and cataloged batch procedure each concatenate three members of this PDS as the SAS `CONFIG` file. The first member is `COMMON`. The second member is named `llxx`, where "`ll`" is the language code and "`xx`" is the media and data set code. For example, it is `ENW0` for English and `FRW0` for French. The third member is named `SITE`. These files are discussed in more detail later in this section.

Users can also specify options in any of the following locations:

- in a user configuration file, which is specified using the `CONFIG` operand in the CLIST or REXX exec and the `CONFIG` parameter in the cataloged procedure
- on the command line when invoking SAS under TSO
- with the `OPTIONS` parameter in the cataloged procedure, if invoking it under batch
- from within a SAS session, either on an `OPTIONS` statement or from the `OPTIONS` window in the windowing environment

The following indicates the order in which SAS 9.4 Foundation processes options from the various sources:

1. Options in the Default Options Table (if assembled)
2. Options in a system configuration file (if one is used)
3. Options in a user configuration file (if one is specified)

4. Options supplied on the command line in the invocation of the CLIST or REXX exec (TSO) or using the `OPTIONS` parameter in the cataloged procedure (batch)
5. Options in the Restricted Options Table (if assembled)
6. Options specified on an `OPTIONS` statement or in the `OPTIONS` window

Later specification of an option overrides an earlier specification. For example, options set in a user configuration file override options set in the system configuration file. Depending upon where you set them, you can streamline system startup or restrict use of options by users.

To streamline system startup for users under TSO, for example, you can add all option defaults needed for TSO at your site to the Default Options Table. Then, since users need not allocate a system configuration file at startup time, you can remove its allocation from the CLIST. In this case, you still need a system configuration file for batch jobs to override the values tailored for TSO in the Default Options Table.

To restrict use of options by users, you can set them in the Restricted Options Table. Because this table is processed last after the Default Options Table, configuration files, and command line options, values set here override all earlier specifications.

STEP 1: Determine default values for SAS system options at your site.

SAS system options that can be used in any operating system environment in which SAS 9.4 Foundation is deployed are described as *portable.* For more information, see *SAS 9.4 System Options: Reference.* Any z/OS-specific system options are discussed in *SAS 9.4 Companion for z/OS*.

Note: *z/OS-specific options that warrant special attention at installation time are discussed below. These options include the SMF- and SVC-related options and the superblocking options.*

SMF and SVC Options

The following options should be set in the Restricted Options Table with values that you select at installation time. The settings of these options are unlikely to require change. Because of the measurements that these options govern, SAS recommends removing these option settings from general user access.

- **`SMF|NOSMF`** causes an SMF record to be written for every PROC or DATA step containing resource usage statistics for CPU time, memory, and `EXCP` count. `NOSMF` disables writing of utilization statistics by SAS 9.4 Foundation. The default is `NOSMF`.

Note: *SAS 9.4 Foundation option `STIMER` must also be on for SMF records to be written.*

- `SMFEXIT=`***`name`*** identifies the user SMF recording exit load module. This load module is loaded and given control before SMF records are written. It allows the user to modify the contents of the record to be written or to disallow the writing of the record (if SMF option is in effect). There is no default.

  For information on installing the SMF exit, see "Installing the SAS SMF Exit" on page 3837.

- SMFTYPE=*recnum* identifies the SMF record type to record and specifies the default user type for SAS 9.4 Foundation to place in the SMF records it generates when the SMF option is on. The value must be greater than 127 for the SAS 9.4 Foundation SVC to write the SMF record. The default is 128.
- SVC0R15=*value* specifies the value to be placed in Register 15 before invoking the SAS SVC. Only used if SVC0SVC=109. The default is 4.

  For information about installing the SAS SVC, see "Installing the SAS 9.4 Foundation SVC Routine" on page 3434.
- SVC0SVC=*number* specifies the SVC number invoked for functions requiring the SAS SVC. The default is 109.

  For information about installing the SAS SVC, see "Installing the SAS 9.4 Foundation SVC Routine" on page 3434.

Superblocking Options

To decrease memory fragmentation, SAS 9.4 Foundation has the ability to obtain large blocks of memory from the operating system to satisfy multiple requests for smaller blocks of memory. This scheme, which is called *superblocking*, not only reduces fragmentation but also reduces the number of system GETMAIN calls that are issued. This facility is controlled by setting the superblocking options. When they are set to zero, no superblocking is performed.

The superblocking options warrant special attention at installation time because useful values can depend on the mode (batch or TSO) in which SAS 9.4 Foundation runs. The default values for these options are based on early and fairly limited experience in running SAS 9.4 Foundation. In most cases, you should not need to override the default values.

To see the current values of these options and other options related to memory, as well as where they were set, submit

```
proc options group=memory value; run;
```

SAS issues superblock overflow warning messages if secondary (OSA) superblock memory is needed. These messages can help you to tune these values for your site.

The following options can be used at SAS invocation or in a configuration file to specify the size of the superblocks. The values can be specified in bytes, kilobytes (K), or megabytes (M).

- **PSUPISA=*value*** – the size of the Initial Size Allocation (ISA) for the portable supervisor.
- PSUPOSA=*value* – the size of the Overflow Size Allocation (OSA) for the portable supervisor.
- **VMCTLISA=*value*** – the size of the ISA for SAS memory management control blocks.
- **VMNSISA=*value*** – the initial size allocation of NOSIG pools.
- **VMNSOSA=*value*** – the overflow size allocation of NOSIG pools.
- **VMPAISA=*value*** – the size of the ISA for permanent memory above the 16 Mb line. Permanent memory is memory that is used past procedure or task termination, typically by the host and core supervisor.
- **VMPAOSA=*value*** – the size of the OSA for permanent memory above the 16 Mb line.
- **VMPBISA=*value*** – the size of the ISA for permanent memory below the 16 Mb line.
- **VMPBOSA=*value*** – the size of the OSA for permanent memory below the 16 Mb line.
- **VMTAISA=*value*** specifies the size of the ISA for temporary memory above the 16Mb line. Temporary memory is that memory which needs to be resident only while the task or procedure

is active. This type of memory is highly transient, so it is always cleaned up at the end of the task or procedure. Almost all PROC step memory, DATA step memory, and I/O buffers come from this class of memory.

- **VMTAOSA=*value*** – the size of the OSA for temporary memory above the 16Mb line.
- **VMTBISA=*value*** – the size of the ISA for temporary memory below the 16Mb line.
- **VMTBOSA=*value*** – the size of the OSA for temporary memory below the 16Mb line.

WTO Options

Three options control the way system operator messages are issued if a SETINIT failure occurs. All these options are invocation-only so that you can restrict them at your site by including them in the Restricted Options Table. If you work with a systems programmer at your site, you can set values for these options that cause SETINIT error messages to be trapped by operating system automation software.

To see the values of these options, specify the following:

```
proc options group=install; run;
```

Each of the WTO options corresponds to one of the keywords that can be supplied on the WTO system macro.

- **WTOSYSTEMDESC=*n*** Use this option to specify the message descriptor code. The value of this option is passed to the `WTO` macro with the DESC keyword.
  See the IBM documentation for the meaning of the various values (0 to 16) that can be supplied.

- **WTOSYSTEMMCSF=*(list-of-keywords)*** With this option you supply keywords that control the display of the message. Multiple values are permitted. If you specify more than one value, you must enclose them in parentheses; if you specify only one value, the parentheses are optional.

Here are the keywords that you can supply:

`BRDCST`	Broadcast the message to all active consoles.
`HRDCPY`	Queue the message for hard copy only.
`NOTIME`	Do not append time to the message.
`BUSYEXIT`	Do not wait for WTO buffers.

This option corresponds to the MCSFLAG keyword on the WTO macro.

- **WTOSYSTEMROUT=n** Use this option to specify the message routing code. The value of this option is passed to the WTO macro with the ROUTCDE keyword.
  See the IBM documentation for the meaning of the values (0 to 16) that can be supplied.

The BNDLSUFX= Option

Specifies a character that is to be appended to every bundle load module name before it is searched for or loaded. The character is appended to the name of every bundle load module (these modules have a prefix of SAB and includes the module SASXAL). If the name of the bundle is eight characters long already, the suffix character replaces the last character. The value for the `BNDLSUFX=` option can be enclosed in quotation marks, but they are not required. See "Selecting a Bundled Configuration" on page 1515 for more information.

The BNDLSUFX= option is typically used only by system administrators, and not by the general user.

The SUBSYSID= Option

Tells the cross-memory services communication facility to use the z/OS subsystem ID that was chosen in the installation process to anchor its resource descriptors. The default value is SAS0. This option is used in conjunction with SAS/SHARE software.

The OPRESTRICTIONS= Option

Sets the name of the Restricted Options Table load module, which sets initial options and prevents the user from overriding them. The syntax is OPRESTRICTIONS=AAAAAAAA where AAAAAAAA is the name of a z/OS load module that must be in LPA or the linklist. See "Step 5: Create a Restricted Options Table" for more information.

The DLINITDEFER Option

Suppresses synchronization of VTOC entry at library creation time. If your site uses SMS management classes which specify partial release = yes immediate, or if you utilize a system exit to release space when data sets are closed, you might want to consider specifying DLINITDEFER as a default option for your site.

STEP 2: Determine where to set your option defaults.

Review the procedures for setting default option values in the next three steps. Decide which options should be set in the Default Options Table, the system configuration files, and the Restricted Options Table. Save these lists for use in later steps.

STEP 3: Customize the supplied DFLTOPTS table.

Customizing DFLTOPTS (Default Options Table) is optional. If you decide to customize it, edit the DFLTOPTS assembler source by adding the options to the table that you would like to include and removing those you do not want.

The source for the DFLTOPTS table resides in the &prefix.BAMISC library member DFLTOPTS. This source represents the DFLTOPTS table that is linked into the SAS load modules on your installation tape. Instructions for modifying the DFLTOPTS table are included in comments in the source code. The JCL to assemble and link it is in member BAOPTS1 in the CNTL data set. Run the job to assemble the DFLTOPTS CSECT and link it into SASHOST individually, and into the bundles of which SASHOST is a part. Be sure that if you run a bundled configuration, you re-link the bundles that you use.

Because the DFLTOPTS table is linked with SASHOST, it does not have to be loaded to be read. If you can put all the default options that you need in the DFLTOPTS table, you do not have to use a system configuration file.

The DFLTOPTS table can contain as many option length/value pairs as needed. An option length/value pair consists of a half word length field, followed by a character string of the form option, NOoption, or option=value. The OPT macro in the assembler source calculates the length fields given the character strings. The table must be terminated by a pair with a length field of 0.

STEP 4: Customize the system configuration files.

Set up system configuration files to establish installation-wide default values for commonly-used options. The default SAS CLIST, REXX exec, and cataloged procedure installed from the tape always allocate a system configuration file and allow for specification of a user configuration file using the CONFIG

operand and parameter. They use sample system configuration files that are created in the `prefix.CONFIG` data set; these samples are described below.

All SAS 9.4 Foundation installations, whether domestic or foreign, are now encoded images. Encoded images allow SAS 9.4 Foundation to be invoked using different character sets. For example, W0 is English and W3 is German, etc. Other examples of the encoding abbreviations can be found in "Table 1 – Mapping of Encodings by Country."

Create your system configuration files by customizing the following default system configuration files that are supplied with the installation:

- `COMMON` option values that are used both under TSO and in batch mode
- `llxx` option values that are used with language `ll` and encoding `xx`. The member DBW0 is used for double-byte character support, for character sets such as those used in Japanese or Chinese.
- `SITE` option values customized for your site

These default system configuration files are installed into the `prefix.CONFIG` data set as samples for you to review. They contain some of the options for which you might want to establish installation-wide default values, depending on batch or interactive execution mode. However, not all these options are required.

The `prefix.CONFIG` data set, where the sample configuration files reside, is a partitioned data set with fixed-length, 80-byte records. You can create a system or user configuration file as any sequential data set or member of a PDS with any record format.

The sample configuration files contain option settings separated into logical groups by comments. Records in a configuration file are either comment lines (indicated by an asterisk in column 1) or option lines. In the sample configuration files, options are listed one per line to make them easier to read and maintain. However, this is not required; more than one option can be included on a single line. You can also add a comment anywhere between options by enclosing it between /* and */.

For options that require a value, the option must be specified as `option=value` with no blanks before or after the equal sign. Any SAS system option can be specified in the system configuration file.

Once you have entered the options in the system configuration file, no further processing is necessary (unlike options specified in the Default Options Table or Restricted Options Table). However, if you rename or move the configuration files from the installation `prefix.CONFIG` data set, be sure to update your CLIST, REXX exec, and cataloged procedure accordingly to reference the new data set names.

STEP 5: Create a Restricted Options Table (optional).

Options specified in the Restricted Options Table cannot be overridden by the user. For options in the Restricted Options Table to be processed, the restricted options module must come from a linklist library but does not have to be APF authorized.

The format of the Restricted Options Table is the same as that for the `DFLTOPTS` table, except that the option length/value pairs must be preceded by the header `***SASOPTRS***`. Like the Default Options Table, the Restricted Options Table is an assembler source module that must be assembled and linked.

The source for the sample Restricted Options Table is in the `&prefix.BAMISC` library member `SASOPTRS`. The JCL to assemble and link this table is in member `BAOPTS2` in the `CNTL` data set. The Restricted Options Table is optional. If you decide to install it, edit the `SASOPTRS` member containing the assembler source, adding the options to the table that you would like to include and removing those that

you do not want to use. Run the `BAOPTS2` job to assemble the `SASOP940` module and link it into a linklist library.

Be sure that the `SYSLMOD DD` statement in the `BAOPTS2` job points to a linklist library since the `SASOP940` module must come from a linklist library in order to be processed.

Note: *The sample table provided with this installation restricts access to VSAM data sets. You should not run this sample "as is" unless you want to restrict the use of this feature.*

SAS checks for the restricted options module, SASOP940, but it is not required. Do not create the module unless you have customized the sample contents to meet your installation requirements.

In SAS 9.4, you can apply different restricted options tables in different situations. This might be useful, for example, in a scenario that required multiple releases of SAS 9.4 to run concurrently. Modify the JCL in the `BAOPTS2` member to assemble another restricted options module. Simply change the name on the NAME statement from `SASOP940` to a name of your choice. Then specify the following in the config file:

```
OPRESTRICTIONS=name_of_your_choice
```

OPRESTRICTIONS must be specified to override the default restricted options table identifier SASOP940. Should OPRESTRICTIONS be omitted, SAS continues to assume module SASOP940 as the restricted options module table identifier.

STEP 6: Verify your default option settings.

After performing the various steps described in this section to set default option values for your site, you should run `PROC OPTIONS VALUE; RUN;` to verify that the desired defaults are in place. The z/OS-specific SMF and SVC options are not normally displayed by `PROC OPTIONS`. Specify `PROC OPTIONS VALUE GROUP=INSTALL; RUN;` to review these options.

Selecting a Bundled Configuration

Note: *You should complete this task if appropriate for your installation.*

Note: *If the Double Byte Character Set (DBCS) is to be used in conjunction with a bundled configuration, the modules that are to be renamed MUST come from the* `&prefix.DBCS.LIBRARY`*. This holds true for the non-LPA bundle configuration when using the* `BNDLSUFX=` *system option. The renamed modules are in the* `&prefix.DBCS.LIBRARY`*.*

SAS 9.4 Foundation for z/OS is distributed in two bundled configurations that are tailored for execution in the z/OS environment, and one unbundled configuration. The two bundled configurations differ in that one is tailored for execution with some modules installed in the Link Pack Area (LPA), while the other is tailored for execution with no modules installed in the LPA. Note that the code is identical across the three versions; the only difference is in the packaging.

The `ENTRY` parameter of the JCL cataloged procedure or TSO CLIST or REXX exec determines which configuration is used. The default entry point name is `SAS`, which runs the unbundled configuration. If you want to run a bundled configuration, which is highly recommended, edit your installed SAS CLIST, REXX exec and cataloged procedure to specify the appropriate entry point name for your site as described in the following section. (See "Customizing Your SAS CLIST, REXX Exec, and Cataloged Procedure" on page 3030.)

z/OS non-LPA (ENTRY=SASB)

The bundled components of the z/OS non-LPA configuration consist of the following modules located in your installed SAS LIBRARY data set: SASXA1, SABXSPL, SABXINI, SABXTRM, SABDSC, SABDSX, SABZPLM, SABZPLC, as well as others when other Institute Program Products (IPPs) are installed. SABXINI and SABXTRM are transient modules used during initialization and termination, respectively. SABDSC and SABDSX are the DATA step compilation and execution modules, respectively.

z/OS LPA (ENTRY=SASLPA)

The bundled components of the z/OS LPA configuration consist of the modules listed in the following section.

All modules that are eligible to be loaded into the LPA will reside above the 16Mb line. SASLPA is the entry point. To execute SAS using this configuration, you will need to follow the procedures outlined in the next section.

Installing SAS 9.4 Foundation into the LPA/ELPA

Note: *You should complete this task if appropriate for your installation.*

You can install the bundled modules in the LPA/ELPA. If the bundled modules are not installed in the LPA/ELPA, they are loaded into the address space of each SAS user. This can cause a significant increase in the working set size, placing a heavy burden on the paging subsystem. If you have many SAS users, this might be an important consideration.

STEP 1: Decide whether to install SAS 9.4 Foundation in the LPA/ELPA.

Contact your systems programming staff to discuss the particular considerations involved at your site. You can install only the basic supervisor bundles (SASXAL, SABXSPL, SABXDML, SABDS), or the supervisor bundles and the other bundles listed below as optional. The table shows module sizes:

z/OS Configuration

Bundled Modules for ELPA	Size	
SASXAL	3086K	
SABXSPL	5360K	(recommended)
SABODSL	3719K	(recommended)
SABXDML	1052K	(recommended; Display Manager)
SABDS	1002K	(recommended; DATA step)
SABSCLL	3077K	(optional; used by SAS/ASSIST and SCL applications)
SABDBGM	302K	(optional; SCL debugger)
SABZPLH	75K	(optional; printing routines)
SABXGPH	1694K	(optional; part of SAS/GRAPH)
Total for ELPA	**19367K**	

STEP 2: Install the modules into the LPA/ELPA using the standard procedure at your site.

STEP 3: Ensure that modules in LPA/ELPA do not have the same names as modules in the installed SAS LIBRARY data set.

Note: *STEP 3 is necessary to prevent z/OS from loading LPA/ELPA modules into the user's address space when a* `JOBLIB` *or* `STEPLIB` *DD statement (batch), or a* `LOAD` *or* `SASLOAD` *CLIST parameter (TSO) references a SAS* `LOAD LIBRARY`*.*

You can do this in one of three ways:

- Delete the bundled modules from the installed SAS LIBRARY data set.
- Rename the bundled modules in the installed SAS LIBRARY data set.
- Rename the bundled modules in the LPA/ELPA, leaving them in the installed SAS LIBRARY data set with their original names.

The third approach has two advantages. It simplifies the application of maintenance by allowing you to apply maintenance directly to the SAS LIBRARY data set. It also facilitates the concurrent running of different releases of SAS 9.4 Foundation.

To rename the bundles, take the following steps:

- Select a single character (`0-9, A-B, D-E, G, I-W, Y-Z, #, @, $`) as your suffix character. This will be the value of the `BNDLSUFX`= SAS system option. Specify `BNDLSUFX=`*character* in default CONFIG files, default options tables, or restricted options tables.
- Rename the bundles, adding the suffix character to the old name to get the new name. For example, if you choose the character `0` as your suffix character, rename the modules listed as follows:

Original Name	New Name
SASXAL	SASXAL0
SABXSPL	SABXSPL0
SABXDML	SABXDML0
SABDS	SABDS0
SABSCLL	SABSCLL0
SABDBGM	SABDBGM0
SABZPLH	SABZPLH0
SABXGPH	SABXGPH0

Note*: Your list of modules might not include all those in this example. Your list will depend on the SAS 9.4 Foundation products that you have licensed.*

If you change the name of `SASXAL`, you must specify the new name in the TKMVSENV member located in the `&prefix.TKMVSENV` data set. For example, using the example above, add the following line to the TKMVSENV member:

```
set TKOPT_LPANAME=SASXAL0
```

STEP 4: Make sure the ENTRY parameter of the JCL cataloged procedure and TSO CLIST and REXX exec defaults to the appropriate name, SASLPA.

STEP 5: If you have renamed the modules in the LPA/ELPA, use the BNDLSUFX= SAS system option to instruct SAS which set of bundled modules to use:

BNDLSUFX=character

If you use `BNDLSUFX=` for any of the bundled modules, you must rename them all, including those bundles which are added later. A mixture of bundles which have been renamed and those that have not will cause the system to fail. This applies to bundles in both the LPA/ELPA and the SAS LIBRARY data set.

System Configuration for Using SAS with TCP/IP

Planning for TCP/IP

Here are the primary configuration issues to consider when preparing your site for TCP/IP to run under the z/OS operating environment.

1. For the IBM IP Communications Server, configure or verify the host name configuration by adding a `HOSTNAME` statement in the appropriate IBM TCPIP.DATA file.

   For information about TCP/IP stacks and how to determine whether a system uses single or multiple TCP/IP stacks, see "1. Verify TCP/IP Stack Definition" on page 1919.

2. In SAS 9.4, you will use the IBM z/OS Resolver for the resolution of domain names.

3. Verify that appropriate services are configured for SAS/CONNECT or SAS/SHARE in the SERVICES file. For details, see "The Services File" on page 2929.

SAS 9.4 supports IPv4 and IPv6. For information about IPv6, see *z/OS V1R9.0 Communication Server: IP Configuration Guide*, IBM SC31-8775-11.

The steps in this section are required if you are to take full advantage of Base SAS functionality. TCP/IP must be configured if you will be using certain features, such as EMAIL, URL, Socket and FTP ACCESS methods. If you choose not to complete the steps in this section, notify SAS users that this functionality is not available.

In addition, many SAS solutions and products might require the configuration steps that are described in this section. A partial list of these products would include SAS/SHARE, SAS/CONNECT, SAS/IntrNet Software, SAS IT Resource Management, SAS Enterprise BI Server, SAS Enterprise DI Server, SAS Integration Technologies and SAS OLAP Server Software. Refer to product-specific appendices for more information.

Overview and Software Requirements

TCP is a set of routines that applications use to communicate over a network. Not all applications use TCP. However, all network applications require the services that are provided in IP. IP consists of routines that TCP calls, but the IP routines are also available to applications that do not use TCP. SAS uses both TCP and IP and requires that certain types of information be made available to the operating environment.

Although you might refer to a computer by using its host name, TCP/IP applications refer to computers by using their IP addresses. To facilitate the use of host names in a network, the Domain Name System (DNS) translates host names to IP addresses. DNS provides host-to-IP address mapping through network server hosts, which are called domain name servers. DNS also provides other information about server hosts and networks, such as the TCP/IP services that are available to the server host and the location of the domain name servers in the network.

Software Requirements

Verify that these software requirements have been met:

- The IBM z/OS V1R8 IP Communications Server TCP/IP package has been installed.
  ***Note**: SAS supports any vendor's TCP/IP software that is functionally compatible with the IBM z/OS IP Communications Server package.*
- The UNIX System Services (USS) file system is available.
- A default OE segment (or an individual OE segment for each user ID) is required and must be defined in the security software (such as RACF).
- The IBM z/OS DNS Resolver must be active.

Configuring SAS to Communicate with TCP/IP

To configure SAS to communicate with TCP/IP or to verify your configuration, the following steps ***MUST*** be completed, and the results made available to SAS:

1. Verify TCP/IP Stack Definition (below19)
2. Verify System and USS Process Limits (on page 2121)
3. Verify TCP/IP Host Name Configuration (on page 2121)
4. Verify TCP/IP Configuration Files Affecting SAS (on page 2323)
5. Host Name Resolution for Systems that Run Multiple TCP/IP Stacks (on page 2424)
6. SAS Environment Variables (en page 2727)
7. Diagnosing Configuration Problems (on page 2929)

1. Verify TCP/IP Stack Definition

TCP/IP Communication Stack Definition

TCP/IP stack is a term for the set of protocols that comprise TCP/IP. A TCP/IP Communication stack that runs under the z/OS operating environments is implemented as a UNIX System Services (USS) physical file system (PFS). An operating environment can run using one or more TCP/IP stacks. The stack definitions are located in `SYS1.PARMLIB(BPXPRM`*`nn`*`)`.

The IBM INET physical file system type supports a single TCP/IP stack. The IBM CINET physical file system type supports multiple stacks.

***Note**: If you will configure only one TCP/IP stack and you have access to both INET and CINET, it is advisable to configure the stack under INET because of its efficiency over CINET. For complete details, see the IBM documentation* z/OS UNIX System Services Planning.

Sample Definitions of TCP/IP Stacks

USS physical file systems are configured in the IBM parmlib member BPXPRMnn. These examples show typical entries in the BPXPRMnn parmlib member for INET and CINET physical file systems.

- Defining a Single IBM TCP/IP Stack

  This example shows the statements in the IBM parmlib member BPXPRMnn that define an INET physical file system for a single IBM TCP/IP stack system.

```
FILESYSTYPE TYPE(INET) ENTRYPOINT(EZBPFINI)
   NETWORK DOMAINNAME(AF_INET)
```

```
      DOMAINNUMBER(2)
      MAXSOCKETS(64000)
      TYPE(INET)
```

- Defining a Single IBM TCP/IP Stack that supports IPv4 and IPv6

  This example shows the statements in the IBM parmlib member BPXPRMnn that define an INET physical file system for a single IBM TCP/IP stack that enables IPv4 and IPv6 For details, see "About TCP/IP Internet Protocol (IP) Addressing" on page 2020.

```
FILESYSTYPE TYPE(INET) ENTRYPOINT(EZBPFINI)
  SUBFILESYSTYPE NAME(TCPIP)
  TYPE(INET)
  ENTRYPOINT(EZBPFINI)
   NETWORK DOMAINNAME(AF_INET)
      DOMAINNUMBER(2)
      MAXSOCKETS(64000)
      TYPE(INET)
   NETWORK DOMAINNAME(AF_INET6)
      DOMAINNUMBER(19)
      TYPE(INET)
```

- Defining Multiple IBM TCP/IP Stacks

  This example shows the statements in the IBM parmlib BPXPRMnn member that define a multiple TCP/IP stack system. This example includes two IBM stacks, TCPIP and TCPIP2.

  The values of the FILESYSTYPE substatements, TYPE and ENTRYPOINT, define the PFS type and the entry point for the CINET PFS. CINET requires a SUBFILESYSTYPE statement for each stack. The NAME substatement names the TCP/IP stack that is being defined. This name is also used for the Started Task that invokes the TCP/IP stack.

  ***Note**: CINET requires the NAME substatement.*

  An optional SUBFILESYSTYPE substatement named DEFAULT defines the default TCP/IP stack for a multiple stack system. If DEFAULT is not specified or if the default stack is not active, the first stack that is activated is the default stack.

```
  FILESYSTYPE TYPE(CINET) ENTRYPOINT(BPXTCINT)
     NETWORK DOMAINNAME(AF_INET)
        DOMAINNUMBER(2)
        MAXSOCKETS(64000)
        TYPE(CINET)
        INADDRANYPORT(63000)
        INADDRANYCOUNT(1000)

     SUBFILESYSTYPE NAME(TCPIP)
        TYPE(CINET)
        ENTRYPOINT(EZBPFINI)
        DEFAULT

     SUBFILESYSTYPE NAME(TCPIP2)
        TYPE(CINET)
  ENTRYPOINT(EZBPFINI)
```

About TCP/IP Internet Protocol (IP) Addressing

TCP/IP applications refer to networked computers via their fully qualified domain names (FQDN) and their IP addresses. Because IP addresses can change easily, SAS applications that contain hard-coded IP

addresses are prone to maintenance problems. To avoid such problems, use of an FQDN is preferred over an IP address. The name-resolution system that is part of the TCP/IP protocol is responsible for locating the IP address that is associated with the FQDN.

SAS 9.4 supports both IPv4 and IPv6. IPv6 uses a 128-bit address scheme, which provides more IP addresses than did IPv4.

Here are examples of an FQDN, an IPv6 address, and an IPv4 address:

```
d6292.us.company.com
db8::01
10.23.2.3
```

2. Verify System and USS Process Limits

The following IBM system values are set in the `SYS1.PARMLIB` member `BPXPRM`*nn* and affect the number of TCP/IP sockets that SAS can use.

MAXSOCKETS — system limit; specifies the maximum number of sockets that can be obtained for a given file system type. IBM recommends that this value be set to 10000.

MAXFILEPROC — process limit; specifies the maximum number of file descriptors that a single process can have open concurrently, such as all open files, directories, sockets, and pipes. This value is usually set to 256. However, for heavy server use, it is advisable to set this number to 64000.

Consult your IBM manual, *z/OS UNIX System Service Planning,* for current IBM recommendations for your z/OS version.

For z/OS versions 1.7 and above, you may consult the *IBM Health Checker for z/OS User's Guide*. This will help you identify potential problems with your settings before they impact your availability.

You can check these IBM system values by using the following commands:

`D OMVS,OPTIONS` — a z/OS command that enables you to check the value of MAXFILEPROC.

`D OMVS,PFS` — a z/OS command that enables you to check the value of MAXSOCKETS.

Note: *You can use the RACF ALTUSER or ADDUSER system commands to set MAXFILEPROC on a per-user basis. For complete details about MAXSOCKETS and MAXFILEPROC, see the IBM document "*z/OS UNIX System Services Planning*."*

3. Verify TCP/IP Host Name Configuration

IP Addresses

In order for a process to connect to a computer via TCP/IP, the process must know the IP address of the computer host name. To obtain the IP address, the process calls these name resolver functions:

`getnameinfo()` — retrieves a string that contains its host name.

`getaddrinfo()` — resolves the host name string to its IP address.

Because each host name is associated with a TCP/IP stack, it is critical that the host name be configured correctly for each TCP/IP stack.

TCP/IP Host Name Configuration for Communications Servers

When an IBM TCP/IP stack starts, the configuration process searches for the host name in the TCPIP.DATA data set.

- Search Order to Locate Stack Host Name

  1. If the IBM stack reads a TCPIP.DATA HOSTNAME configuration statement, it saves this value as the stack's host name.
  2. If a TCPIP.DATA HOSTNAME configuration statement is not read, the TCP/IP stack searches for the Virtual Machine Communication Facility (VMCF) node name from VMCF and uses its node name as the stack's host name.
     ***Note**:* *VMCF should be running before any TCP/IP stacks are started.*
  3. If VMCF is not running when the TCP/IP stack is started, the TCP/IP stack's host name is set to the CVTSNAME, which is the SYSNAME=value in IEASYSnn that was used when the system was started.

- Multiple Host Names in a Single File

  As an option, you can insert a prefix *system_name* in TCPIP.DATA configuration statements. Using prefixes enables you to configure multiple hosts in a single TCPIP.DATA data set. The *system_name* prefix is matched against the system name that the TCP/IP stack is started under. The *system_name* is identical to the VMCF node name. The TCP/IP stack reads and processes the TCPIP.DATA configuration statements in the order that they appear in the data set.

  This example shows a HOSTNAME statement in a TPCIP.DATA data set:

```
SDCMVS:   HOSTNAME   PROD
SDCESA:   HOSTNAME   TEST
S390DEVA: HOSTNAME   DEV
```

  The *system_name* is specified in the first column; the associated *host_name* is specified in the final column.

  A TCP/IP stack that was started on the system named SDCMVS would set its host name to PROD. A TCP/IP stack that started on the system named SDCESA would set its host name to TEST. A TCP/IP stack that started on the system named S390DEVA would set its host name to DEV.

  The following rules are used to process HOSTNAME statements:

  1. If the *system_name* prefix does not match a host name, the configuration statement is ignored.
  2. If the *system_name* prefix is not located, the configuration statement is applied to all hosts.
  3. If the *system_name* matches a host name, the associated configuration statement is applied to that host.
  4. The final configuration statement that matches a *system_name* remains in effect.

  A HOSTNAME statement is ignored under these conditions:

- the *system_name* prefix did not match the VMCF node name
- VMCF is unavailable

- The system name does not match the MVS name of the system that the TCP/IP stack started under

  Therefore, it is critical that VMCF is running before any TCP/IP stacks are started.

- IBM System Name Configurations

  The Virtual Machine Communication Facility (VMCF) node name is used as the *system_name* prefix when processing IBM TCPIP.DATA configuration statements. The VMCF can be configured in two ways:

  - as a restartable subsystem
    If you have configured VMCF as a restartable subsystem, the node name is obtained from the value of the P= parameter in the EZAZSSI started procedure.
  - as a non-restartable subsystem
    If you configured VMCF as a non-restartable subsystem, the node name is specified in the IEFSSNnn member of PARMLIB.

  Note: *IBM recommends that the MVS system name be used for the VMCF node name specification. For details about configuring VMCF, refer to the IBM document* z/OS Communication Server: IP Configuration Guide.

4. Verify TCP/IP Configuration Files Affecting SAS

When a TCP/IP stack is started, the TCP/IP stack reads one or more configuration files that contain statements that define its default behavior. Here are the configuration files:

IBM PROFILE.TCPIP File

The following PROFILE.TCPIP statements can restrict the ports that SAS servers can use:

PORT
Reserves ports for server tasks. The PORT statement specifies only the job names and PROC names that are allowed access to the port.

PORTRANGE
Resembles PORT parameter, but is used for a range of ports.

RESTRICT
Defines a list of user IDs that are prohibited from using TCP/IP.

RESTRICTLOWPORTS
Restricts the use of ports 1 to 1023 to specific job names or PROC names that are specified in the PORT or the PORTRANGE statement.

For details about the IBM PROFILE.TCPIP statements, see "References" on page 2929.

The search order that is used by the IBM TCP/IP stack to locate PROFILE.TCPIP involves both explicit and dynamic data-set allocation, as follows:

1. //PROFILE DD DSN=
2. jobname.nodename.TCPIP
3. hlq.nodename.TCPIP

4. jobname.PROFILE.TCPIP
5. TCPIP.PROFILE.TCPIP

Note: *IBM recommends explicitly specifying the PROFILE DD statement in the TCPIPROC JCL.* When the PROFILE DD statement is specified, no dynamic allocation is performed.

SAS does not access the PROFILE.TCPIP file directly. However, because this file is used to configure the IBM TCP/IP stack, the statements in this file can have an indirect effect on how SAS operates.

IBM TCPIP.DATA File

The TCPIP.DATA file contains the following statements that are used to configure the IBM TCP/IP stack and Communication Server applications.

TCPIPJOBNAME (or TCPIPUSERID)
Specifies the member name of the procedure that is used to start the TCPIP address space, which is the TCP/IP stack name.

HOSTNAME
Is used by the TCP/IP stack to determine its host name.

DOMAINORIGIN (or DOMAIN)
Specifies the name of the domain origin, which is appended to the host name to form the fully-qualified domain name of the host.

DATASETPREFIX
A high-level qualifier (hlq) for the dynamic allocation of data sets in IBM TCP/IP servers and clients.

For details about the IBM TCPIP.DATA statements, see "References" on page 2929.

The IBM TCP/IP stack and the IBM Communication Server applications, including the IBM z/OS Resolver, use the following search order to locate the data set that contains the TCPIP.DATA configuration statements:

1. GLOBALTCPIPDATA value
2. RESOLVER_CONFIG environment variable
3. `/etc/resolv.conf`
4. SYSTCPD DD
5. *userid*.TCPIP.DATA
6. SYS1.TCPPARMS(TCPDATA)
7. DEFAULTTCPIPDATA value
8. TCPIP.TCPIP.DATA

5. Host Name Resolution for Systems that Run Multiple TCP/IP Stacks

There is usually a one-to-correspondence between a TCP/IP stack and its host name. The host name is obtained using the `gethostname()` function. However, for systems that run using multiple TCP/IP stacks, the identity of the stack's host name is ambiguous. Here is the process for resolving the host name for a multiple TCP/IP stack system:

- The process that calls `gethostname()` might be bound to a specific TCP/IP stack. The binding is referred to as TCP/IP stack affinity. If affinity to a specific TCP/IP stack has been established, the `gethostname()` function will return the host name for the specific stack.

There are several methods of setting the TCP/IP stack affinity. SAS uses the UNIX System Services call, `pfsctl(BPX1PCT)`. For details, see "Determining SAS TCP/IP Stack Affinity" below.

- Otherwise, the process returns the host name of the default TCP/IP stack. For details about the default stack, see "1. Verify TCP/IP Stack Definition" on page 1919.

Determining SAS TCP/IP Stack Affinity

The SAS TCP/IP interface uses the UNIX System Services call pfsctl(BPX1PCT) to determine whether the z/OS system is running a single TCP/IP stack environment (INET) or a multiple TCP/IP stack environment (CINET). Here is the process:

- If the `pfsctl()` call returns zero, the z/OS System is running an INET environment.
- If the `pfsctl()` call returns the number of CINET TCP/IP stacks and their names, the z/OS System is running a CINET environment.

1. The SAS TCP/IP interface searches for the SAS environment variable, TCPIPMCH, in the data set that is specified by the DDNAME, TKMVSENV.

   For details about specifying the TCPIPMCH environment variable and the TKMVSENV data set, see "6. SAS Environment Variables" on page 2727.

   - If the TCPIPMCH environment variable is set to a value of "*", the SAS TCP/IP interface will not attempt to set the TCP/IP stack affinity.
   - If the TCPIPMCH environment variable is set to a valid stack name, the SAS TCP/IP interface will set the TCP/IP stack affinity to this value.
   - Otherwise the TCP/IP stack affinity will be set to the first TCP/IP stack name that was returned by the previous call to `pfsctl()`.

2. The SAS TCP/IP interface resets the stack affinity by making another call to `pfsctl()` using the appropriate flags and specifying the TCP/IP stack name.

The value that is passed to `pfsctl()` must match the value of the NAME substatement, which is included in the SUBFILESYSTYPE statement, which is defined in the CINET PFS TCP/IP stack in the IBM BPXPRMxx parmlib member.

A name resolver is a set of routines that act as a client on behalf of an application to read a local host file or to access one or more domain name servers (DNS) for name-to-address or address-to-name resolution. Name resolution occurs by calling the name resolver functions `gethostbyname()` and `gethostbyaddr()`. A name resolver must be configured for each host.

IP address-to-host name mapping is performed by the Domain Name System if the `NSINTERADDR` statement(s) in the `TCPIP.DATA` file point to the IP address(es) of the Name Servers.

The following keyword/value combinations are involved in resolving host names in the TCP/IP Data file:

`NSINTERADDR`	value; Required
`NSPORTADDR`	value; Optional, will use default values
`RESOLVEIA`	value; Optional, will use default values
`RESOLVERTIMEOUT`	value; Optional, will use default values
`RESOLVERUDPRETRIES`	value; Optional, will use default values

Note: *NSINTERADDR is the only one that needs to be verified. The defaults for the others are sufficient for SAS to function properly.*

Note: *If Name Servers are not being used for Host Name Resolution, the* NSINTERADDR *statement(s) would be commented out.*

Using a Host Table

Using an FQDN with a Host Table

Note: *An administrator must make this change.*

The system host table must be defined such that an FQDN is used for each local network interface so that all clients will be able to use remote browsing.

For IPv4, SAS uses the ETC.HOSTS file for address resolution. This file lists IP addresses with names and aliases for each IP address. Only the entries for the addresses of the local network interfaces need to be changed.

ETC.HOSTS entries have the syntax (keywords in bold):

```
ipaddress          hostname alias alias
```

For the system to use the FQDN as the host name, the first entry would be the FQDN, with aliases following as partial host names. For example:

```
10.4.86.63  sashost
```

becomes

```
10.4.86.63  sashost.mycompany.com sashost
```

For IPv6, hostnames are contained in the data set ETC.IPNODES, which has the format:

```
Address Hostname1 HostName2 HostName3 …. Hostname35
```

Using the FQDN is similar to IPv4, with the first host name specified as the FQDN. For example:

```
1:2:3:4:5:6:7:8 sashost
```

becomes

```
1:2:3:4:5:6:7:8 sashost.mycompany.com sashost
```

Refer to the IBM document *z/OS Communications Server: IP Configuration Guide* for more information on configuring host names.

If a DNS server is not available for Host Name Resolution, a Host Table can be used. However, SAS cannot use the IBM TCP/IP file 'tcpprefix.HOSTS.LOCAL' or the 'tcpprefix.HOST.SITEINFO' due to the format of the data. See the following list for the default SAS search order of finding a HOSTS table:

1. ETC_HOSTS environment variable
2. //hfs:/etc/hosts z/OS (integrated sockets only)
3. tso-prefix.ETC.HOSTS under TSO
4. ETC.HOSTS
5. tcpip-prefix.ETC.HOSTS, if TCPIP_PREFIX is not blank.

To use a Host table that is not found with the default search order, use SAS environment variables. For details on setting up environment variables, see "6. SAS Environment Variables" on page 27.

6. SAS Environment Variables

TCPIPMCH Environment Variable

Environment variables, which are specified in the file to which the DDNAME TKMVSENV points, are used to customize TCP/IP for SAS 9.4. SAS 9.4 uses the TCPIPMCH SAS environment variable to alter default processing for TCP/IP initialization.

The TCPIPMCH SAS environment variable is useful at sites that simultaneously run multiple TCP/IP packages: either multiple TCP/IP vendor packages or multiple instances of the same vendor's TCP/IP. The TCPIPMCH environment variable is used to specify the name of the TCP/IP stack name, such as a started task. Setting this environment variable is the equivalent of the TCPIPJOBNAME/TCPIPUSERID configuration keywords within the IBM TCPIP.DATA file. If the default value for the TCP/IP stack name is not specified, the value is the first TCP/IP stack that is defined to the system.

For more information about TCPIPMCH, see "Configuring SAS to Use the IBM z/OS Name Resolver " on page 2828.

TKMVSENV Data Set

A SAS data set can be used to specify the SAS environment variables, which is referred to as the TKMVSENV data set file. If you use SAS environment variables, you must allocate the TKMVSENV DD in the JCL or CLIST that executes SAS 9.4.

For example, here are the allocation statements for the data set SAS.DATA.TKMVSENV, which contains the desired environment variable information:

Note: *Line numbering in the TKMVSENV data set must be disabled.*

BATCH statement:

```
//TKMVSENV DD DISP=SHR,DSN=SAS.DATA.TKMVSENV
```

TSO statement:

```
ALLOC F(TKMVSENV) DA('SAS.DATA.TKMVSENV') SHR
```

Each logical record contains an environment variable assignment in this format:

SET environment_variable_name=value

TCP/IP Name Resolver Configuration

A name resolver is a set of routines that act as a client on behalf of an application to read a local host file or to access one or more domain name servers (DNS) for name-to-address or address-to-name resolution. Name resolution occurs by calling the name resolver functions `getnameinfo()` and `getaddrinfo()`.

A name resolver must be configured for each host. Here are locations of UNIX configuration files:

- `/etc/hosts` contains the local host configuration data
- `/etc/resolve/conf` contains the DNS domain name and the name servers' IP addresses
- `/etc/service` contains the service configuration data

IBM z/OS Name Resolver

Starting with z/OS V1R4, IBM introduced support for IPv6. New USS BPX calls for the IBM z/OS Name Resolver were introduced to implement the protocol-independent resolver functions that are described in the RFC 3493 specification. Here is a list of the IBM z/OS Name Resolver functions that are supported in IPv6 and IPv4:

IPv6	IPv4
`getnameinfo(BPX1GNI)`	`gethostbyname()`
`getaddrinfo(BPX1GAI)`	`gethostbyaddr()`

When the IBM z/OS Name Resolver is started, it reads the IBM configuration file that is pointed to by the DD statement SETUP, which can contain the following SETUP directives.

- COMMONSEARCH | NOCOMMONSEARCH
- DEFAULTIPNODES
- DEFAULTTCPIPDATA
- GLOBALIPNODES
- GLOBALTCPIPDATA

***Note**: The most important SETUP directives are GLOBALTCPIPDATA and DEFAULTTCPIPDATA*

GLOBALTCPIPDATA
Identifies a global TCPIP.DATA file. Any TCPIP.DATA directive that is specified in this file are system-wide and cannot be overridden by a local TCPIP.DATA file.

DEFAULTTCPIPDATA
Identifies a default TCPIP.DATA file, which overrides the TCPIP.DATA file that is named TCPIP.TCPIP.DATA.

If a GLOBALTCPIPDATA statement is located in the resolver setup file, the IBM z/OS Name Resolver will read any name resolver directives that are located in this global TCPIP.DATA file. The IBM z/OS Name Resolver will then search for a local TCPIP.DATA file in this order:

1. RESOLVER_CONFIG environment variable
2. `/etc/resolv.conf`
3. SYSTCPD DD
4. `jobname.TCPIP.DATA`
5. `SYS1.TCPPARMS(TCPDATA)`
6. DEFAULTTCPIPDATA value (if specified in the z/OS Name Resolver setup file)
7. `TCPIP.TCPIP.DATA`

Configuring SAS to Use the IBM z/OS Name Resolver

SAS 9.4 uses the IBM z/OS Name Resolver by default. No configuration is necessary unless SAS is running under a multiple TCP/IP stack system (CINET). If SAS is running under a multiple TCP/IP stack system, the SAS TCP/IP interface will need to set the TCP/IP stack affinity. Specify this SAS environment variable:

```
set TCPIPMCH=stack-affinity
```

TCPIPMCH is a SAS environment variable that is used to specify the name for the TCP/IP stack, which is also known as a started task. TCPIPMCH is equivalent to the TCPIPJOBNAME and TCIPPUSERID configuration keywords that are used in the IBM TCPIP.DATA file.

If a value is not specified for TCPIPMCH, the SAS TCP/IP interface will use the first TCP/IP stack name that is returned by a call to the UNIX System Service, pfsctl(), which might not be the appropriate TCP/IP stack. For more information, see "TCPIPMCH Environment Variable" on page 2727.

The Services File

The SERVICES file defines port resources that are used when TCP/IP is used to connect client/server sessions. Examples of port services include the Telnet port, spawner ports, MP CONNECT pipes, and SAS/SHARE servers. For more information, see "Configuring the Services File" in the *SAS/CONNECT 9.4 User's Guide*.

Configure SAS Services

A service for each SAS server session (SAS/CONNECT or SAS/SHARE) must be defined in the SERVICES file on each machine where a SAS client session runs.

***Note**: Some TCP/IP stacks can restrict the range of available ports. For details, see TCP/IP stack configuration files.*

Sample TCP/IP SERVICES File

Some SAS products (those that use a server) require an entry in the TCP/IP `SERVICES` file. See the individual product requirements to verify the need for an entry in the TCP/IP SERVICES file.

Entries in the ETC.SERVICES file will follow the sample form below:

```
#   this is a comment in the ETC.SERVICES file
#
telnet   23/telnet
ftp      21/tcp
sassrv1  5010/tcp      #   SAS/SHARE Server number 1
sassrv2  5011/tcp      #   SAS/SHARE Server number 2
appsrv1  5224/tcp      #   SAS/IntrNet Application Server number 1
appsrv2  5225/tcp      #   SAS/IntrNet Application Server number 2
spawner1 5227/tcp      #   z/OS SAS/Connect Spawner
```

The Services File Search Order

The z/OS Resolver searches for the SERVICES file, using this order:

1. Value of the ETC_SERVICES environment variable
2. `/etc/services`
3. `tso-prefix.ETC.SERVICES` under TSO or `user-ID.ETC.SERVICES` under batch execution
4. `ETC.SERVICES`
5. `TCPIP.ETC.SERVICES`
6. `tcpip-prefix.ETC.SERVICES`

7. Diagnosing Configuration Problems

Executing the SAS code "`PROC TCPTEST;run;`" provides information about how SAS sees the TCP/IP configuration.

References

- z/OS V1R9.0 Communication Server: IP Configuration Guide, IBM SC31-8775-11
- z/OSV1R9.0 Communication Server: IP Configuration Reference, IBM SC31-8776-12
- z/OS V1R9.0 UNIX System Services Planning, IBM GA22-7800-12
- DNS and BIND, 4th Ed., by Paul Albitz & Cricket Liu, O'Reilly and Associates, Inc.

Customizing Your SAS CLIST, REXX Exec, and Cataloged Procedure

Note: *Complete this task if appropriate for your installation.*

For details about customizing the REXX exec, refer to Appendix 2, "Starting SAS with SASRX," in the *SAS 9.4 Companion for z/OS*.

There are CLIST, REXX exec, and PROC examples created in the `&PREFIX.INSTALL.CNTL` data set for invoking SAS. You can invoke the encoded SAS using `CLSTllxx`, `REXXllxx`, or `PROCllxx` where *ll* is the language and *xx* is the encoding value that can be found in "Languages and Installation Codes" on page on page 43. (All installations of SAS, whether domestic or foreign, are now encoded images. Encoded images enable SAS to be invoked using different character sets. For example, W0 is English and W3 is German.)

The installation job(s) that you ran to complete the installation created tailored versions of the `SAS CLIST`, `REXX` exec, and cataloged procedures.

- The tailored CLISTs were written to the `&PREFIX.INSTALL.CNTL` data set as member `CLSTllxx` for the single-byte image and `CLSTDBxx` for the double-byte image, and they were also written to the command procedure library, `&prefix.CLIST`.
- The tailored cataloged procedure was written to the `&PREFIX.INSTALL.CNTL` data set as member `PROCllxx` for the single-byte image and `PROCDBxx` for the double-byte image. The installation job(s) copied it to the procedure library, `&prefix.PROCLIB`.
- The tailored REXX execs were written to the CNTL data set as member `REXTllxx` and copied to the `SASRX` data set as member `REXXllxx`. Configuration files for the REXX execs were written to the `&PREFIX.INSTALL.CNTL` data set as members `REXXCFG`, `REXXDB`, `REXXMSG`, and `REXXSITE`, and copied to the `SASRXCFG` data set as members with the same names (this is a difference from the way that `SASRXCFG` was structured in SAS 9.2). SAS recommends that you never modify `REXXCFG`, `REXXDB`, or `REXXMSG`, but instead make any site customizations in `REXXSITE`.

  Note that the `REXXllxx` exec is actually just a stub that passes parameters and the names of configuration files to another REXX exec named `SASTREXX` that contains the greater part of the code. You need only one copy of `SASTREXX` but it can be shared by multiple instances of the `REXXllxx` code, which also share the configuration files in the `SASRXCFG` data set. The members of the `SASRXCFG` data set have the high-level qualifiers coded symbolically as "`&hlq.`". This symbol is resolved from the hlq value that is set in the `REXXllxx` exec, so if you need to move your installation to different high-level qualifiers, the `REXXllxx` exec is the only place where you need to change the name.

Note: *Only Action A moves the CLIST, REXX execs, and cataloged procedure into the default libraries. Actions B and C require that you manually copy and rename* `CLSTllxx`, `REXTllxx`, `PROCllxx`, *and the members of the SASRXCFG data set to the appropriate libraries.*

To make the CLIST generally available, you will need to include the `&prefix.CLIST` data set in your SYSPROC concatenation or else copy the members to a data set that is in the concatenation.

To make the REXX exec generally available, you will need to include the `&prefix.SASRX` data set in either your SYSPROC or SYSEXEC concatenation, or else copy the members to a data set that is in one of the concatenations.

You might want to rename the CLIST from `CLSTllxx` to SAS and to rename the REXX exec from `REXXllxx` to SASRX for consistency with names used in the *SAS 9.4 Companion for z/OS*.

Placing SAS Modules into LINKLIST or LPA

SAS load modules and program objects should not be placed into LPA or LINKLIST unless this document specifically identifies them as eligible. The SAS `SVC`, the `SASVXMS` and `SASCP` modules, the restricted options table, and the bundle modules, including any affected by the `BNDLSUFX` option, are examples of modules that are eligible for LINKLIST or LPA.

You can further customize these tailored versions of the CLIST, REXX exec, and cataloged procedure. For example, the CLIST includes statements that allocate a permanent `SASUSER` data library for each user. If no permanent `SASUSER` data library exists for the user, the CLIST creates one. If you do not want each user at your site to maintain an individual permanent `SASUSER` data library, you can remove these statements from the CLIST. You might also want to make changes as part of selecting a bundled configuration to run or as part of installing SAS 9.4 Foundation in the LPA/ELPA, as previously described. STEP 1 in this section describes some of the changes you might want to make.

Important: *If you are using the REXX exec, many customizations can be made in the REXXSITE configuration file in the SASRXCFG data set, without a requirement to change the REXX code. Options that can be set here are documented in Appendix 2, "Starting SAS with SASRX," in the SAS 9.4 Companion for z/OS. And the user exit feature provides extensive additional customization capability, also documented in Appendix 2.*

Determine the changes you want to make as described in STEP 1 and apply the changes according to your standard procedures. Make the changes to the CLIST, REXX exec, and cataloged procedure in the libraries to which they were copied.

STEP 1: Determine the customizations you need.

Review the following information that discusses changes you might want to make. You might also have some site-specific issues to address.

- Product-specific customizations

  The appendices describe product-specific customizations that might be required. Review the appendices for the products you are installing to see what customizations to the SAS CLIST, REXX exec, and PROC you need.

- SASUSER Considerations

  The SAS 9.4 Foundation CLIST allocates a permanent `SASUSER` data library for each user the first time the user invokes SAS 9.4 Foundation. When there is no `SASUSER` data library allocated to a session, the system by default assigns the `SASUSER libref` to the temporary `WORK` data library. In this case, data written to `SASUSER` disappears when the `WORK` data library is deleted.

  Although individual `SASUSER` data libraries are not required, they allow users to take advantage of many interactive features in SAS 9.4 Foundation. The SAS/ASSIST product uses the `SASUSER` library to store all SAS data sets created and to save all work (programs, output, and catalogs) from a SAS/ASSIST session. The SAS windowing environment uses this library for storing various types of information. Users can save profiles in their `SASUSER` data libraries to customize window sizes, function key settings, and other aspects of the SAS full-screen environment. You can use SAS windowing environment `SAVE` and `COPY` commands to transfer data or program statements

between windows and catalogs in your SASUSER data library. The FORMS command stores forms entries used in printing from the windowing environment.

The default CLIST creates a permanent SASUSER data library for each user who invokes SAS 9.4 Foundation using the CLIST statements, as shown in the following:

```
IF &SYSPREF EQ THEN +
   SET &USRPREF=&SYSUID
ELSE +
SET &USRPREF=SYSPREF
IF &STR(&SASUSER) EQ THEN +
   SET &SASUSER = &STR('&USRPREF..SAS9.SASUSER')
SET STATE = &SYSDSN(&SASUSER)
IF &STATE = OK  THEN +
    ALLOC F(&DDSASUSR) DA(&SASUSER) OLD REU
ELSE +
IF &STATE = DATASET NOT FOUND THEN DO
    WRITE Warning: SASUSER file does not exist, will be created.
    ALLOC   F(&DDSASUSR) DA(&SASUSER) NEW CATALOG +
            SP(30 5) ROUND DSORG(PS) RECFM(F S) +
            BLKSIZE(6144) REU
      END
ELSE DO
      WRITE Warning: SASUSER file: &STATE
      WRITE        SASUSER file not allocated to this session
      END
```

You might want to determine a different naming convention for SASUSER data libraries or alter the default space allocation.

The REXX exec creates a permanent SASUSER data library for each user according to the specification of the SASUSER option in the SASRXCFG(REXXCFG) file. Here is the default:

```
-SASUSER '&syspref.SAS9.SASUSER'
```

where &syspref is replaced with the user's system prefix, or if the prefix is null, with the userid. You can override this default specification in the SASRXCFG(REXXSITE) file with a –SASUSER option that meets your requirements, or with –NOSASUSER if you do not want to use SASUSER libraries.

Special Cataloged Procedure Parameters

The default cataloged procedure includes two symbolic parameters that allow concatenation of user libraries before your SAS installation libraries:

- LOAD=
  The LOAD= parameter allows you to specify a user load library DSN to concatenate before the SAS load library data set.

- SASAUTO=
  The SASAUTO= parameter allows you to specify a user autocall macro library DSN to concatenate before the system autocall macro library.

  These parameters are intended to provide added flexibility for invoking SAS in batch mode. Any existing JCL that you use to invoke earlier releases of SAS might not work as expected with the new default PROC. If large volumes of production JCL at your site contain DD overrides for

`//SASAUTOS` and `//STEPLIB`, you might want to customize the `SAS9 PROC` to change the concatenation order of the data sets for these DD statements.

- **Entry Name Considerations**

  If you plan to run a bundled configuration, change the default `ENTRY` name in your CLIST and cataloged procedure. If you are using the REXX exec, override the default with a -ENTRY option in your SASRXCFG(REXXSITE) config file, not in the REXX exec. The default is SAS. Valid standard `ENTRY` names are as follows:

  - SAS for z/OS non-bundled configuration
  - SASB for z/OS non-LPA bundled configuration
  - SASLPA for z/OS LPA bundled configuration.

  See "Selecting a Bundled Configuration" on page 1515 and "Installing SAS 9.4 Foundation into the LPA/ELPA" on page 1616 for considerations relating to `ENTRY` name selection.

- **Running Multiple Versions of SAS 9.4 Foundation Concurrently**

  If your users run multiple versions of SAS concurrently in the same TSO session, you might also want to customize the CLIST to avoid `DDname` conflicts.

  To do so, determine a naming convention for the SAS file `DDnames` allocated, such as `WORK`, and specify your `DDnames` in the CLIST. The CLIST includes special `DDname` operands for you to use when specifying your SAS file `DDnames`. When you use these operands, the CLIST specifies the appropriate corresponding SAS system options for you. Specify alternate `DDnames` in this manner for any files that would cause conflicts at your site.

  If you are using the REXX exec, add the alternate DDname options to the SASRXCFG(REXXSITE) config file, not in the code of the exec.

- **Additional DDname Considerations**

  With SAS 6 and later, SAS no longer uses the FORTRAN-style `DDnames` that were used in Version 5 for the SAS log, print, and `PARMS` data sets. If you want to use the same `DDnames` in SAS 9.4 as in Version 5, customize your CLIST, PROC, and system default options.

  Customize the CLIST and PROC by changing the `DDnames` as follows:

  - change SASLOG to FT11F001
  - change SASLIST to FT12F001
  - change SASPARM to FT15F001.

  Customize your default options by adding the following option values to your `DFLTOPTS` table or system configuration file.

  - LOG=FT11F001
  - PRINT=FT12F001
  - PARMCARDS=FT15F001

  See page 13 of the section "Setting up SAS 9.4 Foundation" for details on customizing your `DFLTOPTS` table and system configuration file.

- **SORT Library Considerations**

  If your users run `PROC SORT` and your site does not provide your system sort routine in a linklist library, set the `SORTLINK CLIST` operand to null and specify the load library that contains your system sort routine in the `SORTLDSN` operand. If you are using the REXX exec, add these options in the SASRXCFG(REXXSITE) config file. In the cataloged procedure, concatenate your system sort load library to the `STEPLIB DD` statement.

STEP 2: Make the CLIST and PROC changes according to standard procedures at your site.

Customizing Your NEWS File

Note: *You should complete this task if appropriate for your installation.*

The installation process unloads the default `NEWS` member into your `&prefix.NEWS` partitioned data set. You can update this member with information appropriate for your site.

Information contained in the `NEWS` data set is displayed on the SAS log at invocation time when the `NEWS=` SAS system option is specified. The `NEWS=` option specifies either a logical or a physical name for the `NEWS` data set. The default system configuration files loaded into your `prefix.CONFIG` data set at installation time contain the `NEWS=` system option specifying the physical name of the `NEWS` member in the `NEWS` data set allocated by the installation jobs. You can modify the `NEWS` member to contain any information appropriate for your site, or, if you do not want to display standard information at invocation time, you can remove the `NEWS=` option from your system configuration files.

Installing the SAS 9.4 SVC Routine

Note*: This task is optional, but recommended.*

STEP 1: Decide whether to install the SAS 9.4 Foundation SVC routine.

In most cases, where the function provided or supported by the SVC routine is not utilized or required, installation of the SAS SVC routine is ***not*** absolutely necessary. However, this step should be completed for the following reasons:

- At a later time, you might decide to use the otherwise unavailable functions.
- You might purchase a SAS product that requires the SAS SVC routine.

Installation of the SAS SVC is **absolutely required** in any of these six situations:

- The SMF SAS system option is used to write SAS user SMF records.
- SAS/SHARE 9.4 is installed.
- The SAS/IntrNet product is installed and `PROC APPSRV` is used with the `AUTH=HOST` option.
- The SAS/CONNECT Spawner for z/OS or SAS/CONNECT to the same multi-process machine on z/OS is to be implemented.
- The SAS/Integration Technologies Object Spawner is installed.
- You are installing either the OLAP server or the OMR server in SAS 9.4 Foundation and you will be running these servers with the SECURITY option enabled (the default).

The SAS SVC routine provides all the functions available with the SAS SVC in previous releases and can be used in place of SAS SVC routines in previous SAS releases. However, SAS 9.4 is not compatible with prior releases of the SAS SVC routine. Therefore, if the SAS SVC routine is to be used with SAS 9.4, the SAS 9.4 SVC routine must be installed.

z/OS System Integrity Considerations

Some features of SAS 9.4 Foundation for z/OS require the use of z/OS authorized facilities. Access to these facilities has been designed and coded using IBM guidelines for system integrity. SAS is committed to preserving z/OS system integrity and will resolve any reports of system integrity exposures resulting from use of SAS software.

The SAS SVC provides controlled access to z/OS authorized facilities. The SVC that is delivered with each SAS release is compatible with older releases, and its use is supported with any older SAS release that is still in Standard support.

The SAS SVC might be in the system LINKLIST and can be loaded into the System LPA without affecting z/OS system integrity.

SAS provides the SASSVCUP program that allows z/OS System staff to install the SAS SVC without requiring an IPL. It is delivered in an unauthorized state. The z/OS system staff must take action to enable it in order to install the SVC, and it is their responsibility to ensure the program is then restricted to authorized staff. Additional details on installing SASSVCUP follow.

For more information, see:

- Chapter 21 of *MVS Authorized Assembler Services Guide* (SA22-7608): protecting the z/OS system.
- IBM's z/OS system integrity statement:
  http://www-03.ibm.com/systems/z/os/zos/features/racf/zos_integrity_statement.html

STEP 2: Select the type of SVC to install.

The SAS 9.4 Foundation SVC routine can be installed in one of two ways:

- As a Type 4 Extended Support Router SVC (ESR SVC 109) entry. SAS recommends that the SAS SVC routine be installed as a Type 4 ESR SVC (SVC 109). This technique has several advantages. One is that a user SVC reserved exclusively for the SAS SVC is not required. In addition, if the Type 4 ESR SVC routing code is selected for use by another software vendor, it is relatively easy to change the routing code used by the SAS SVC.
  To select the Type 4 ESR SVC (SVC 109) routing code to use, first determine which routing codes are already installed or in use by the operating system or other software products. To do this, list the names of members beginning with `IGX00` in the `SYS1.LPALIB` and all other libraries listed in the `LPALSTxx` member of `SYS1.PARMLIB`. Also check the `IEALPAxx` member of `SYS1.PARMLIB` for `IGX00nnn` modules that can be placed in MLPA. The `nnn` suffix is the routing code (always in decimal) by which the ESR SVC routine is invoked. IBM has reserved routing codes between 200-255 for customer use under z/OS. SAS recommends that you choose a routing code within this range. It is necessary to select an unused routing code to avoid interfering with the operation of other software already using another routing code.

- As a standard "user" SVC (SVC Routines 200-255) defined in member `IEASVCxx` of `SYS1.PARMLIB`. To install the SAS SVC as a "user" SVC routine, ensure that the selected user SVC number is currently unused. Check the `IEASVCxx` member in `SYS1.PARMLIB`. For example, to install the SAS SVC as SVC 200, code the following `SVCPARM` statement in `IEASVCxx`:

  ```
  SVCPARM 200,REPLACE,TYPE(4)
  ```

  The SAS SVC is installed as a Type 4, preemptive, unauthorized SVC with no locks held.

  The first "load" of a Type 4 SVC routine is named according to z/OS convention. That is `IGC00nnc`, where nnc is the zoned EBCDIC representation of the SVC routine's number, resulting

from the unpacking of the positive, packed decimal value that is the SVC routine number. For example, the first load of a Type 4 SVC routine invoked using SVC 234 would be named IGC0023D. This is because x'234', when unpacked, yields x'F2F3C4' or C'23D'.

STEP 3: Copy and Rename the SAS 9.4 Foundation SVC routine into SYS1.LPALIB or a LNKLSTxx library.

Copy and rename the SAS 9.4 Foundation SVC into SYS1.LPALIB or any other LPA library pointed to by the LPALSTxx member of SYS1.PARMLIB. Optionally, the SAS 9.4 Foundation SVC can be installed into a LNKLSTxx library and brought into LPA at IPL time by a specification in the IEALPAxx member of SYS1.PARMLIB and an MLPA=xx specification in the IEASYS00 member of SYS1.PARMLIB.

The load module (SVC0MVS) must be installed into SYS1.LPALIB (or other appropriate library) with a valid SVC name. A utility like IEBCOPY could be used to copy the SAS 9.4 Foundation SVC routine into an appropriate LPA library. Refer to member SVC0CPYJ in your &prefix.BAMISC SAS installation library for a sample job.

As an alternative, SMP/E can be used to install the SAS 9.4 Foundation SVC as an SMP/E USERMOD. Refer to member SVC0SMPJ in your &prefix.BAMISC SAS installation library for a sample job.

You must IPL after the SAS 9.4 Foundation SVC routine is actually copied into an appropriate operating system library. Specify the CLPA parameter in response to the IEA101A message.

To proceed with the SAS installation without waiting for an IPL to install the SAS SVC, you can install it dynamically.

To dynamically install the SAS SVC, you must first install it into the system Dynamic LPA using the system SETPROG command. Only system operators and system programmers may issue this command. An example of this command would be as follows:

- For an ESR SVC - SVC0SVC=109 SVC0R15=220:

  ```
  SETPROG LPA,ADD,MODULE=IGX00220,DSNAME=SAS.LIBRARY,SVCNUMDEC=(109,220)
  ```

- For a user SVC - SVC0SVC=220:

  ```
  SETPROG LPA,ADD,MODULE=IGC0022{,DSNAME=SAS.LIBRARY,SVCNUMDEC=220
  ```

You must use a utility program to tell the system to use the SAS SVC module and by what SVC and optionally ESR number to invoke it. Freeware versions of such utilities are available, on the CBTTAPE website for example (http://www.cbttape.org/). SAS has also provided a similar utility called SASSVCUP. These utilities use the system SVCUPDTE service to dynamically add an SVC. This is an authorized system service, so the program invoking it must be running in authorized mode.

Installing SASSVCUP

Before you can use the SAS provided utility, you will need to mark it APF-authorized and move or install it into an APF-authorized library. This library should be protected so that only appropriate individuals may execute the SASSVCUP program; otherwise, any user might execute it and do damage to the system by overlaying existing SVC routines.

To install the SASSVCUP utility, you may either use the provided SMP/E job or you may simply relink the module to set the AC=1 flag and copy it to an authorized library. A sample job has been provided for this as well.

Note: *You can also use any other utility such as the PDS command to set the AC=1 flag.*

To relink the module, edit and run the job SSVCULNK in the `&prefix.BAMISC` library. The SASLOAD DD statement should point to the existing SAS library containing the SASSVCUP module. The SYSLMOD DD statement should point to a RACF-protected APF-authorized library.

To use SMP/E to install the job, you will need to edit the members SVCISMP and SVCISMPJ in the `&prefix.BAMISC` library. The first member, SVCISMP, is the SMP/E MCS. You will need to set the proper fmid for your system and pick a usermod name. The second member, SVCISMPJ, is a sample job for running SMP/E. You will need to edit the JCL to suit your site; you will also need to point to the correct libraries, and specify the usermod name that you picked. The JCL comments describe exactly what needs to be changed in both members.

Running SASSVCUP

To run SASSVCUP, see the example job SASSVCUP in the `&prefix.BAMISC` library. Edit the parameter list on the EXEC statement. The first parameter is the SVC number to be used to invoke the SAS SVC (the same value will be specified as the SVC0SVC= option). The second parameter is the load module name of the SAS SVC. If the SAS SVC is to be installed as an Extended Routing SVC, the load module name must conform to the standard IGX00*nnn*, where *nnn* is to be the Extended Routing index (the same value that will be specified for the SVC0R15= option). The parameters are separated by commas, and no spaces are allowed.

You can test the SASSVCUP job by running it from a non-authorized library. If it is run non-authorized, it cannot invoke the SVCUPDTE service and so cannot make any changes to the system. It will validate the parameters and search the LPA for the specified module name. All messages are written to the job console log. When SASSVCUP is run authorized, it invokes the SVCUPDTE service; if the add request is successful, it sends a message to both the job console log and to the system log (ROUTCDE=2,DESC=11) so that the change to the system is logged.

STEP 4: Verify and update SAS options for the SAS SVC.

If the default values are not used, these options need to be set in the Restricted Options Table. See "Customizing Default Options and System Configuration Files" on page 99 for details on creating a Restricted Options Table.

The following SAS system options are directly related to the SAS SVC routine and how it is installed. You must set these options as follows in order to invoke the SAS SVC routine correctly:

- **SVC0SVC=**

  The default is `109` for the ESR SVC 109. If using the "user" SVC instead of the ESR SVC, this option should be set to the SVC number that was defined in STEP 2.

- **SVC0R15=**

  This option only applies if the SAS 9.4 Foundation SVC was installed as an ESR Type 4 SVC. The default is `4` for compatibility with previous releases of SAS 9.4 Foundation. It should specify the routing code that was chosen when the SAS 9.4 Foundation SVC was installed into your operating system.

STEP 5: Verify and update SAS 9.4 options for the SAS TK SVC.

Note: *This section is for the SAS/CONNECT Spawner, the Integration Technologies Object Spawner, and any product that requires these two products.*

If the default values are not used, the options must be set in the `'&prefix.TKMVSENV(TKMVSENV)` data set with the file reference of `TKMVSENV`.

The following SAS TK options are directly related to the SAS 9.4 SVC routine and the manner in which it is installed. You must set these options as described below to invoke the SAS 9.4 SVC routine correctly.

- `set TKOPT_SVCNO=`

  The default value is 109 for the ESR SVC 109. If using the "user" SVC instead of the ESR SVC, this option should be set to the SVC number that was defined in STEP 2.

- `set TKOPT_SVCR15=`

  This option only applies if the SAS 9.4 SVC was installed as an ESR Type 4 SVC. The default is 4 for compatibility with previous releases of SAS 9.4. It should specify the routing code that was chosen when the SAS 9.4 SVC was installed into your operating system.

(Optional) Installing the SAS SMF Exit

The `SMF` SAS system option controls whether SMF records formatted by SAS 9.4 Foundation are written to the SMF file at the termination of every SAS Software step. If you intend to enable the SMF option in order to write SMF records, and if you would like to tailor the SMF records that SAS 9.4 Foundation writes, you must install the SMF exit and set the `SMFEXIT=` system option.

STEP 1: Decide whether you need to install the SMF exit.

You can use the SMF exit to examine the SMF record that SAS 9.4 Foundation has formatted modify fields within the record, write the record to a user file, and suppress the writing of the record by SAS 9.4 Foundation.

Note: *If SMF records are to be written to the SMF file, the SAS 9.4 Foundation SVC must be installed. See "Installing the SAS 9.4 SVC Routine" on page 3434 for more information.*

The use of the SMF exit is entirely optional. Even if it is not installed, records are written to the SMF file if the `SMF` and `STIMER` options are in effect and the SAS 9.4 Foundation SVC is installed. If you install the SMF exit, you can still use the `SMFEXIT=` option to specify whether it is to be invoked. If you do not specify the `SMFEXIT=` option, the exit is not invoked. If the value specified is the name of a load module in the search path, the exit is invoked.

STEP 2: Tailor the SMF exit source to meet the requirements of your site.

The sample SMF exit is an assembler source module that must be modified to suit your needs. The source for the sample SMF exit is in the `BAMISC` library member `SMFEXIT`.

Note: *In the z/OS environment, the exit is entered in `AMODE 31`. If you are writing to a user file using an access method that requires you to be in `AMODE 24`, change `AMODE` for the access method calls, then return to `AMODE 31` before returning from the exit.*

The exit is called at SAS 9.4 Foundation initialization, at SAS Software step termination, and at SAS 9.4 Foundation termination. At entry to the exit, `R15` contains the entry point address, `R14` contains the return address, `R13` points to a standard register save area, and `R1` contains the address of a fullword. If the fullword is `0`, the call is being made after SAS 9.4 Foundation is initialized so that the exit can perform any initialization necessary. If your exit is writing records to a user file, you probably want to open the file on this call. If the fullword is `-1`, the call is being made before SAS 9.4 Foundation is terminated so that the exit can perform any termination necessary. If your exit is writing records to a user

file, you probably want to close the file on this call. If the fullword contains neither `0` nor `-1`, it is assumed to be the address of the SMF record to be written.

If the exit returns `0` in `R15`, SAS 9.4 Foundation writes the SMF record pointed to by `R1`. If the exit returns a non-zero value in `R15`, SAS 9.4 Foundation suppresses the writing of the SMF record. The following approaches might be taken in the exit:

- Zero R15 and return immediately to write the SMF record, as is.
- Return immediately leaving a non-zero value in `R15` to suppress the writing of the record.
- Modify the record pointed to by `R1`, as desired, including adding data in the user area provided, or perhaps changing the record type. Note, however, that record type must be greater than `127`. If it is not, the SAS 9.4 Foundation SVC does not write the record to the SMF file. Note also that the record length in the standard header on input does not include the user area. If data is added in this area, the length field must be incrementally lengthened by the number of bytes added. Return a `0` in `R15` to cause SAS 9.4 Foundation to write the modified SMF record.
- Modify the record pointed to by `R1`, as desired, and write the record to a user file. In this case, record-type checking is up to you. Return a non-zero value in `R15` to cause SAS 9.4 Foundation to suppress writing of the SMF record. The format of the record formatted by SAS 9.4 Foundation is as follows:

Hex Offset	SMFREC	DSECT		
00	SMFRLEN	DS	BL2	Record length
02	SMFSEG	DS	BL2	Segment descriptor
04	SMFFLG	DS	BL1	Header flag
05	SMFRTYP	DS	BL1	Record type
06	SMFTIME	DS	BL4	Time given to smf
0A	SMFDATE	DS	PL4	Date given to smf
0E	SMFSID	DS	CL4	System id
12	SMFJOBN	DS	CL8	Jobname
1A	SMFRTME	DS	BL4	Reader time
1E	SMFRDTE	DS	PL4	Reader date
22	SMFSTEP	DS	XL1	Step number
23	SMFRSVD	DS	XL1	Reserved
24	SMFPROC	DS	CL8	Proc name
2C	SMFCPU	DS	F	Proc CPU time in timer units
30	SMFEXCP	DS	F	Proc excp count
34	SMFCORE	DS	F	Proc storage used
38	SMFVUSE	DS	F	Vector usage in .01 sec
3C	SMFVAFF	DS	F	Vector affinity time in .01 sec
40	SMFHSP	DS	F	RSM hiperspace time in .01 sec
44	SMFUSER	DS	XL64	User space (Not included in SMFLEN)

STEP 3: Assemble and link the tailored source.

The JCL required to assemble and link the SMF exit is located in member `BASMF` in the `&PREFIX.INSTALL.CNTL` data set. You can tailor the link step so that the name supplied on the `ENTRY` statement is the entry point that you want to use when invoking the exit. `SMFEXIT1`, which is the name on the `ENTRY` statement in the JCL, is the entry point in the sample SMF exit that simply zeroes `R15` and returns, causing the SMF record to be written to the SMF file as formatted by SAS 9.4 Foundation.

STEP 4: If your site requirements dictate that SMF always be ON, and that the SMFEXIT= always be set, ensure that SMF, SMFTYPE=, SMFEXIT=, STIMER, SVC0SVC=, and SVC0R15= are set in the Restricted Options Table.

See "Customizing Default Options and System Configuration Files" on page 9 for more information.

Required UNIX File System Components

Starting with SAS 9.2, a SASHOME directory is created as part of the installation process which contains key portions of the SAS installation. Beginning with SAS 9.4, help documentation and products that include additional optional functionality provided through Java require components to be installed into the UNIX file system.

The UNIX file system components are automatically unloaded during the installation process. In addition, the SAS deployment registry is created and maintained in the UNIX file system. The SAS deployment registry catalogs all the SAS software deployed at your site.

Configuring SAS Software for Use with the Java Platform

In SAS 9.4 Foundation, products that provide optional Java functionality include the following:

- Base SAS
- SAS/Access Interface to R/3
- SAS Enterprise Miner
- SAS/GRAPH support for the `javaimg` device
- SAS IT Resource Management
- SAS/OR
- SAS/SECURE

A small number of features require the use of IBM's Java SDK. If you specified a location for an installed Java SDK at the time of installation, SAS was configured to use it.

Starting with SAS 9.4M9, the IBM Java SDK is required. For more information, see the SAS 9.4 Support for Java web page.

With previous maintenance releases, SAS will run without the SDK, but you will not be able to take advantage of the features that require an SDK. If you decide at a later time that you want to use the features, you can install an appropriate SDK and configure SAS as described in the steps below.

Note: *JAVA, JAVAIMG, ACTIVEX, and JAVAMETA all require the SDK (JRE on most systems), although they are* ***not*** *new to SAS 9.4. Usage of these device drivers is optional, but the JAVAIMG device driver is used by default for graphs created by stored processes on non-Windows systems. If your configuration is part of a BI installation, configure for Java, or tell all creators of stored processes to specify a different default device driver, such as PNG or GIF.*

SAS/GRAPH support for the JAVAIMG device feature in SAS 9.4 Foundation requires an SDK.

Complete the following steps to configure SAS for use with the Java Platform **ONLY if you did NOT specify the JRE location at installation time:**

STEP 1: Determine the version or release of the SDK that you require.

Visit the SAS 9.4 Support for Java page to determine the minimum version of the SDK required by SAS. This page contains up-to-date information about SAS Java requirements.

If you already have an SDK installed on your system, identify the version or release of the SDK and verify that it meets the requirements shown on the web page.

STEP 2: Install and configure the SDK.

If you have not installed a version or release of an SDK that is specified on the SAS 9.4 Support for Java page, install and configure an SDK now. Click the appropriate **Download** button on the "Third Party Software Downloads" page and follow the instructions on the IBM download site.

For more information about IBM's implementation of Java, visit `http://www-1.ibm.com/servers/eserver/zseries/software/java`, and click **Links**.

STEP 3: Set TKMVSENV options.

Add the following additional option to the member TKMVSENV in your `&prefix.TKMVSENV` data set, where `&prefix` is the High Level-Qualifier of your SAS 9.4 Foundation installation libraries. The syntax depends on your maintenance release of SAS 9.4 because of changes to Java support:

SAS Maintenance Release	Command
SAS 9.4M0 – 9.4M8	`SET TKJNI_OPT_LIBPATH=/java/bin/j9vm:/java/bin`
SAS 9.4M9	`SET TKJNI_OPT_LIBPATH=/java/lib/j9vm:/java/lib`

Where "*java*" is the path to your installed IBM Java SDK.

STEP 4: Set the Region Size.

The region size of a typical SAS job needs to be increased when features are used that incorporate Java. The increase will vary depending upon the application. At a minimum, regions for SAS jobs using Java require 512 MB. For a batch job, add either `REGION=512M` or `REGION=524288K` to the JOB card. For a TSO session, specify `SIZE(524288)`.

Locating Hot Fixes

Occasionally, SAS might find it necessary to supply hot fixes in order to quickly correct reported problems in a SAS installation. A hot fix, along with installation instructions and an audit file, is used to repair a specific reported problem, or problems, built on specific host(s), tested for accuracy, and immediately delivered from the SAS Support website at `http://ftp.sas.com/techsup/download/hotfix/hotfix.html`. An audit file enables SAS Technical Support consultants to determine which hot fixes have been installed. Hot fixes are provided by host, by release, and by product. The tool SASHFADD is available from this site to help you easily identify the hot fixes for your installation.

When hot fixes are released, information regarding the hosts, products, and the problems resolved by the hot fix is sent to you in an email message from SAS.

Chapter 2 – Post-Installation Configuration for National Language Support (NLS)

This chapter contains information on post-installation configuration for Asian and European language support.

Introduction

The following sections describe different methods for configuring your system for locale the options settings you will see in the config and TKMVSENV files, and changes you may need to make if your SAS image uses DBCS support.

Different international locations use variations of the z/OS EBCDIC character set known as encodings to support the language and character set in use, and use various locale-dependent conventions for notation of dates, currency, and more. SAS is distributed in encoding support groupings that usually support a single encoding and a group of related locales. SAS system files containing material translated into local languages are distributed matching these specific encoding support groupings. SAS system installation media VOLSERs, and certain installed SAS system filenames, contain a two-character code that identifies the encoding(s) and locale(s) supported.

z/OS media are available in the following 13 encoded versions listed in the following table, which support multiple locales and regions. The table also shows the locale and region values.

To verify the EBCDIC code page for the media you received, refer to the following table, which maps countries to appropriate encodings.

z/OS SAS Media ID	Default Supported Encodings	Alternate Encodings (*)	Supported Locales
F0	1160	838	Thai_Thailand
C0	870	1153	Bosnian_ BosniaHerzegovina, Croatian_Croatia, Czech_CzechRepublic, Hungarian_Hungary, Polish_Poland, Romanian_Romania, Slovak_Slovakia, Slovenian_Slovenia
	1153	870	Albanian_Albania
R0	1025	1154	Russian_Russia, Bulgarian_Bulgaria, Serbian_Serbia,
	1154	1025	Russian_Ukraine, Macedonian_Macedonia
	875		Greek
W0	1047	924	Afrikaans_SouthAfrica, English_Canada, English_UnitedStates, English_Australia, English_NewZealand, English_SouthAfrica, English_India, English_Philippines, English_HongKong French_Canada, Faroese_FaroeIslands Spanish_Argentina; Spanish_Bolivia; Spanish_Chile; Spanish_Colombia; Spanish_Ecuador; Spanish_ElSalvador; Spanish_Mexico; Spanish_Nicaragua; Spanish_Panama; Spanish_Paraguay; Spanish_Peru; Spanish_PuertoRico; Spanish_UnitedStates; Spanish_Uruguay; Spanish_Venezuela
	933		Korean_Korea

z/OS SAS Media ID	Default Supported Encodings	Alternate Encodings (*)	Supported Locales
	935		Chinese_China, Chinese_Singapore
	937		Chinese_Taiwan, Chinese_Macau, Chinese_HongKong
	939	939E	Japanese_Japan
W2	1140	037	Dutch_Netherlands, Portuguese_Portugal ***Note:*** *W2 can also be used with en_US*
		424	Hebrew_Israel
		425	Arabic_*
	1156	1112	Lithuanian_Lithuania, Latvian_Latvia
W3	1141	273	German_Austria, German_Germany
W5	1142	277	Danish_Denmark, Norwegian_Norway
W6	1143	278	Finnish_Finland, Swedish_Sweden
	1157	1122	Estonian_Estonia
W7	1144	280	Italian_Italy
W8	1145	284	Spanish_Spain
W9	1146	285	English_UnitedKingdom
WA	1147	297	French_France
WB	1148	500	Dutch_Belgium, French_Belgium, French_Switzerland, German_Switzerland, Italian_Switzerland, International
	1164	1130	Vietnamese_Vietnam
WU	930		Japan_Japanese

Table 1 – Mapping of Encodings by Country

Languages and Installation Codes

Important: *The following table is a common reference of possible values for the "`llxx`" values used in the names of certain installed data sets, and referenced in these instructions, PROCS and CLISTS.*

	Required Media & Data Set Codes	
Language	**Language (`ll`) value**	**Encoding (`xx`) value**
English	EN	W0
		All SBCS encoded media (*)
	DB (**)	W0
		WU
French	FR	WB
		W0
		WA

	Required Media & Data Set Codes	
Language	**Language (ll) value**	**Encoding (xx) value**
German	DE	W3
		WB
Spanish	ES	W0
		W8

Table 2 - Required Media & Data Set Codes

(*) English is provided with every SBCS encoded image.

(**) DB does not represent a language. This is the 2-letter code for the DBCS encoded media.

Selecting LOCALE During SAS Foundation Deployment

SAS 9.4 allows the installer to select the locale to use for SAS software being installed on your system. The selected locale is set as the value of the SAS LOCALE system option in the configuration file used to initialize the SAS system.

Beginning in 9.4, the ENCODING system option is set explicitly in the configuration file for all images *except* the DBW0 image.

When LOCALE is set and the ENCODING system option is not explicitly set in the configuration file, it will be implicitly set to an encoding that supports the language for the locale. SAS expects user data to be in the encoding that matches the ENCODING option.

The path encoding, set in the TKMVSENV file, must also match the explicit or implicit ENCODING option, or must be set to the "compiler" encoding, open_ed-1047. The locale and encodings are specified in the TKMVSENV file that sets the environment variables for the SAS environment.

For example, if you have selected German_Germany from the locales selector for a W3 installation, the configuration file and TKMVSENV file will be updated accordingly to your selection:

```
CONFIG(ENW3):
      ENCODING=open_ed-1141
LOCALE=de_DE  TKMVSENV(TKMVENW3):
      set TKOPT_ENV_ENCODING=open_ed-1141
      set TKOPT_ENV_ENCODING_PATH=open_ed-1141
      set TKOPT_ENV_LOCALE=de_DE
```

Or, if you install the W0 image and selected Chinese_China from the locales selector, the DBW0 configuration file and TKMVSENV file will be updated with the following:

```
CONFIG(DBW0):
      DBCS
      LOCALE=zh_CN
  TKMVSENV(TKMVDBW0):
      set TKOPT_ENV_ENCODING=open_ed-935
      set TKOPT_ENV_ENCODING_PATH=open_ed-935
      set TKOPT_ENV_LOCALE=zh_CN
```

For information about the environment variables used by the TK subsystems to specify locales, refer to the table of TKMVSENV name/value pairs supported by the TK interface in "TK Options" on page 66.

Verifying the Settings for the NLS Options

A quick way to verify your installation regarding the NLS setting is to submit the following SAS program:

```
Proc options group=languagecontrol value ; run ;
```

The result is displayed in the log and should indicate the value that has been set for various NLS options:

Option Value Information For SAS Option ENCODING
Value: OPEN_ED-1141
Scope: SAS Session
How option value set: Locale

Option Value Information For SAS Option LOCALE
Value: GERMAN_GERMANY
Scope: SAS Session
How option value set: SAS Session Startup Command Line

Option Value Information For SAS Option NLSCOMPATMODE
Value: NONLSCOMPATMODE
Scope: Default
How option value set: Shipped Default

NOTE: The PROCEDURE OPTIONS used 0.00 CPU seconds and 28538K.

Running SAS in a Different Encoding

If you prefer an encoding other than the most common encoding for the locale, you can also explicitly set the ENCODING system option in the configuration files (CONFIG and TKMVSENV).

Note: *The value for TKOPT_ENV_ENCODING_PATH environment variable must be identical to the value set for the SAS session or it can be set to OPEN_ED-1047.*

On z/OS, the **encoding used by SAS must be compatible with the encoding of the installed media**. If an incompatible encoding is set, the following warning is displayed in the SAS log:

WARNING: *There is an incompatibility between session encoding and the SASHELP encoding. When such a mismatch occurs, some features might not behave as expected. For additional information, contact your Site Administrator.*

The encoding set by the ENCODING system option will also be used by applications that create output in or that establish communications with applications whose syntax and protocols are not determined by SAS. For example, when ODS generates HTML, RTF, or JavaScript, the output will use, by default, the encoding set by the ENCODING system option. If you want your output to be created using a different encoding, refer to the documentation for the Output Delivery System (ODS).

When the ENCODING option is set, the TRANTAB option will default to match the ENCODING system option. The transport format trantabs (translation tables), set by the TRANTAB option, are used for transferring external files. You can override the default by setting TRANTAB at startup or during the SAS

session, though only experienced users should alter settings or do this under the direction of SAS Technical Support.

Running SAS in a Different Locale

When you install SAS 9.4 Foundation on z/OS for a given encoding, the SAS deployment tools select a default locale for the encoding installed. The SAS Deployment wizard provides a list of alternate locales. The installation-generated configuration files set the LOCALE system option to the value selected during the deployment.

A given SAS installation supports one encoding, and sometimes two encodings, along with a specific list of locales that are compatible with those encodings. **Specification of any locale or encoding that does not appear in the installed configuration file is unsupported, and it will produce unpredictable results.**

If you want to change the default locale setting for SAS, you can change the LOCALE system option to the appropriate value in your system configuration files (CONFIG and TKMVSENV). Your configuration file lists all of the locales supported for the default encoded media.

Be aware that changing the locale may change the SAS session encoding if the ENCODING System option is not set explicitly. You may need to set the ENCODING System option in the configuration files (CONFIG and TKMVSENV) that is appropriate for your encoded media.

Running SAS with Localizations

Localizations are available on z/OS for some languages. To run a localized SAS image, select the CLIST with the 2-letter name of the language and 2-letter name of the encoding. For example, if you install the media with the encoding W3 for German, use the CLIST CLSTDEW3. The default TKMVSENV file in the &prefix.TKMVSENV data set has the TKOPT_ENV_LOCALE option set to match the default locale. TKOPT_ENV_ENCODING and TKOPT_ENV_ENCODING_PATH are set by default to match the session encoding. Note that the TKOPT_ENV_ENCODING_PATH value must be set to the SAS session encoding or open_edition-1047. Otherwise, SAS will fail to start. See "Languages and Installation Codes" on page 4343 for details about the CLIST names.

Some encodings support one or more additional localizations. For example, the W0 encoding defaults to English, but also supports the French and Spanish localizations. CLSTFRW0 and CLSTESW0 reside in the &prefix.CLIST data set. If you want to invoke SAS with French localizations every time, use the CLSTFRW0 CLIST.

TKMVSENV files will reside in the &prefix.TKMVSENV dataset. For example, Spanish(ES) and French(FR) belong to the W0 encoding and TKMVESW0 and TKMVFRW0 files are present. If you want to invoke the French localization every time SAS executes, you will need to replace the current TKMVSENV with the content of TKMVFRW0. If you want to replace the shipped version of TKMVSENV, SAS recommends renaming that file rather than deleting it. Remember if you change the TKOPT_ENV_LOCALE value in TKMVSENV, you must also set the LOCALE option in the CONFIG file to the same locale.

You can run a SAS image with English text for any encoded media you install. To run with English text, use the CLIST with 'EN' in its name.

Using NONLSCOMPATMODE

Overview

The default for the NLSCOMPATMODE option is NONLSCOMPATMODE. The NONLSCOMPATMODE option specifies that data is processed in the session encoding, including reading and writing external files as well as processing SAS syntax and user data. The session encoding is the encoding set in the ENCODING= system option.

An additional change you will find with NONLSCOMPATMODE is that the appropriate Open Edition version of the EBCDIC encoding will be used as the session encoding, which uses the new-line character as the end-of-line character. NLSCOMPATMODE used the traditional EBCDIC encodings.

For more information about the NLSCOMPATMODE and NONLSCOMPATMODE system options, refer to the system option documentation for your operating environment or to the Technical Support document TS-758: "*Ensuring Compatibility of Encoding across Different Releases of the SAS® System in the z/OS Environment*".

Obsolete NLCOMPATMODE Option

The NLS Compatibility mode specified with the NLSCOMPATMODE has been deprecated. When the SAS machine is started in NLSCOMPATMODE mode, a warning is displayed.

Base SAS provides:

- separately encoded versions for various locales
- the NLSCOMPATMODE option for legacy code

The NLSCOMPATMODE option enabled mainframe programmers to use replacement characters for variant characters that are part of the SAS language syntax. For example, programmers in the United Kingdom can use the £ character as a replacement for the $ character.

For example, the NLSCOMPATMODE allowed the replacement characters in the following code to run successfully:

```
data mylib.employee;
   input name £char8. +2 income comma6.;
   datalines;
Peterson  21,000
Morgan    17,132
;
run;
```

Replacement characters were effective because compiler encoding was used for all data processing and the hex value for the characters $ and £ are identical (0x5b), whereas the display depends on the terminal emulator settings.

Use of NLSCOMPATMODE perpetuated uncertainties about character handling and caused issues with exchanging data with other software.

Running Legacy Code with NONLSCOMPATMODE

If you run SAS Business Intelligence solutions and any other new products and features under z/OS, you must use NONLSCOMPATMODE.

The NONLSCOMPATMODE option does not use compiler encoding. For example, compiler encoding recognizes £ as a character format. Therefore, legacy code will not run successfully.

To enable legacy code with compiler coding to run, you can do either of the following:

- Replace the £ character (0x5b) with a $ character from session encoding. In this case, 0x4a from code page 1146 because $ is the only recognized format identifier.
- Use a new set of translation tables that override the manner in which SAS interprets characters used in syntax.

Using Translation Tables

The new translation tables map variant characters to session encoding so that you can run your legacy programs with the default NONLSCOMPATMODE setting.

The translation tables are named by the 4-byte version of the SAS® encoding name plus the addition of "_SNP" appended to the end of the name.

To set up your SAS environment to map compiler characters to session encoding, assign the name of the translation table to the 6th slot in the TRANTAB option.

For example, the encoding for English (United Kingdom) is open_ed-1146 and the 4-byte name is eo46. The scanner table is set to eo46_snp.

You can specify the TRANTAB= option for English (United Kingdom) in a configuration file or at SAS invocation:

```
TRANTAB=(eo46wlt1,wlt1eo46,eo46_ucs,eo46_lcs,eo46_ccl, eo46_snp,,eo46_scc).
```

For information about the 4-byte SAS encoding names, refer to "SBCS, DBCS, and Unicode Encoding Values for Transcoding Data" in the *SAS 9.4: National Language Support Reference Guide*.

Note: The 4-byte names for the open edition encodings begin with 'eo'.

Exception for Physical Operating-System Data Set Services Defined in Code Points

Interfaces to z/OS data set services are defined in terms of code points, not characters. The interfaces perform no transcoding

Therefore, variant characters that appear in physical operating-system data set names must be specified in compiler encoding. This exception applies to data set names, volume serial number names, or system managed ssstorage data, storage, and management class names.

.For example, for the 8th character in the data set name userid.$.myfile, a British customer will specify the character £--which is at code point 0x5B in the English-UK 1146 code page--to reference the data set userid.$.myfile.

SAS IT Recourse Management and MXG Code in Compiler Encoding

The SAS® IT Resource Management solution uses MXG code that is only delivered in compiler encoding (The encoding is in code page 1047). You can use the SASAUTOS= option to enable concatenated autocall libraries with different encodings. You must also specify statements to define the encodings. For example:

```
filename sourcelib 'sasprod.mxg.autocall.cass'
         encoding='open_ed-1047';
filename concat 'sastjv.autocall.sas'
         encoding'open_ed-1143';
options sasautos=(concat, sourclib, sasautos);
```

DBCS Support

This section provides information about Asian font catalogs and additional steps to complete the WU install.

Asian Font Catalogs

Commonly used DBCS fonts reside in the font catalog. (However, the configuration file for DBCS extensions does **NOT** contain font definitions.) To use Traditional Chinese fonts, you must specify them in your SAS session.

Specifying the Font Catalog in a SAS Session for Traditional Chinese Fonts

To specify the font catalog in a SAS session, submit the following LIBNAME statement:

libname gfont*x* *'high-level-qualifier*.zt.*font*.GFONT';

In this statement:

- *x* represents a value from 0-9
- *high-level-qualifier* is a user-supplied value
- *font* is the name of the font

The UPPERCASE (WU) image Installation

The WU encoded image is mainly for Japanese customers who wish to use Japanese Katakana characters. This encoded image uses encoding IBM-930 which support Katakana characters. However, it does not support English lowercase characters. The locale selection for the WU image is limited to ja_JP.

In order to complete a WU image installation, add the following options in your CONFIG file:

```
NOGSTYLE, MSGCASE, and VALIDVARNAME=UPCASE
```

Nogstyle option disables the use of ODS styles in the generation of graphs. MSGCASE specifies that the text of SAS messages is displayed in uppercase characters. VALIDVARNAME=UPCASE indicates that a variable name will be mapped to a valid SAS name as described in VALIDVARNAME=V7 except that variable names are in uppercase.

Chapter 3 – Post-Installation Configuration for SAS/ACCESS

SAS/ACCESS Interface to ADABAS

Important: *Starting with SAS 9.4M8, SAS/ACCESS Interface to Adabas is not available. If you upgrade or migrate SAS Foundation to SAS 9.4M8 or later, SAS recommends that you install and use a different data source and a different SAS/ACCESS engine instead. Contact your SAS representative for assistance.*

If you have an existing installation of SAS/ACCESS Interface to Adabas in your environment, SAS recommends that you first unconfigure and uninstall before upgrading or migrating to SAS 9.4M8 or later. A best practice is to unconfigure retired SAS products before you upgrade and to uninstall them after you upgrade. For more information, see "Unconfiguring and Uninstalling Retired Products" in the SAS Guide to Software Updates and Product Changes.

Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

If the ADABAS interface is used extensively at your site, and if your ADABAS system load library is not a link list library, you might want to modify the SAS CLIST and cataloged procedure to allocate the necessary files by default. As an alternative, users must specify the `SASLOAD` operand shown below when invoking the CLIST and must override `STEPLIB` when running batch to concatenate the library that contains your site's Software AG load modules. In either case, they might also be required to allocate `DDCARD`, as appropriate.

Note: *The concatenation order of the SAS LOAD LIBRARY and the load library for database access can be interchanged. Generally, if more database access activities will occur in the SAS session or batch job, place the database load library first in the concatenation (or ahead of the SAS LOAD LIBRARY). The opposite is true when SAS processing dominates the session or JOB.*

- Make the following changes to the CLIST:
  - Replace the following line in the installation-supplied CLIST:
    ```
    SASLOAD('''&prefix.LIBRARY''')
    ```
    with the line
    ```
    SASLOAD('''your.ADABAS.loadlib'' ''&prefix.LIBRARY''')
    ```
  - Optionally, add an appropriate `ALLOCATE` statement for the fileref `DDCARD`, unless appropriate parameters are included in the ADARUN module.
- Make the following changes for the REXX exec:
  - If you need to concatenate the ADABAS load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
    ```
    -DBMSLIBS 'your.ADABAS.loadlib'
    ```
  - If you want the ADABAS load library concatenated before, rather than after, the SAS load library, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
    ```
    -DBMSCONCAT FIRST
    ```
- Make the following changes to the cataloged procedure:
  - Modify the `STEPLIB DD` statement to concatenate your Software AG load library as follows:

```
//STEPLIB   DD DISP=SHR,DSN=&LOAD
//          DD DISP=SHR,DSN=&prefix.LIBRARY
//          DD DISP=SHR,DSN=your.ADABAS.loadlib
```

- (Optional) Add an appropriate `DDCARD DD` statement, unless appropriate parameters are included in the `ADARUN` module. The following example illustrates the `DDCARD` parameters required by this interface:

```
ADARUN DATABASE=001  /*site-specific value*/
ADARUN DEVICE=3380   /*site-specific value*/
ADARUN MODE=MULTI          /*multi (default) or single*/
ADARUN SVC=253       /*site-specific value*/
ADARUN PROGRAM=USER  /*required*/
```

(Optional) Using the NATURAL Security Interface

The modules `NSCDDM` and `NSCDDM22` unloaded to your SAS LIBRARY data set are the NATURAL Security Interface modules supplied by Software AG. If you do not have the NATURAL Security Software, or do not want to use it, delete or rename the load module `NSCDDM`.

The module `NSCDDM` is necessary for support of NATURAL Release 2.3.1 or later or ADABAS Version 6 when using the NATURAL Security Interface. If you are using NATURAL Release 2.2 and you want to use the NATURAL Security Interface, delete or rename the existing `NSCDDM` module, and then rename the module `NSCDDM22` to `NSCDDM`.

(Optional) Using the Samples

Three samples for the ADABAS interface are unloaded into your `&prefix.SAMPLE` library at install time:

- `ADBDOC`, which is used to create sample data sets. These sample data sets are referenced in *SAS/ACCESS Interface to ADABAS Software, Version 8 and later*.
- `ADBUTL` contains input to the ADABAS Utilities that are used to create the four ADABAS files referenced in *SAS/ACCESS Interface to ADABAS Software, Version 8 and later*. You might want to create NATURAL DDMs to match these files using the NATURAL software.
- `ADBSTMT` contains line-mode statements used to create sample access and view descriptors. These access and view descriptors are referenced in *SAS/ACCESS Interface to ADABAS Software, Version 8 and later*.

(Optional) SAS Options for this Interface

To see a list of SAS options for this interface, invoke SAS and submit the following statements:

```
proc options group = adabas; run;
```

Appendix 1 in the *SAS/ACCESS Interface to ADABAS Software* manual also lists SAS options for this interface. You might want to review these during installation. All options have defaults.

The options applicable to this interface are all invocation options. That means you can change them when you invoke SAS but not during a SAS session. Some system options can be overridden by using the corresponding data set options. You can supply data set options when a data set is referenced in a SAS procedure or DATA step.

You are allowed to restrict changes to invocation options by placing them in the Restricted Options Table, `SASOPTRS`. This process is described in "Customizing Default Options and System Configuration Files" on page 99.

One option, ADBUPD=, is suggested as a restricted option. The ADBUPD= option determines whether the SAS/ACCESS Interface to ADABAS engine can perform both read and update operations or is restricted to read-only. The default option value, Y, specifies that both reads and updates are allowed. The option value, N, specifies that the engine can only read ADABAS data; any attempt to update an ADABAS file results in an error.

Note: *The ABDUPD= option replaces the ADBENGMD= option available in earlier releases.*

Other options you might want to examine first are those listed in Appendix 1 of the *SAS/ACCESS Interface to ADABAS Software, Version 8 and later*, under the heading ADABAS System Options.

NATURAL Date and Time Support

The SAS/ACCESS Interface to ADABAS now supports NATURAL Date (D) and Time (T) datatypes. Such fields will have their data values translated into the equivalent SAS dates and times respectively.

When an access descriptor is created on a NATURAL or PREDICT DDM containing a date field, the default SAS format and informat is set to DATE9. As with other default formats and informats displayed with the LIST commands, this can be changed to another compatible SAS format/informat.

A NATURAL time field has a default SAS format and informat of TIME8. This can be changed to another SAS format/informat.

If the NATURAL time field is an extended time field, this can be interpreted as a SAS datetime value by specifying an E in the DB Content field. In this case, the default SAS format and informat is changed from TIME8. to DATETIME18. This can be changed to another compatible SAS format/informat.

While it is not possible to display these values as simple numeric values (date or time values), NATURAL date and time values are stored differently than SAS date and time values. Thus the values displayed in this manner will not be the same as if they had been displayed through a NATURAL application. It is not recommended that these values be used as other than their corresponding date or time values.

Reentrancy

Load module SASIOADB is non-reentrant because of the non-reentrant program ADAUSER provided by Software AG, which must be linked with SASIOADB in order to communicate with the ADABAS DBMS.

SAS/ACCESS Interface to CA-DATACOM/DB

Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

If the CA-DATACOM/DB interface is used extensively at your site, and your DATACOM system load library is not a link list library, you might want to modify the SAS CLIST and cataloged procedure to allocate the necessary files by default. If you do not, users will be required to specify the SASLOAD operand shown below when invoking the CLIST, and to override STEPLIB when running batch to concatenate the library that contains your site's CA-provided load modules.

Note: *The concatenation order of the* SAS LOAD LIBRARY *and the load library for database access can be interchanged. Generally, if more database access activities will occur in the SAS session or batch job, place the database load library first in the concatenation (or ahead of the* SAS LOAD LIBRARY*). The opposite is true when SAS processing dominates the session or JOB.*

- Make the following change to the CLIST:
  Replace the following line in the installation-supplied CLIST:

```
SASLOAD('''&prefix.LIBRARY''')
```

with the line:

```
SASLOAD('''your.DATACOM.loadlib'' ''&prefix.LIBRARY''')
```

- Make the following changes for the REXX exec:
  - If you need to concatenate the DATACOM load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
    ```
    -DBMSLIBS 'your.DATACOM.loadlib'
    ```
  - If you want the `DATACOM` load library concatenated before, rather than after, the SAS load library, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
    ```
    -DBMSCONCAT FIRST
    ```
- Make the following change to the cataloged procedure:
  Modify the `STEPLIB DD` statement to concatenate your CA-provided load module library as follows:

```
//STEPLIB DD DISP=SHR,DSN=&LOAD
//              DD DISP=SHR,DSN=&prefix.LIBRARY

//              DD DISP=SHR,DSN=your.DATACOM.loadlib
```

(Optional) Using the Samples

Three samples for the CA-DATACOM/DB interface are unloaded into your `&prefix.SAMPLE` library at install time:

- `DDBDOC` is used to create sample data sets referred to in SAS/ACCESS Interface to CA-DATACOM/DB: Reference, Version 8 and later, First Edition.
- `DDBUTL` contains input to the CA-DATADICTIONARY Batch Utility that creates the database referred to in *SAS/ACCESS Interface to CA-DATACOM/DB: Reference, Version 8 and later, First Edition*. `DDBUTL` creates a new CA-DATACOM/DB database with four tables.
- `DDBDESC` contains code to create the access and view descriptors to be used in recreating the examples in *SAS/ACCESS Interface to CA-DATACOM/DB: Reference, Version 8 and later, First Edition*.

(Optional) SAS Options for this Interface

To see a list of SAS options for this interface, invoke V9 SAS and submit the following statements:

```
proc options group=datacom; run;
```

You can also refer to Appendix 1 in *SAS/ACCESS Interface to CA-DATACOM/DB: Reference, Version 8 and later, First Edition* for a list of SAS options for this interface. You might want to review this information during installation. All options have defaults.

The options applicable to this interface are invocation options. That means you can change them when you invoke SAS, but not during a SAS session. Some system options have corresponding data set options; these can be set during a SAS procedure or DATA Step. Refer to Appendix 2 in *SAS/ACCESS Interface to CA-DATACOM/DB: Reference, Version 8 and later, First Edition* for information on these data set options.

One invocation option, DDBMISS=, specifies a value to be used for representing null values when the SAS/ACCESS Interface engine for CA-DATACOM/DB inserts or updates records in a database table. Valid values for the DDBMISS= option are blank (0X40) which is the default, and 0X00.

You are allowed to restrict changes to invocation options by placing them in the Restricted Options Table, SASOPTRS. This process is described in "Customizing Default Options and System Configuration Files" on page 99.

One option, DDBUPD=, is suggested as a restricted option. The DDBUPD= system option determines whether the SAS/ACCESS Interface for CA-DATACOM/DB engine can perform both read and update operations, or is restricted to Read-Only. The default option value, Y, specifies that both reads and updates are allowed. The option value, N, specifies that the engine can only read CA-DATACOM/DB tables; any attempt to update a CA-DATACOM/DB table results in an error.

SAS/ACCESS Interface to CA-IDMS

Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

Note: *This task is required.*

If the CA-IDMS interface is used extensively at your site, and your IDMS system load library is not a link list library, you might want to modify the SAS CLIST and cataloged procedure to allocate the necessary files by default. Refer to Steps 1, 2 and 3 below for instructions on making these changes.

If you do not make the changes described in Steps 1, 2 and 3, you are required to specify the LOAD operand shown below when invoking the CLIST and/or the LOAD= parameter when executing the cataloged procedure to concatenate the library that contains your site's CA-IDMS load modules.

If you are accessing the IDMS databases using central version, you might also be required to allocate the SYSCTL file. If you are accessing the IDMS databases using local mode, you might also be required to allocate the database files.

- Enter the following command under TSO to start a SAS session in which you will execute the IDMS interface:

  ```
  SASname LOAD('''your.IDMS.loadlib''')
  ```

  or

  ```
  SASRXname -DBMSLIB 'your.IDMS.loadlib'
  ```

  Where your.IDMS.loadlib is the IDMS system load library and the SASname is the name of the CLIST or SASRXname is the name of the REXX exec you use to invoke SAS.

- To use the IDMS interface with the SAS cataloged procedure, use the LOAD= parameter of the PROC to specify your site's load library. For example, you can use the following code in your JCL:

  ```
  //SASIDM EXEC SASname,LOAD='your.IDMS.loadlib'
  ```

Note: *The name that the user supplies must consist of all capital letters.*

Step 1: Update your SAS CLIST (optional)

If the IDMS interface is used extensively at your site, you might want to alter the CLIST so that your IDMS load library is allocated and concatenated before the SAS load library in the TASKLIB symbol.

Replace the following line in the installation-supplied CLIST (where SASLOAD is the replacement parameter in the CLIST).

- Replace the following line in the installation-supplied CLIST:

  ```
  SASLOAD('''&prefix.LIBRARY''')
  ```

  with the line:

  ```
  SASLOAD('''your.IDMS.loadlib'' ''&prefix.LIBRARY''')
  ```

- Add an appropriate `ALLOCATE` statement for the fileref `SYSCTL`, if you are accessing your IDMS databases using central version. Add the appropriate `ALLOCATE` statements for the IDMS databases and dictionary you have the authority to access, if you are accessing your IDMS databases using local mode.

Step 2: Update your SAS REXX exec (optional)

- If you need to concatenate the IDMS load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
  `-DBMSLIBS 'your.IDMS.loadlib'`
- If you want the `IDMS` load library concatenated before, rather than after, the SAS load library, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
  `-DBMSCONCAT FIRST`
- In the SASRXCFG(REXXSITE) config file, add an appropriate `ALLOCATE` command for the fileref `SYSCTL` as the value of the -inittsooption, if you are accessing your IDMS databases using central version. Add the appropriate `ALLOCATE` commands for the IDMS databases and dictionary you have the authority to access, if you are accessing your IDMS databases using local mode.

Step 3: Update your SAS cataloged procedure (optional)

- If the IDMS interface is used extensively at your site with the SAS cataloged procedure, you might want to make the `STEPLIB DD` statement change permanent to your cataloged procedure. To always execute SAS 9.4 Foundation using the IDMS interface, modify the STEPLID DD statement in your cataloged procedure to reflect the following lines:

  ```
  //STEPLIB   DD  DISP=SHR,DSN=&LOAD
  //          DD  DISP=SHR,DSN=&prefix.LIBRARY

  //          DD  DISP=SHR,DSN=your.IDMS.loadlib
  ```

  - Add the appropriate DD statements for the fileref SYSCTL, if you are accessing your IDMS databases using central version.
  - Add the appropriate DD statements for the IDMS databases and dictionary you have authority to access, if you are accessing your IDMS databases using local mode.

SAS/ACCESS DATA Step Interface Enhancement

The DATA step component of the SAS/ACCESS Interface to CA-IDMS is the only component available with this release. This component allows you to directly access network data using special SAS system extensions for the standard SAS INFILE statement, along with DATA step programming statements. The

INFILE statement extensions, along with the DATA step programming statements, allow you to generate DML calls to the database.

SAS/ACCESS Interface to DB2

Note: *This task is required.*

See the SAS Hot Fix site to download the Database Request Modules (DBRMs) that are needed for the DB2 bind: http://ftp.sas.com/techsup/download/hotfix/hotfix.html
Under the **Additional Downloads** section on the left-hand side of the page, select SAS 9.4 DBRM members for SAS/ACCESS Interface to DB2 for z/OS.

The Db2 database administrator at your site must perform the following steps.

Even if you installed and implemented any previous release of SAS 9.4 Foundation, and SAS/ACCESS Interface to DB2, and completed the `BIND` and `GRANT` steps at that time, you will need to repeat them for this installation.

STEP 1: Bind the DBRMs into an application plan (required).

You can bind the application plan by using the `BIND` option from the DB2I panels under ISPF. As an alternative, you can issue the `BIND` command from TSO through the DSN command processor running in either foreground or background. A new plan will be created. If a plan with this name already exists on your system, it will be replaced.

- The syntax of the `BIND` command is as follows:

```
BIND PACKAGE (SAS94)
     OWNER          (SAS) -
     CURRENTDATA    (YES) -
     EXPLAIN        (NO) -
     ISOLATION      (CS) -
     QUALIFIER      (SAS) -
     RELEASE        (COMMIT) -
     SQLERROR       (NOPACKAGE) -
     VALIDATE       (RUN) -
     KEEPDYNAMIC    (YES) -
     DEGREE         (ANY) -
     ENABLE         (*) -
     MEMBER         (SAS94) -
     LIBRARY        ('<DBRM library>')

BIND PLAN (SAS94)
     OWNER          (SAS) -
     ACQUIRE        (USE) -
     CACHESIZE      (1024) -
     CURRENTDATA    (YES) -
     EXPLAIN        (NO) -
     ISOLATION      (CS) -
     QUALIFIER      (SAS) -
     RELEASE        (COMMIT) -
     DISCONNECT     (EXPLICIT) -
     SQLRULES       (DB2) -
     VALIDATE       (RUN) -
     KEEPDYNAMIC    (YES) -
     DEGREE         (ANY) -
     RETAIN        -
```

```
     NODEFER        (PREPARE) -
     ENABLE         (*) -
     PKLIST         ( -
                    *.SAS94.* -
                    )
```

Note: *These are the option values that have been run and tested. Other values can cause different behaviors. Consult the Db2 manuals for more information about bind options.*

If you plan to use SAS to access remote servers, you must bind SAS at each remote location as well as the local servers. For a Db2 database on z/OS servers, use the SAS94 DBRM member and the `BIND PACKAGE` command as shown above. For the Db2 on non-z/OS servers, the actual DBRM member used will depend on the version of Db2 that you are accessing.

- If you are accessing DB2 V11 or later, the SAS94 DBRM member should be used.
- When accessing DB2 V10, another DBRM called SAS94X is provided which does not reference any DB2 under z/OS specific statements. Rename the SAS94X member to SAS94 before binding. The syntax of the BIND command is as follows:

```
BIND PACKAGE       (SAMPLE.SAS94) - /* SAMPLE -> remote location */
     MEMBER        (SAS94) -
     ACTION        (REPLACE) -
     ISOLATION     (CS) -
     SQLERROR      (CONTINUE) -
     DEGREE        (ANY) -
     VALIDATE      (BIND) -
     RELEASE       (COMMIT) –
     LIBRARY       ('<DBRM library>')
```

Note: *These are the values that have been tested. Other values can cause different behaviors. Consult the DB2 manuals for more information about bind options.*

STEP 2: Bind the SASTKTS DBRMs into an application plan (required).

You can bind the application plan by using the `BIND` option from the DB2I panels under ISPF. As an alternative, you can issue the `BIND` command from TSO through the DSN command processor running in either foreground or background. A new plan will be created. If a plan with this name already exists on your system, it is replaced.

The syntax of the `BIND` command is as follows:

```
BIND PACKAGE (SASTKTS)
     OWNER          (SAS) -
     CURRENTDATA    (YES) -
     EXPLAIN        (NO) -
     ISOLATION      (CS) -
     QUALIFIER      (SAS) -
     RELEASE        (COMMIT) -
     SQLERROR       (NOPACKAGE) -
     VALIDATE       (RUN) -
     KEEPDYNAMIC    (YES) -
     DEGREE         (ANY) -
     ENABLE         (*) -
     MEMBER         (SASTKTS) -
     LIBRARY        ('<DBRM library>')
```

```
BIND PLAN (SASTKTS)
     OWNER          (SAS) -
     ACQUIRE        (USE) -
     CACHESIZE      (1024) -
     CURRENTDATA    (YES) -
     EXPLAIN        (NO) -
     ISOLATION      (CS) -
     QUALIFIER      (SAS) -
     RELEASE        (COMMIT) -
     DISCONNECT     (EXPLICIT) -
     SQLRULES       (DB2) -
     VALIDATE       (RUN) -
     KEEPDYNAMIC    (YES) -
     DEGREE         (ANY) -
     RETAIN         -
     NODEFER        (PREPARE) -
     ENABLE         (*) -
     PKLIST         ( -
                    *.SASTKTS.* -
                    )
```

Note: *These are the option values that have been run and tested. Other values can cause different behaviors. Consult the DB2 manuals for more information about bind options.*

STEP 3: Grant EXECUTE authority to users (required).

For anyone to be able to run SAS, `EXECUTE` authority should be granted to PUBLIC on the SAS94 and SASTKTS plans. This can be done by issuing the `GRANT` command as follows:

```
GRANT EXECUTE ON PLAN SAS94 TO PUBLIC;

GRANT EXECUTE ON PLAN SASTKTS TO PUBLIC;
```

A similar `GRANT` command must be executed at each of the remote locations where the SAS DBRM was bound in order to access those locations through SAS.

Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

Note: *This task is required.*

If the DB2 interface is used extensively at your site, and if your DB2 system load library is not a link list library, you might want to modify the SAS CLIST, REXX exec, and cataloged procedure to allocate your DB2 system load library by default. See Steps 1, 2 and 3, which follow, for the changes to make.

If you do not make the changes described in Steps 1 and 2, users must specify the `LOAD` operand shown in the following example when invoking the CLIST. The user must also specify the `LOAD=` parameter when executing the cataloged procedure to concatenate your DB2 system load library.

Note: *The concatenation order of the SAS LOAD LIBRARY and the load library for database access can be interchanged. Generally, if more database access activities will occur in the SAS session or batch job, place the database load library first in the concatenation (or ahead of the SAS LOAD LIBRARY). The opposite is true when SAS processing dominates the session or JOB.*

- Enter the following command under TSO to start a SAS session in which you will execute the DB2 interface:

```
SASname LOAD('''your.db2.loadlib''')
```

or

```
SASRXname -DBMSLIB 'your.db2.loadlib'
```

where `your.db2.loadlib` is the DB2 system load library and `SASname` is the name of the CLIST or SASRXname is the name of the REXX exec you use to invoke SAS.

- To use the DB2 interface with the SAS cataloged procedure, use the `LOAD=` parameter of the PROC to specify your site's DB2 load library. For example, you can use the following code in your JCL:

```
//SASDB2     EXEC SASname,LOAD='your.DB2.loadlib'
```

STEP 1: Update your SAS CLIST (optional).

If the DB2 interface is used extensively at your site, you might want to alter the CLIST so that your DB2 load library is allocated and concatenated before the SAS load library in the `TASKLIB` symbol. Replace the following line in the installation-supplied CLIST (where `SASLOAD` is a replacement parameter in the CLIST):

```
SASLOAD('''&prefix.LIBRARY''')+
```

with this line:

```
SASLOAD('''your.db2.loadlib''  ''&prefix.LIBRARY''') +
```

STEP 2: Update your SAS REXX exec (optional).

If you need to concatenate the DB2 load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:

```
-DBMSLIBS 'your.DB2.loadlib'
```

The DB2 load library can be concatenated before, rather than after, the SAS load library. Insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line: `-DBMSCONCAT FIRST`

STEP 3: Update your SAS cataloged procedure (optional).

If the DB2 interface is used extensively at your site with the SAS cataloged procedure, you might want to permanently change the `STEPLIB DD` statement in your cataloged procedure. To always execute SAS 9.4 Foundation using the DB2 interface, modify the `STEPLIB DD` statement in your cataloged procedure to reflect the following lines.

```
//STEPLIB      DD DISP=SHR,DSN=&LOAD
//             DD DISP=SHR,DSN=&prefix.LIBRARY
//             DD DISP=SHR,DSN=your.db2.loadlib
```

where `&prefix.LIBRARY` is SAS load library and `your.db2.loadlib` is the DB2 System load library.

Creating and Loading the Sample Tables (optional)

Refer to the SAS/ACCESS Sample Repository (sas-access-samples) GitHub project for coding examples based on sample IBM Db2 tables that can be created at your site. Creating these tables will assist the users at your site in learning how to use SAS/ACCESS Interface to DB2.

(Optional) SAS Options for this Interface

To see a list of SAS options for this interface, invoke SAS and submit the following statements:

```
proc options group = db2; run;
```

As an alternative, you can refer to online Help for the current information. You might want to review these system options during installation. All options have defaults.

Most of the options applicable to this interface are invocation options. That means you can change them when you invoke SAS, but not during a SAS session. Some system options have corresponding data set options; these can be set during a SAS procedure or DATA step.

Other system options such as `DB2SSID=` can be set at invocation time or within a SAS session. You are allowed to restrict changes to invocation options by placing them in the Restricted Options Table. This process is described in "Customizing Default Options and System Configuration Files" on page 99.

One option, `DB2UPD=`, is suggested as a restricted option. The `DB2UPD=` option determines if, in certain circumstances, the SAS/ACCESS Interface for DB2 engine can perform both read and update operations, or is restricted to read-only. The default option value, `Y`, specifies that both reads and updates are allowed. The option value, `N`, specifies that the engine can only read DB2 data; users who are using access views or libname engine and attempt to update a DB2 file will receive an error message.

Note: *Even with the DB2UPD= set to N, DB2 Tables can be updated by users with PROC SQL Pass-Through Facility, PROC DBLOAD and the Version 5 Compatibility procs.*

Special Consideration for Using the RRS Attachment Facility.

The V9 DB2 Access Product supports both RRSAF and CAF as the attachment facility. For usage details, see the sections "SAS 9.4 Options and Settings for DB2" and "Information for the Database Administrator" in "DB2 under z/OS" of "SAS/ACCESS Software: DBMS-Specific Information" in *SAS/ACCESS 9.4 for Relational Databases: Reference*.

Three installation requirements must be met before RRSAF can be used as the attachment facility:

The SAS 9.4 Foundation SVC Routine must be installed, and must be at Level 9 or greater. See "Installing the SAS 9.4 Foundation SVC Routine" on page 3434 for details on the SAS SVC.

The load module named DSNRLI is part DB2 and should be found in DB2's SDSNLOAD load library. DSNRLI must be in an APF authorized load library that is included in the LINKLIST.

The load module named SASD2RUB is part of the SAS DB2 Access Product and should be found in your SAS load library. SASD2RUB must be in an APF authorized load library, which is recommended to be included in the LINKLIST.

With these requirements met, the full functionality of the RRSAF is supported.

Notes: *If these requirements are not met, RRSAF can still be used by a non-server SAS 9.4 session, but the Authorized Sign-on support will not work. Because Authorized Sign-on support is the main reason for having SAS 9.4, to use this facility, these requirements are not considered optional.*
Starting with SAS 9.4M2, the requirement that SASD2RUB must be in an APF authorized library was removed. As a result, the SAS SVC must be updated to Level 11. (The SVC routine shipped with SAS 9.4 and 9.4M1 is Level 10.)

SAS/ACCESS Interface to IMS-DL/I

Customizing the SAS CLIST and Cataloged Procedure

***Note**: This task is required.*

If the IMS interface is used extensively at your site, you might want to alter the CLIST and cataloged procedure so that your IMS libraries are allocated and concatenated.

Note: *The concatenation order of the SAS LOAD LIBRARY and the load library for database access can be interchanged. Generally, if more database access activities will occur in the SAS session or batch job, place the database load library first in the concatenation (or ahead of the SAS LOAD LIBRARY). The opposite is true when SAS processing dominates the session or JOB.*

STEP 1: Determine the IMS library data set names and identify the IMS libraries that you need to allocate.

In order to access your IMS databases, you need to add allocations for the following IMS data sets to your SAS CLIST and cataloged procedure:

- DFSRESLB
- `DFSVSAMP` (only required for VSAM and OSAM access methods, as defined in the DBD)
- IEFRDER
- IMS
- `DD` statements for your IMS databases if using DLI or DBB region.

Depending on how your site runs IMS, you might also want to add allocations for these additional data sets:

- IMSACB
- IMSERR
- RECON1
- RECON2.

STEP 2: Add allocations to your SAS CLIST, specifying your IMS library data set names as determined in STEP 1.

- Replace the following line in the installation-supplied CLIST (where SASLOAD is a replacement parameter in the CLIST) so that your IMS `RESLIB` is concatenated before the SAS Load library in the `TASKLIB` symbol.
  Replace the following:

  ```
  SASLOAD('''&prefix.LIBRARY''')
  ```

  with the following:

  ```
  SASLOAD('''your.ims.reslib'' ''&prefix.LIBRARY''')
  ```

- Add the following parameter at the top of the default CLIST to allow optional user input of the `IMS LOG IEFRDER` data set name:

```
/* --------Allow IMS LOG DSN input ------------------- */ +
IMSLOG(NULLFILE)    /* IMS LOG DSN              */ +
```

Add lines like the following in the default SAS CLIST to allocate required IMS libraries:

```
ALLOC F(DFSRESLB) DA('your.ims.reslib') SHR REU
ALLOC F(IMS) DA('your.ims.psblib' 'your.ims.dbdlib') SHR REU
IF &STR(IMSLOG) NE THEN ALLOC F(IEFRDER) DA('&IMSLOG.') OLD
ALLOC F(DFSVSAMP) DA('your.parmlib(DFSVSAMP)') SHR REU
ALLOC F(database) DA('your.ims.database') OLD
```

Note: *You only need to allocate your database data sets with disposition OLD if you will be writing to them with the SAS/ACCESS software. Database data set allocations are not required for a region type of BMP.*

- If needed, add lines like the following immediately after the statements you added for required IMS libraries:

```
ALLOC F(IMSACB) DA('your.ims.acblib') SHR REU
ALLOC F(RECON1) DA('your.recon1') SHR REU
ALLOC F(RECON2) DA('your.recon2') SHR REU
ALLOC F(IMSERR) DA('your.dump.data set') SHR REU
```

STEP 3: Add allocations to your SAS cataloged procedure, specifying your IMS library data set names as determined in STEP 1.

Concatenate your `IMS RESLIB` to your `STEPLIB` statement in your default SAS cataloged procedure as follows:

```
//STEPLIB  DD DISP=SHR,DSN=&LOAD
//         DD DISP=SHR,DSN=&prefix.LIBRARY
//         DD DISP=SHR,DSN=your.ims.reslib
```

where `&prefix.LIBRARY` is SAS 9.4 Foundation load library and `your.ims.reslib` is the IMS `RESLIB` library.

Add lines like the following in your default SAS cataloged procedure. If you want to use the IMS LOG facility, be sure to include appropriate `DD` specifications for your site in the `IEFRDER DD` statement.

```
//DFSRESLB DD DISP=SHR,DSN=your.ims.reslib
//IMS      DD DISP=SHR,DSN=your.ims.psblib
//         DD DISP=SHR,DSN=your.ims.dbdlib
//IEFRDER  DD DSN=NULLFILE,DISP=(,KEEP),
//            UNIT=(TAPE,,DEFER),VOLSER=xxxxxx,
//            DCB=(RECFM=VB,BLKSIZE=1920,LRECL=1916,BUFNO=2)
//database DD DISP=OLD,DSN=your.ims.database
```

Note: *You only need to allocate your database data sets with disposition* `OLD` *if you will be writing to them with the SAS/ACCESS software. Database data set allocations are not required for a region type of BMP.*

If needed at your site, also add lines like the following in your default SAS cataloged procedure:

```
//IMSACB DD  DISP=SHR,DSN=your.ims.acblib
//RECON1 DD  DISP=SHR,DSN=your.recon1
//RECON2 DD  DISP=SHR,DSN=your.recon2
//IMSERR DD  DISP=SHR,DSN=your.dump.dataset
```

Verifying Installation of the SAS/ACCESS Interface to IMS

Note: *This task is required.*

STEP 1: Set up JCL to invoke the standard SAS cataloged procedure at your site.

The additional allocations described in the preceding STEP 3 are not required because TESTIMS does not attach to the IMS DBMS.

Before executing the TESTIMS verification job, you must edit either the LIBNAME or CREATE statement contained in that member of your `&prefix.TESTS PDS`. You must either supply a valid permanent SAS data library name in place of 'your.sas.library' in the LIBNAME statement, or you might choose to delete the LIBNAME statement and write the test access descriptor to the WORK library by changing the libref in the CREATE statement from 'mylib' to 'work'.

Include a SYSIN DD statement like the following to run the installation verification program:

```
//SYSIN  DD  DISP=SHR,DSN=&prefix.TESTS(TESTIMS)
```

STEP 2: Submit the job and verify the results.

This job should complete with return code `0`.

When this job completes successfully, you have verified the ability to create descriptors in SAS/ACCESS. Since this job does not attach to the DBMS, it does not test the actual interface.

SAS Options for this Interface

Note: *This task is required.*

To see a list of SAS options for this interface, invoke V9 SAS and submit the following statements:

```
proc options group=ims; run;
```

You can also refer to Appendix 1 in *SAS/ACCESS Interface to IMS-DL/I Software, Version 8 and later, First Edition* for a list of SAS options for this interface. You might want to review the default option settings during installation, as they determine whether the interface can attach successfully to the IMS DBMS.

The `BMPREAD=, DLIREAD=, IMSBPUPD=, IMSDLUPD=, IMSID=, IMSREGTP=, and IMSWHST=` options for this interface are invocation options. That means you can change them when you invoke SAS, but not during a SAS session. All other options are classified as session options.

You are allowed to restrict changes to invocation options by placing them in the Restricted Options Table. This process is described in "Customizing Default Options and System Configuration Files" on page 99.

`BMPREAD=, DLIREAD=, IMSBPUPD=, and IMSDLUPD=` are suggested as restricted options. The `BMPREAD=` and `DLIREAD=` options determine whether the SAS/ACCESS Interface to IMS DATA step can perform both read and update operations, or is restricted to read-only. The default value, `N`, allows update processing. The option value, `Y`, causes SAS to return a status code of `SE` and set `_ERROR_=1` if a DL/I update call is issued. The `IMSBPUPD=` and `IMSDLUPD=` options determine whether the SAS/ACCESS Interface for IMS engine can perform both read and update operations, or is restricted to read-only. The default option value, `Y`, specifies that both reads and updates are allowed. The option value, `N`, specifies that the engine can only read IMS data; any attempt to update an IMS file results in an error. For more information about these options, refer to *SAS/ACCESS Interface to IMS-DL/I Software, Version 8 and later*.

SAS/ACCESS Interface to Oracle

Important: *Starting with SAS 9.4M8, SAS/ACCESS Interface to Oracle is not available. If you upgrade or migrate SAS Foundation to SAS 9.4M8 or later, SAS recommends that you install and use a different data source and a different SAS/ACCESS engine instead. Contact your SAS representative for assistance.*

If you have an existing installation of SAS/ACCESS Interface to Oracle in your environment, SAS recommends that you first unconfigure and uninstall before upgrading or migrating to SAS 9.4M8 or later. A best practice is to unconfigure retired SAS products before you upgrade and to uninstall them after you upgrade. For more information, see "Unconfiguring and Uninstalling Retired Products" in the SAS Guide to Software Updates and Product Changes.

Oracle Multithreading Support

You will need to perform additional setup in order to successfully use SAS/ACCESS Interface to Oracle and to take advantage of multithreading. The additional setup is now needed because part of the product now runs with POSIX(ON).

You must **copy** environment variables from the ORA$ENV DD to the TKMVSENV DD. Since the multithreaded Oracle calls are running with POSIX(ON), Oracle will not read the ORA$ENV DD for these calls. The syntax for the TKMVSENV DD is different, so you must also add 'set' to the front of each line. In addition, to avoid collisions, "TKBOX_OPT_" must be added. This will be stripped out before it is sent to the Oracle program.

Important: *The ORA$ENV DD needs to remain in order to support single-threaded calls. Copy its contents to the TKMVSENV DD to provide support for the additional multithreaded calls.*

For example, if the content of the ORA$ENV DD is:

```
NLS_LANG='AMERICAN_AMERICA.WE8EBCDIC1047'
NLS_DATE_FORMAT='YYYY-MM-DD'
ORACLE_HOME='/usr/lpp/oracle/V10.2.0.3/OraHome'
ORACLE_SID=ORA1
```

The TKMVSENV DD would contain the following:

```
set TKBOX_OPT_NLS_LANG=AMERICAN_AMERICA.WE8EBCDIC1047
set TKBOX_OPT_NLS_DATE_FORMAT=YYYY-MM-DD
set TKBOX_OPT_ORACLE_HOME=/usr/lpp/oracle/V10.2.0.3/OraHome
set TKBOX_OPT_ORACLE_SID=ORA2
```

Here is a list of the other variables that can be set in this manner:

- TKBOX_OPT_TNS_ADMIN
- TKBOX_OPT_TWO_TASK
- TKBOX_OPT_ORA_NLS33
- TKBOX_OPT_NLS_NCHAR

In particular, TKBOX_OPT_TNS_ADMIN needs to be set to point to an HFS location for the `tnsnames.ora` file because the TNSNAMES DD will also not be read by an Oracle POSIX application.

Customizing the SAS CLIST, REXX Exec, and Cataloged Procedure (Required)

If the Oracle interface is used extensively at your site, and if your Oracle system load library is not a link list library, you might want to modify the SAS CLIST, REXX exec, and cataloged procedure to allocate your Oracle system load library by default. See Steps 1 and 2 below for the changes to make.

If you do not make the changes described in Steps 1 and 2, users must specify the `LOAD` operand shown below when invoking the CLIST and the `LOAD=` parameter when executing the cataloged procedure to concatenate your Oracle system load library.

- Enter the following command under TSO to start a SAS session in which you will execute the Oracle interface:
  SASname LOAD(’’’your.ORACLE.cmdload’’’)

  where `your.ORACLE.cmdload` is the Oracle system load library and `SASname` is the name of the CLIST or REXX exec you use to invoke SAS.

  To use the Oracle interface with the SAS cataloged procedure, use the `LOAD=` parameter of the PROC to specify your site’s load library. For example, you can use the following code in your JCL:

  ```
  //SASORA EXEC SASname,LOAD='your.ORACLE.cmdload'
  ```

- If you are running SQL*Net Version 2 or above and you will be accessing the Oracle server as an z/OS client, you might also need to allocate the TNSNAMES file to your current SAS session. Check with your DBA to see if the TNSNAMES file is centrally maintained. If it is not, get the correct data set name for your site’s TNSNAMES file.
  If you do not have the TNSNAMES file allocated, you will receive the following error from Oracle:

  ```
  ORACLE connection error: ORA-12154 TNSNAMES: could not resolve service name.
  ```

- Optionally, you might want to allocate the data set that contains the default pathname for connecting to the Oracle database server. This is especially convenient if you are always connecting to the same server. Check with your DBA for the correct `DDname` to the data set name.

STEP 1: Update your SAS CLIST (Optional)

If the Oracle interface is used extensively at your site, you might want to alter the CLIST so that your Oracle load library is allocated and concatenated before the SAS load library in the `TASKLIB` symbol. Replace the following line in the installation-supplied CLIST (where `SASLOAD` is a replacement parameter in the CLIST):

```
SASLOAD(’’’&prefix.LIBRARY’’’)+
```

with this line:

```
SASLOAD(’’’your.ORACLE.cmdload’’ ’’&prefix.LIBRARY’’’)+
```

If you are running SQL*Net Version 2 or above and you will be accessing the Oracle server as an z/OS client, you might also need to allocate the TNSNAMES file to your current SAS session. Check with your DBA to see if the TNSNAMES file is centrally maintained. If it is not, get the correct data set name for your site’s TNSNAMES file:

```
ALLOC F(TNSNAMES) DA('Your.TNSNAMES.dataset') SHR
```

If you do not have the TNSNAMES file allocated, you will receive the following error from Oracle:

```
ORACLE connection error: ORA-12154: TNSNAMES: could not resolve service
name.
```

Optionally, you might want to allocate the data set that contains the default pathname for connecting to the Oracle database server. This is especially convenient if you are always connecting to the same server. Check with your DBA for the correct DDname to the data set name.

```
ALLOC F(Your site's DDname) DA('Your.data.setname') SHR
```

STEP 2: Update your SAS REXX exec (Optional)

If the Oracle interface is used extensively at your site, you might want to alter the REXX exec so that your Oracle load library is allocated and concatenated before the SAS load library in the `TASKLIB` symbol. Add the following line to the installation-supplied SASRXCFG(REXXSITE) config file:

```
-SASLOAD('your.ORACLE.cmdload' &sasload)
```

If you are running SQL*Net Version 2 or above and you will be accessing the Oracle server as an z/OS client, you might also need to allocate the TNSNAMES file to your current SAS session. Check with your DBA to see if the TNSNAMES file is centrally maintained. If it is not, get the correct data set name for your site's TNSNAMES file, and in the SASRXCFG(REXXSITE) config file, insert the following option:

Check with your DBA to see if the TNSNAMES file is centrally maintained. If it is not, get the correct data set name for your site's TNSNAMES file, and in the SASRXCFG(REXXSITE) config file, insert an –inittso option to allocate the data set. For example, if your TNSNAMES data set is named DBI.ORACLE.V10203.DEVA.PARMLIB(TNSNAMES), add

```
-inittso "ALLOC F(TNSNAMES) DA('DBI.ORACLE.V10203.DEVA.PARMLIB(TNSNAMES)') shr"
```

If you want these allocations to be freed at SAS termination, also add

```
-termtso "FREE F(TNSNAMES)"
```

```
-termtso "FREE F(ORA$LIB)"
```

If you do not have the TNSNAMES file allocated, you will receive the following error from Oracle:

```
ORACLE connection error: ORA-12154: TNSNAMES: could not resolve service
name.
```

Optionally, you might want to allocate the data set that contains the default pathname for connecting to the Oracle database server. This is especially convenient if you are always connecting to the same server. Check with your DBA for the correct DDname to the data set name, and in the SASRXCFG(REXXSITE) config file, insert the following option:

```
-inittso "ALLOC F(Your site's DDname) DA('Your.data.setname') SHR"
```

STEP 3: Update your SAS cataloged procedure (optional)

If the Oracle interface is used extensively at your site with the SAS cataloged procedure, you might want to make the `STEPLIB DD` statement change permanent to your cataloged procedure. To always execute

SAS 9.4 Foundation using the Oracle interface, modify the `STEPLIB DD` statement in your cataloged procedure to reflect the following lines.

```
//STEPLIB      DD DISP=SHR,DSN=&LOAD
//             DD DISP=SHR,DSN=&prefix.LIBRARY

//             DD DISP=SHR,DSN=your.ORACLE.cmdload
```

where `&prefix.LIBRARY` is SAS 9.4 Foundation load library, and `your.ORACLE.cmdload` is the Oracle System load library.

If you are running SQL*Net Version 2 or later and you will be accessing the Oracle server as a z/OS client, you might also need to allocate the TNSNAMES file to your current SAS session. Check with your DBA to see if the TNSNAMES file is centrally maintained. If it is not, get the correct data set name for your site's TNSNAMES file.

```
//TNSNAMES DD DISP=SHR,DSN=your.TNSNAMES.dataset
```

If you do not have the TNSNAMES file allocated, you will receive the following error from Oracle:

```
ORACLE connection error: ORA-12154: TNSNAMES:could not resolve service
name.
```

Optionally, you might want to allocate the data set that contains the default pathname for connecting to the Oracle database server. This is especially convenient if you are always connecting to the same server. Check with your DBA for the correct `DDname` to the data set name.

```
//yourDDname DD DISP=SHR,DSN=your.dataset.name
```

Creating and Loading the Sample Tables (optional)

Refer to the SAS/ACCESS Sample Repository (sas-access-samples) GitHub project for coding examples based on sample Oracle tables that can be created at your site. Creating these tables will assist the users at your site in learning how to use SAS/ACCESS Interface to Oracle.

SAS/ACCESS Interface to R/3

SAS/ACCESS Interface to SAP R/3 requires extensive post-installation configuration before it can be used. Refer to the *Post-Installation Instructions for SAS/ACCESS 9.4 Interface to R/3* for detailed information.

SAS/ACCESS Interface to SYSTEM 2000

Customizing the SAS CLIST, REXX Exec, and Cataloged Procedure

Note: *This task is required.*

If the SYSTEM 2000 Interface is used extensively at your site, and if your SYSTEM 2000 load library is not a link list library, you might want to modify the SAS CLIST and cataloged procedure to allocate your SYSTEM 2000 load library by default. The file `S2KCOM` is required for SYSTEM 2000 Multi-User, the file `S2KPARMS` is required for SYSTEM 2000 single-user, and the file `S2KDEFC` is required for both Multi-User and single-user. These must also be allocated to your CLIST and procedure. See Steps 1 and 2 below for the changes to make.

Note: ***The concatenation order of the `SAS LOAD LIBRARY` and the load library for database access can be interchanged. Generally, if more database access activities occur in the SAS session or batch job, place the database load library first in the concatenation (or ahead of***

the SAS LOAD LIBRARY). The opposite is true when SAS processing dominates the session or JOB.

STEP 1: Add allocations to your SAS CLIST (Optional)

Replace the following line in the installation-supplied CLIST (where `SASLOAD` is a replacement parameter in the CLIST):

```
SASLOAD(’’’&prefix.LIBRARY’’’)+
```

with this line:

```
SASLOAD(’’’your.SYSTEM2000.load’’ ’’&prefix.LIBRARY’’’)+
```

In addition, allocate three SYSTEM 2000 specific files in your SAS CLIST by adding the following:

```
/* Allocate SYSTEM 2000 file for DBLOAD
ALLOC F(S2KDEFC) SP(1 1) CYL +
    LRECL(100) BLKSIZE(100) RECFM(F) REU

/* Allocate SYSTEM 2000 Multi-User specific file
ALLOC F(S2KCOM) DA(‘your.SYSTEM2000.s2kcom.file’) SHR REU

/* Allocate SYSTEM 2000 single-user specific file
ALLOC F(S2KPARMS) DA(‘your.SYSTEM2000.CNTL(NLPARM)) SHR REU
```

STEP 2: Update your SAS REXX exec (Optional)

If you need to concatenate the System 2000 load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:

```
-DBMSLIBS 'your.SYSTEM2000.load'
```

If you want the System 2000 load library concatenated before, rather than after, the SAS load library, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:

```
-DBMSCONCAT FIRST
```

In the SASRXCFG(REXXSITE) config file, insert the following option:

```
/* Allocate SYSTEM 2000 files for DBLOAD, Multi-User specific file,
   and single-user specific file */
  -inittso "ALLOC F(S2KDEFC) SP(1 1) CYL LRECL(100) BLKSIZE(100) RECFM(F) REU"
  -inittso "ALLOC F(S2KCOM) DA('your.SYSTEM2000.s2kcom.file') SHR REU"
  -inittso "ALLOC F(S2KPARMS) DA('your.SYSTEM2000.CNTL(NLPARM)') SHR REU"
```

If you would like these allocations freed at SAS termination, also add:

```
  -termtso "FREE F(S2KDEFC)"
  -termtso "FREE F(S2KCOM)"
```

```
-termtso "FREE F(S2KPARMS)"
```

STEP 3: Add allocations to your SAS cataloged procedure (optional)

Concatenate your SYSTEM 2000 LOAD library to your `STEPLIB` statement in your default SAS cataloged procedure as follows:

```
//STEPLIB DD DISP=SHR,DSN=&LOAD
// DD DISP=SHR,DSN=&prefix.LIBRARY
// DD DISP=SHR,DSN=your.SYSTEM2000.load
```

Allocate the following SYSTEM 2000 Multi-User file in your SAS procedure for Multi-User access:

```
//S2KCOM    DD DSN=your.SYSTEM2000.s2kcom.file,DISP=SHR
```

Allocate the following SYSTEM 2000 single-user specific file in your SAS procedure for single-user access:

```
//S2KPARMS DD,DSN=your.SYSTEM2000.CNTL(NLPARM),DISP=SHR
```

Allocate the following SYSTEM 2000 file in your SAS procedure for both Multi-User and single-user access:

```
//S2KDEFC DD UNIT=SYSDA,SPACE=(CYL,(1,1)),
//     DCB=(RECFM=FB,LRECL=100,BLKSIZE=100)
```

With all three `S2KCOM`, `S2KDEFC`, and `S2KPARMS` files present, your SAS procedure can access both Multi-User and single-user SYSTEM 2000.

SAS/ACCESS Interface to Teradata

Important: *SAS/ACCESS Interface to Teradata is not available in SAS 9.4M8. Support for this SAS/ACCESS engine has been restored in SAS 9.4M9.*

To use a Teradata data source with SAS 9.4 Foundation, upgrade or migrate SAS Foundation to SAS 9.4M9. Contact your SAS representative if you need assistance.

Support for Teradata on z/OS remains unsupported for SAS/ACCESS on SAS 9.4M8. If you have an existing installation of SAS/ACCESS Interface to Teradata in your environment, SAS recommends that you first unconfigure and uninstall before upgrading or migrating to SAS 9.4M9. A best practice is to unconfigure retired SAS products before you upgrade and to uninstall them after you upgrade. For more information, see "Unconfiguring and Uninstalling Retired Products" in the SAS Guide to Software Updates and Product Changes.

(For SAS 9.4M9 and Later) Configuring SAS/ACCESS Interface to Teradata

Starting with SAS 9.4M9, some changes apply to SAS 9.4 Foundation for z/OS support for Teradata. As a result, some configuration procedures have changed with SAS 9.4M9.

For environments that have not upgraded to SAS 9.4M9, documentation remains. Consult the sections below labeled "For SAS 9.4M7 and Earlier."

(For SAS 9.4M9 and Later) Defining the Interface to Teradata

Teradata CLIv2 and TDP software for z/OS must be installed before you can use SAS/ACCESS Interface to Teradata. The procedure to install this software is described in the *Teradata Tools and Utilities Installation Guide for MVS* supplied by Teradata. SAS 9.4 Foundation supports specific versions of the Teradata

database and client software. Be sure to consult the SAS/ACCESS Interface to Teradata requirements in the *SAS 9.4 Foundation for z/OS System Requirements*.

When you have completed the installation of the Teradata software, you must verify connectivity to the Teradata RDBMS. Connectivity can be tested using SAS/ACCESS, BTEQ, or the TPT utility on z/OS.

Starting with SAS 9.4M9, functions that were previously provided by the Teradata FastLoad, MultiLoad, and FastExport utilities are now provided by analogous functions of the Teradata Parallel Transport Utility (TPT). This functionality requires release 17.20 or later of Teradata Tools and Utilities (TTU). For more information about these utilities, see Support for Teradata TDP on page 71.

If you are already running Teradata TTU 17.20, you must ensure that SAS/ACCESS is configured to use the additional library that is required by the Teradata TPT utility.

In addition to installing your Teradata client software, a system administrator must have started and initialized the Teradata Director Program (TDP) before you define the interface. This communication task must be running before SAS/ACCESS Interface to Teradata can communicate with the Teradata server. Names of the TDPs must be communicated to SAS users at your enterprise if a default TDP is not established.

Note: *A typical installation might include two TDPs, one named "TDP0 - production TDP" and another named "TDP1 - test TDP." The test TDP could be used for the testing of new versions of Teradata, new applications, and more. The role of the production TDP would be to communicate with the production database server.*

The system administrator can set up a default TDP to be used by Teradata client applications, including SAS sessions. The System Parameter Block (HSISPB and HSHSPB) must be modified to indicate the default name.

Note: *This is a Teradata parameter block and not a SAS parameter block.*

When the default TDP has been defined, SAS 9.4 Foundation uses this TDP by default when connecting to the Teradata server.

(For SAS 9.4M9 and Later) Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

After the Teradata components are in place, tested, and defined, the system administrator must configure SAS 9.4 Foundation to use the Teradata load libraries. SAS/ACCESS Interface to Teradata under z/OS uses the APPLOAD and TPT.LOAD load libraries supplied by Teradata. These libraries must be accessible to SAS to enable SAS/ACCESS Interface to Teradata to function.

You have multiple options for configuring the Teradata load libraries:

- You can modify your SAS CLIST to make the necessary Teradata libraries accessible when you invoke SAS. Take the following steps:
  Edit your SAS CLIST.
  Replace `SASLOAD('''&prefix.LIBRARY''')+` with this line:

  ```
  SASLOAD('''&prefix.LIBRARY'' ''your.ncr.appload'' ''your.ncr.tpt.load''')+
  ```

- You can modify your SAS cataloged procedure by changing the `STEPLIB DD` statement as shown below:

  ```
  //STEPLIB DD DISP=SHR,DSN=&LOAD
  //                          DD DISP=SHR,DSN=&prefix.LIBRARY
  ```

```
//                              DD DISP=SHR,DSN=your.ncr.appload
//                              DD DISP=SHR,DSN=your.ncr.tpt.load
```

Support for Teradata TDP

In previous maintenance releases of SAS 9.4 Foundation, the sole protocol for connecting from z/OS to Teradata using SAS/ACCESS Interface to Teradata was the Teradata Director Program (TDP). With TDP, the client application communicates through the TDP process to a Teradata channel interface on the database that handles only mainframe connections. The physical connection can be FICON or TCP/IP. With TCP/IP, FICON messages are encapsulated for transfer over the network to the Teradata channel interface.

Mainframe channel interfaces must be added to the Teradata database system to support mainframe connections when using TDP with SAS 9.4M7 and earlier.

A more recent alternative for connecting to Teradata from z/OS is **Mainframe Gateway-Mediated (MF-GTW)**. With the MF-GTW method, the client application communicates using TCP/IP on the mainframe to the standard network-attached client gateway on the database machine. SAS intends to provide support for this method in a future release. At this time, it is not supported with SAS 9.4M9.

The MF-GTW connectivity option was introduced with TTU 17.20. MF-GTW supports TLS encryption (requires database 17.20 or higher). SAS recommends that you use TLS for encryption for all SAS 9.4 Foundation deployments.

All Teradata machines are configured with network gateways; therefore, no special configuration is required on the database side for MF-GTW.

Either connectivity method (that is, either TDP or MF-GTW) is supported with Teradata VantageCloud Enterprise version 2.4 (database 17.20). Database 20.00 is expected to be added to Teradata VantageCloud Enterprise in 2025.

MF-GTW is the only supported connectivity method for Teradata VantageCloud Lake (database 20.00).

SAS recommends consulting the following additional resources for gateway-mediated connectivity:

- Teradata Tools and Utilities for IBM z/OS Installation Guide - Gateway Mediated (MF-GTW) 20.00
- Teradata® Tools and Utilities for IBM z/OS Installation Guide - Gateway Mediated (MF-GTW) - 17.20
- Public - What is the difference between Gateway Mediated and TDP Mediated Mainframe Connectivity?
- Public - GTW Mediated Mainframe Connectivity - Minimum Release for Supportability

(For SAS 9.4M7 and Earlier) Defining the Interface to Teradata

Teradata CLIv2 and TDP software for z/OS must be installed before using SAS/ACCESS Interface to Teradata. The procedure to install this software is described in the *Teradata Tools and Utilities Installation Guide for MVS* supplied by Teradata. Once the Teradata software is installed, connectivity to the Teradata RDBMS must be verified. Connectivity can be tested with SAS/ACCESS, or with BTEQ, FastLoad, or other Teradata utilities on z/OS. If you are already running any Teradata utility, no setup is necessary; your Teradata software is already correctly configured for use by SAS/ACCESS Interface to Teradata.

Along with the installation of Teradata client software in your environment, the system administrator must have started and initialized the Teradata Director Program (TDP). This communication task must be running before the SAS/ACCESS Interface to Teradata product can communicate with the Teradata server. Names of the TDPs must be communicated to the SAS user community if a default TDP (see below) is not established.

Note: *A typical installation might include two TDPs; one named "TDP0 - production TDP" and another "TDP1 - test TDP." The test TDP could be used for the testing of new versions of Teradata, new applications, etc. The production TDP's role would be to communicate with the production database server.*

The system administrator can set up a default TDP to be used by Teradata client applications, including SAS sessions. To do this, the System Parameter Block (`HSISPB` and `HSHSPB`) must be modified to indicate the default name.

Note: *This is a Teradata parameter block and not a SAS parameter block.*

Once the default TDP is defined, SAS will use this TDP by default when connecting to the Teradata server.

(For SAS 9.4M7 and Earlier) Customizing the SAS CLIST, REXX exec, and Cataloged Procedure

After the Teradata components are in place, tested and defined, the system administrator needs to configure SAS 9.4 Foundation to use the Teradata load libraries. SAS/ACCESS Interface to Teradata under z/OS uses the `APPLOAD` load library supplied by Teradata. This library must be accessible to SAS in order for SAS/ACCESS Interface to Teradata to function. There are several methods that can be used to accomplish this task:

- Enter the following command under TSO to start a SAS session in which you will execute the SAS/ACCESS Interface to Teradata under z/OS interface:

  ```
  SASname LOAD('''your.ncr.appload''')
  ```

  or

  ```
  SASRXname -DBMSLIB 'your.ncr.appload'
  ```

  where `your.ncr.appload` is the Teradata system load library and `SASname` is the name of the CLIST or SASRXname is the name of the REXX exec you use to invoke SAS.

- To use the Teradata interface with the SAS cataloged procedure, use the `LOAD=` parameter of the PROC to specify your site's Teradata load libraries. For example, you can use the following code in your JCL:

  ```
  //SASTRA   EXEC  SASname,LOAD='your.ncr.appload'
  ```

  where `your.ncr.appload` is the Teradata system load library and `SASname` is the name of the cataloged procedure you use to invoke SAS.

- You can modify your SAS CLIST to make the necessary Teradata libraries accessible when you invoke SAS. Follow these steps to do this:
  Edit your `SAS CLIST`.

  Replace `SASLOAD('''&prefix.LIBRARY''')+` with this line:
  `SASLOAD('''your.ncr.appload'' ''&prefix.LIBRARY''')+`

- To concatenate the Teradata load library in the SAS tasklib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
  ```
  -DBMSLIBS 'your.NCR.appload'
  ```
  If you want the Teradata loadlib concatenated before, rather than after, the SAS loadlib, insert the following option specification into the installation-supplied SASRXCFG(REXXSITE) config file or on the REXX exec command line:
  ```
  -DBMSCONCAT FIRST
  ```
- You can modify your SAS cataloged procedure by changing the `STEPLIB DD` statement as shown below:
  ```
  //STEPLIB  DD DISP=SHR,DSN=&LOAD
  //               DD DISP=SHR,DSN=&prefix.LIBRARY
  //               DD DISP=SHR,DSN=your.ncr.appload
  ```

Creating and Loading the Sample Tables (optional)

Refer to the SAS/ACCESS Sample Repository (sas-access-samples) GitHub project for coding examples based on sample Teradata tables that can be created at your site. Creating these tables will assist the users at your site in learning how to use SAS/ACCESS Interface to Teradata.

TPT or Legacy Teradata Utility (FastExport, FastLoad, and Multiload)

Important *Starting in SAS 9.4M9, Teradata capabilities that require the Legacy Teradata Utility are no longer supported. FastExport, FastLoad, and MultiLoad are no longer supported. The TPT utility is used instead, even when TPT=NO. Documentation remains for users who have not upgraded to SAS 9.4M9.*

The following are the advantages of using TPT instead of the Legacy Teradata utility:

- High-performance, multi-session interface for bulk load, bulk update, and bulk export data transfer.
- TPT API allows SAS to perform in-stream processing via API calls; no control files or intermediate data movement.
- Enhanced performance now, improving in the future.
- Configuration is less complex; no loading and launching of separate Teradata utility processes, no inter-process communication links required.
- TPT is integrated with Teradata Active System Management and Teradata Multi Systems Manager suite of support products.

Legacy Teradata Utility Configuration

FastExport (optional)

Important *Starting in SAS 9.4M9, FastExport, FastLoad, and MultiLoad are no longer supported. Documentation remains for users who have not upgraded to SAS 9.4M9.*

For optimal reads of large tables, SAS/ACCESS can perform "quick exporting." To perform quick exporting, the Teradata FastExport utility must be present on the system where you install SAS. The FastExport utility is not required by SAS 9.4 Foundation; SAS/ACCESS reads large tables quite efficiently without it. For more information, see the `DBSLICEPARM` option in *SAS/ACCESS 9.4 for Relational Databases: Reference*.

Contact Teradata if you want to obtain the Teradata FastExport utility.

MultiLoad (optional)

Important *Starting in SAS 9.4M9, FastExport, FastLoad, and MultiLoad are no longer supported. Documentation remains for users who have not upgraded to SAS 9.4M9.*

SAS/ACCESS can interface with MultiLoad for loading large volumes of data. To perform a MultiLoad operation, the Teradata MultiLoad utility must be present on the system where you install SAS. The MultiLoad utility is not required by SAS 9.4 Foundation; SAS/ACCESS provides other options for loading tables. For more information, see the MULTISTMT option in *SAS/ACCESS 9.4 for Relational Databases: Reference*.

Contact Teradata if you want to obtain the Teradata MultiLoad Utility.

Chapter 4 – Post-Installation Configuration for SAS/ASSIST

This chapter describes how to add an optional master profile to SAS/ASSIST software. You can use a master profile to override the default settings as sent by SAS. This allows you to provide a customized setup for SAS/ASSIST software. With the master profile, you can control the profile options of all SAS/ASSIST users from one central place. For more information on the profile options, refer to *SAS/ASSIST Software: Changes and Enhancements*.

Adding a Master Profile

Complete the following steps to add a master profile to SAS/ASSIST software:

1. Specify the location of the master profile by creating a new SAS library to which all users of SAS/ASSIST software will have read access. The following definitions are needed:

   Data Set Name `prefix.ASSIST.MASTER`
   Space units `CYLINDER`
   Primary quantity `1`
   Secondary quantity `1`
   Directory blocks `0`
   Record format `FS`
   Record length `6144`
   Block size `6144`

   Example: `prefix.ASSIST.MASTER`

   All users with write-access to this library automatically also have write-access to the master profile in SAS/ASSIST software. Select a name that conforms to the naming conventions at your installation. The name of this new library must be stored in an entry in the SASHELP library. This requires that you have write access to the SASHELP library.

   - Modify the CLIST provided with installation to change the disposition of the SASHELP library to OLD.
   - Invoke SAS. Only one user at a time will be able to run SAS using the modified CLIST.
   - On line 1 of the Program Editor type the physical pathname to be used as the SAS data library to store the master profile. Execute the `Save` command to save the entry as shown in the following example:

     SAVE SASHELP.QASSIST.PARMS.SOURCE

     ```
     00001 prefix.ASSIST.MASTER
     00002
     ```

   Note: The entry must be saved with the name SASHELP.QASSIST.PARMS.SOURCE.

   The location of the master profile is now known by SAS/ASSIST software.

   - Exit SAS.
   - Free the SASHELP data set to release the exclusive lock:

     FREE DS('&prefix.SASHELP')

   - Modify the CLIST provided with installation to change the disposition of the SASHELP library back to SHR. Now other users can run SAS concurrently.

2. Create the master profile.

   The first time SAS/ASSIST software is started a master profile is created if the SASHELP.QASSIST.PARMS.SOURCE contains the name of an existing library, and the person who starts SAS/ASSIST software has write-access to this library.

3. Customize the master profile by starting SAS/ASSIST and typing the command `ASSIST` on the SAS command line. Then select

   `Setup ... Profiles ... Master/group ...`

   If you have Write-access to the SAS data library containing the master profile, you can specify default values for your installation. These values will be used by new users as they start SAS/ASSIST software.

Note: *If you restrict values by typing* `R` *in* `Status`*, users will not be allowed to change the values you define.*

You can run SAS/ASSIST software in two different styles – workplace or block menu. The block menu can be new style or old style. You can control this using the profile options below.

Run workplace:

```
SAS/Assist style:          Workplace
```

Run block menu new style:

```
SAS/Assist style:          Block Menu
Save selections on end:    Yes
Menu Style:                New
```

Run old style:

```
SAS/Assist style:          Block Menu
Save selections on end:    Yes
Menu Style:                Old
```

4. Create Group profiles.

   From the master profile, it is possible to create group profiles to allow groups of users to have different setups. The master profile controls group profiles and user profiles when a user is not a member of any group. All users are indirectly controlled by the master profile when option values are set to a restricted
   (R) status.

   From `Setup ... Profile ... Master/Group ...` select `Tools ... Create Group Profile`. To add users to a group profile, select `Tools ... Update User Group`. By default, the user ID is found in the macro variable `&SYSJOBID`. The value is set in the option `Userid` in the master profile (option type `System Administration`). Change the value if your site uses another variable to keep the user ID.

Installing Sample DB2 Tables and a Sample Query Manager

This section describes how to customize SAS/ASSIST software to run queries against DB2 sample data. SAS recommends that you do this in order to get acquainted with the software before you start working

with your own Db2 data. This example sets up a personal query manager. See Chapter 9, "Setting Up Query Managers," in *SAS/ASSIST Software: Changes and Enhancements* for more information.

Complete the following steps:

1. Create Db2 Sample Tables.

   Query and Reporting contains its own sample Db2 tables. These tables are used in the documentation and in the SAS training courses. Complete the following steps to create DB2 sample tables:

   a. Invoke SAS and enter the following on the command line in the Program Editor:

      ```
      COPY SASHELP.QASSIST.SAMPLTAB.SOURCE
      ```

   b. Specify the DB2 subsystem id, creator, and database name (lines 54-59):

   See documentation in the program for further information.

   c. Execute this SAS program and the Db2 sample data will be created.

2. Create Query Manager Tables on the Db2 Sample Data

   Complete the following steps to create query manager tables for installation verification purposes:

   a. In the Program Editor, use the `COPY` command to copy the sample source from `SASHELP.QASSIST.DB2METAU.SOURCE`. This program will generate a set of Query Manager tables, which contain information on Db2 tables available to specific creators.

   b. Specify the creators you want to use in line 21 as shown in the following example:

      ```
      %let creators=('SASQR','DSN9030',USER);
      ```

Note: *In this example,* `SASQR` *and* `DSN9030` *are two specific creators which contain sample data from SAS and IBM. Add* `USER` *if you want access to a DB2 table, which has a creator equal to your user ID.*

   c. Specify the location of the Query Manager tables on line 25.

      ```
      %let saslib=SASUSER; *<--User QM library;
      ```

   d. Execute this SAS program and the Query Manager tables will be created. The following SAS tables will be created:

SASUSER_DB2TAB	Db2 table information
SASUSER_DB2COLS	Db2 column information
SASUSER_DB2RELS	Relations for joining tables
SASUSER_DB2INDX	Defined indexes in Db2.

3. Create the Query Manager Definitions

   The Query Manager contains information that indicates the database to be used, as well as the location of the SAS Query Manager tables created in the last step. You can also specify the name of the program to be used to recreate (refresh) the SAS Query Manager tables. See Chapter 9 in *SAS/ASSIST Software: Changes and Enhancements* for more information.

a. Select the following from the SAS/ASSIST Primary menu to get to the Query Manager Administration window:

```
DATA MGMT...
QUERY...
Query and Reporting...
SETUP...
Administration
```

b. Enter the following information to identify the location of the Query Manager tables as shown below:

- For Database, enter `DB2`
- For Program to Generate Query Manager tables, select `No`
- For Location of the Query Manager tables, enter `SASUSER`

Note: *The remaining parameters are selected by default.*

c. Save the Query Manager definition. Select **Save As** from the File menu.

d. Enter name and description as shown in the following example (the Type and Catalog are already specified):

```
Type:        MANAGER
Name:        db2samp
Description: Sample DB2 data
Catalog:     SASUSER.MANAGER
```

e. Select OK to save it.

4. Using the Query Manager:

a. Select the following from the SAS/ASSIST Primary menu to get to the Query window:

```
DATA MGMT...
QUERY...
Query and Reporting...
QUERY...
```

The Query window might attempt to load a Query Manager other than the one you defined (in Step 3 above), or you might be notified that one or more Query Manager tables are not found. You are prompted to generate them. Select `No`.

b. To use the Query Manager that you defined (see Steps 2 and 3 above), select `Select Query Manager...` from the `Tools` menu.

A pop-up menu is displayed. Select the Query Manager that you just defined DB2SAMP (in Step 3 above).

c. Click the **Select** button to get a list of Db2 data.

Chapter 5 – Post-Installation Configuration for SAS/CONNECT

Note: *The post-installation setup for the SAS/CONNECT Spawner for z/OS is different than previous releases. Read this entire section carefully.*
For additional information about the communication aspects of implementing and using SAS/CONNECT software, refer to the SAS/CONNECT 9.4 User's Guide.

To use SAS/CONNECT software, both the local and remote hosts must be running and licensed for SAS/CONNECT.

Installation of the SAS 9.4 Foundation SVC is **absolutely required** if you are implementing SAS/CONNECT Spawner for z/OS or SAS/CONNECT to the same multi-process machine on z/OS. For more information, see "Installing the SAS 9.4 SVC Routine" on page 3434.

Configuring SAS/CONNECT

This chapter shows you how to configure SAS/CONNECT. The following topics are covered:

- Communication access methods supported for SAS/CONNECT on z/OS
  - TCP (used with basic telnet session and with the z/OS SAS/CONNECT Spawner)
  - XMS (used in Configuration of SAS/CONNECT to same multi-processor machine)
- Setting up TLS
- Storing and locating SAS/CONNECT Script files
- Connection types available with SAS/CONNECT on z/OS:
  - SAS/CONNECT Basic Telnet session
  - SAS/CONNECT Spawner
  - SAS/CONNECT to same multi-processor machine

Communication Access Methods for SAS/CONNECT on z/OS

The access method to use depends on the type of connection being set up. The access methods supported for z/OS are TCP/IP and XMS. Refer to the appropriate sections for the access methods you will be using at your site, for requirement information. Refer to the *SAS/CONNECT 9.4 User's Guide* for additional information about the access methods supported by other systems.

System Configuration for the TCP Access Method

To use the TCP/IP access method with SAS/CONNECT software, you must have installed and configured the IBM z/OS Communications Server package, or another vendor's software that is functionally compatible with the IBM product. For more information about configuring TCP/IP for use with SAS/CONNECT, refer to "System Configuration for Using SAS with TCP/IP" on page 1818.

System Configuration XMS (Cross Memory Services) Access Method

For more information about configuring XMS, refer to "SAS/CONNECT to the Same Multi-Process Machine on z/OS" on page 8686.

Setting Up TLS for SAS/CONNECT

SAS strongly recommends configuring encryption to secure data and communications flows in the SAS deployment. TLS is the preferred mechanism to provide encryption for data in motion.

Starting with SAS 9.4M8, SAS Foundation servers use supported cryptographic libraries that are available on the host operating system to provide encryption for data in motion.

For IBM z/OS with SAS 9.4M8 and later, SAS no longer provides OpenSSL libraries to support TLS. Instead, SAS uses IBM System SSL to provide cryptographic services for z/OS.

You have multiple options for customizing encryption for data flows between SAS Foundation components, particularly when SAS/CONNECT and SAS/CONNECT Spawner are used. SAS recommends a few workarounds for these environments because spawned SAS/CONNECT Servers run under different user IDs, which might conflict with some IBM encryption technologies.

Additional configuration is required to enable encryption using custom certificates and IBM System SSL. For requirements, step-by-step instructions, and examples, see Configure TLS Certificates and FIPS on z/OS in *Encryption in SAS 9.4*.

Storing and Locating SAS/CONNECT Script Files

Several sample script files are shipped with SAS/CONNECT software. SAS/CONNECT software uses these script files to establish a connection to a remote SAS session by physically logging the user on to the remote host. The installation process places the script files into your `'&prefix.CTMISC'` data set on z/OS. These script files must be customized to match the logon procedures that are specific to your site. Script files are used to establish a SAS/CONNECT basic Telnet session and can optionally be used to establish a SAS/CONNECT Spawner session.

To use the script file with the SAS/CONNECT product, a file reference of RLINK must be allocated to a customized script file. Additional methods of allocating the script file are available. For more information, refer to the *SAS/CONNECT 9.4 User's Guide*.

The SAS option, `SASSCRIPT`, points to the location of the SAS/CONNECT script files for Windows and UNIX client hosts. The `SASSCRIPT` option is only used by SAS/ASSIST and by user-written SCL applications.

The value for `SASSCRIPT` can be one or more concatenated data set names (including PDS files):

```
SASSCRIPT=('MVS:prefix.CTMISC' 'userid.CTMISC')
```

The simplest way for you to give your users automatic access to the installed sample script files is to place the above option in your configuration file. The value of `SASSCRIPT` is not set by default on z/OS.

Note: *The script file must be customized to match the logon process to your remote host. Connecting to the remote host via Telnet outside of SAS is recommended to see the necessary screens and messages that need to be handled by the script.*

Types of Connections Available with SAS/CONNECT on z/OS

SAS/CONNECT Basic Telnet Session

Note: *The SAS/CONNECT basic Telnet session requires configuring SAS for communication with TCP/IP. Be sure to complete the steps in "System Configuration for Using SAS with TCP/IP" on page 1818.*

Requirements:

- Telnet must be enabled on both local and remote hosts (either in line mode or fullscreen/TN3270) in order to use the basic Telnet SAS/CONNECT sign-on.
- A customized script needs to be set up on the local host. See the previous section, "Storing and Locating SAS/CONNECT Script Files," for more information.
- A valid user ID for the remote host.

SAS/CONNECT Spawner for z/OS

Note: The SAS/CONNECT Spawner for z/OS requires you to configure SAS for communication with TCP/IP. If necessary complete the steps in "System Configuration for Using SAS with TCP/IP" on page 1818.

Benefits of Using the SAS/CONNECT Spawner

There are several benefits to using the SAS/CONNECT Spawner. Use of the spawner:

- does not require Telnet.
- does not require being physically logged on to TSO. Since this is not a TSO logon session, the user can sign on several times concurrently.
- does not require using a sign-on script (it is optional).
- initiates SAS/CONNECT sessions on z/OS systems without requiring user name and password pairs to be passed over the network in clear text.
- encrypts all data that flows from the local host to the spawner program during sign-on by default (if the local SAS session is running Release 6.09E, Release 6.11 TS040, or a subsequent release,). The SAS/CONNECT Spawner program supports encrypted sign-ons, by default, to a z/OS system with or without scripts.
- facilitates use of SAS/CONNECT through a firewall. The spawner supports socket inheritance thereby limiting the number of open ports. Socket inheritance is enabled by default.

SAS/CONNECT Spawner Security Configuration

Security requirements for the SAS 9.4 SAS/CONNECT Spawner running on z/OS differ from previous SAS releases. The spawner no longer needs to run with a uid=0. Load libraries do not need to be APF-authorized. And the user ID of the spawner process does not require READ access to the BPX.DAEMON RACF profile.

The spawner runs as a daemon process, so correct daemon security needs to be implemented based on the security software running on your machine (RACF, ACF2, or TOP Secret). IBM defines two levels of security for daemon processes, traditional UNIX security and z/OS UNIX System Services security. For SAS/CONNECT Spawner 9.4, these daemon processes do not require a UID of '0' but should be loaded from a program-controlled library if the BPX.DAEMON is defined.

The spawner validates user IDs and passwords using the __passwd service. It then starts an address space for the SAS/CONNECT session, using the USS spawn service.

Spawner Security Requirements for Client/User

- An OMVS segment defined in the security software.
- Read Access for the load library.
- Update access to the HFS logs defined in the started task.
- If the BPX.DAEMON is defined, the `&prefix.LIBRARY` must be under program control.

The SAS/CONNECT Spawner uses the UNIX System Services __passwd service to validate the user ID and password of clients connecting to the spawner. If the client authenticates successfully, the spawner uses the USS spawn service to invoke a remote SAS session for the client.

What follows are examples of security definitions for RACF. If your site uses ACF2 and TOP Secret security products, consult the associated security software documentation for configuring UNIX System Services daemon security.

RACF Security

The RACF Security Server defines the BPX.DAEMON profile of the RACF facility class to allow a daemon program to spawn processes with user IDs different from that of the daemon. This allows the site to run a daemon and associate it with a minimally-authorized user ID that does not have root user (uid=0) authority. To assign a user ID to the started task, either add the started task to the RACF Started Procedures Table ICHRIN03, or define a profile for the started task in the RACF started class.

If RACF program control is enabled on your system and if the BPX.DAEMON profile is active, RACF requires that all modules loaded into the spawner address space come from RACF program-controlled libraries. This applies to the SAS load library (`'prefix.LIBRARY'`). If this is not set up correctly, the spawner will receive a USS JREnvDirty reason code on the __passwd call. These messages will be issued to the system log.

```
ICH420I PROGRAM xxxxxxxx CAUSED THE ENVIRONMENT TO BECOME UNCONTROLLED.
BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR DAEMON (BPX.DAEMON)
PROCESSING.
```

To diagnose a dirty address space, look at informational APAR II08176 and the IBM documentation it references.

To list the status of Program Control, issue the following command:

```
SETROPTS LIST
```

If the results of the above command list `WHEN(PROGRAM)` in the `ATTRIBUTES`, RACF Program Control is already active. Otherwise, to enable Program Control, use the following command:

```
SETROPTS WHEN(PROGRAM)
```

When Program Control is enabled, all modules loaded into the spawner's address space must come from a RACF Program-Controlled Library. Here is the command to make the SAS load library program controlled:

```
RDEFINE PROGRAM ** UACC(READ) ADDMEM +
('prefix.SAS.LOAD'//NOPADCHK)
```

Use RALTER command instead of REDEFINE if * or ** already exists from a previous definition.

To refresh any changes, issue the following RACF command. It does not matter whether the RACF databases are shared.

```
SETROPTS WHEN(PROGRAM) REFRESH
```

To enable the RACF BPX.DAEMON profile of the FACILITY class, issue the following commands:

```
RDEFINE FACILITY BPX.DAEMON UACC(NONE)
SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST(FACILITY)
SETROPTS RACLIST(FACILITY) REFRESH
```

Note that the spawner user ID does not require READ access to the BPX.DAEMON profile. For more information, refer to your *z/OS Security Server (RACF) Command Language Reference*.

Setting up the SAS/CONNECT Spawner Procedure

The SAS/CONNECT Spawner runs as a z/OS started task and uses z/OS UNIX System Services (USS) to start each user's SAS/CONNECT session. Each session runs in a BPXAS address space, executing the UNIX System Services `/bin/tso` command to run the SAS REXX startup command.

Note: *The `/bin/tso` command mentioned above is used by default in the UNIX shell script. If you need to run authorized commands, use the `/bin/tsocmd` command instead. See Usage Note 54530 for information about setting the `/bin/tsocmd` command in the UNIX shell script.*

The spawner module is CNTSPAWN and is located in &prefix.LIBRARY. This library might be installed in LPA, LINKLIST, or allocated to the STEPLIB DD in the Spawner Started Task JCL. A sample spawner started task procedure can be found in '`&prefix.BAMISC(SPNCCNTL)`'.

The Spawner Started Task requires a parms file. A sample parms file can be found in '`&prefix.BAMISC(SPNCPARM)`'. Here is the syntax for some of the options you might specify in the parms file to configure the SAS/CONNECT Spawner program for z/OS:

```
<-HELP>
<-NETENCRALG algorithm>
<-NETENCRYPT>
<-NETENCRKEY n>
<-NOCLEARTEXT>
<-NOSCRIPT>
<-SASCMD command>
<-SERVICE service-name>
<-MGMTPORT 7555>
```

where:

- `MGMTPORT` (SAS 9.4 and later) enables you to specify the service name or port number that will listen for operator connections. Operator connections are connections made through the operator port. The operator port is a unique port number that is used for administrative purposes. Range 1- 6553
- `NETENCRKEY <keysize>` specifies the number of bits in data encryption keys.
- `NETENCRYPT<algorithm>` specifies that network encryption is required. `NETENCRYPTALGORITHM` (alias: NETENCRALG) specifies the algorithm or algorithms to be used for encrypted client/server data transfers.
- `NOCLEARTEXT` prevents sign-ons from clients that do not support user ID and password encryption. Starting with SAS 9.4M5, this option is no longer valid. It prevents clients that are running older releases (prior to SAS 6.09E and SAS 6.11 TS040, which do not support user ID and password encryption) from signing on to the spawner program. However, the default permits both encrypted and plaintext user IDs and passwords.
- `NOSCRIPT` prevents sign-on from clients that use scripts and allows sign-on only from clients that do not use scripts.

NOSCRIPT can be useful if you want to limit SAS start-up commands to the use of the -SASCMD option or to commands defined in metadata. Specifying -NOSCRIPT restricts clients from specifying additional options in SAS start-up commands.

- SASCMD | -CMD<command> specifies a UNIX System Services (USS) shell script for starting a SAS session. You must use -SASCMD and a shell script if you do not specify a sign-on script in the client session using an RLINK fileref. The script interprets the command arguments and environment variables and builds a TSO command that invokes a SAS session. Use the -SASCMD option to do the following:
  - invoke SAS from a directory that is not the default location
  - specify different SAS start-up command options
  - execute other statements before invoking SAS
- SERVICE *port-number*| *service-name* specifies the service name or port number to use to listen for client connections. The -SERVICE option values that are used to start the spawner determine client sign-on. In the following example, the spawner is started by specifying the port number as the value of the -SERVICE option during spawner start-up:

  ```
  SAS-installation-directory\SASFoundation\9.4\cntspawn.exe" -service 5020
  ```

  The client can then sign on by specifying the explicit port-number in the SIGNON statement:

  ```
  signon node-name.5020 -mgmtport 5030
  ```

  ***Note**: If the -SERVICE option is not specified, the spawner listens on the Telnet port (23).*
- TRACE | VERBOSE turns on trace level output.
  In SAS 9.4, the TIMESTAMP option is no longer supported.

For complete documentation about the z/OS Spawner's supported options, refer to the *SAS/CONNECT 9.4 User's Guide*.

Defining the SAS Startup Shell Script

In SAS 9, the SAS/CONNECT Spawner does not execute a TSO CLIST directly as the -SASCMD, but instead as a UNIX System Services (USS) shell script. The spawner invokes the USS shell script specified in either the SAS/CONNECT sign-on script or the -SASCMD spawner parameter. Here is an example of a -SASCMD:

```
-sascmd "SASHOME/spawnsas.sh -nosasuser -dmr -noterminal
-nosyntaxcheck -comamid tcp"
```

In this example, the command assumes a shell script named spawnsas.sh is installed in SASHOME. The command specifies the NOSASUSER SAS REXX option, and four SAS options: DMR, NOTERMINAL, NOSYNTAXCHECK, and COMAMID=TCP.

The shell script interprets the parameters passed in by the spawner and builds a TSO command for invoking SAS. The following is an example shell script. This same example can be found in '&prefix.BAMISC(SPNCSHEL)'. The sample shell script can be copied to USS using the TSO OCOPY or OPUT commands:

```
OPUT '&prefix.BAMISC(SPNCSHEL)' 'SASHOME/spawnsas.sh' text
```

Every connecting user ID must be able to read/exec the USS shell script. The required USS permissions for the shell script must be 755.

Additional TKMVSENV Configuration Note

To ensure responsiveness of SAS spawners and servers to various conditions outside of normal request processing, when idle they periodically awaken to check for requests. The intervals in seconds for these checks are governed by two variables. Generally, the default settings should be acceptable. However, if you wish to configure the intervals, set them in the TKMVSENV file. A value of zero for each of these variables means "remain idle and only awaken for request processing". Set these variables in the TKMVSENV file as follows:

```
set CONNECTWDWAIT=0
set TCP_POLL_INTERVAL=0
```

See the section "TK Options" in the Architectural Changes Overview for more information regarding the specification of the TKMVSENV data set. These variables can be set in both the spawner and in the SAS server sessions it spawns.

Starting/Stopping the SAS/CONNECT Spawner for z/OS Program

To start the spawner, enter the following operator console command:

```
START SPAWNER
```

To stop the spawner, enter the following operator console command:

```
STOP SPAWNER
```

Note: *The spawner continues to run until it is stopped.*

Examples of Starting and Connecting to the z/OS Spawner Program

The following examples illustrate how to start the spawner program and how to connect to it.

Example 1

The following z/OS command starts the spawner program at the remote z/OS host with the service-name spawner and disallows clear-text sign-ons from local hosts that use a script.

START SPAWNER

It uses the following PARMS file:

```
-service spawner
-mgmtport 7555
-nocleartext
```

At a local host, the following statements specify a script file named tcpmvs.scr that makes a connection to the spawner on the machine RMTHOST. The host is listening on the port that is named *spawner*. The value for `REMOTE=` is the host name of the z/OS node, or it can be a macro variable that contains that host name, where the spawner is running.

```
options comamid=tcp;
filename rlink '!sasroot\connect\saslink\tcpmvs.scr';
signon rmthost.spawner;
```

Example 2

In the following examples, the command specified in the `-sascmd` option is a USS (Unix System Services) shell script that invokes a SAS session. The following z/OS command starts the spawner program at the remote z/OS host:

```
START SPAWNER
```

It uses the following PARMS file:

```
    -service spawner
    -mgmtport 7555
    -noscript
    -netencralg rc2
    -netencralg des
-sascmd "/usr/lpp/SAS/spawnsas.sh -nosasuser -dmr -noterminal -nosyntaxcheck
-comamid tcp"
```

Important: *`-inheritance` is no longer a valid PARMS file specification because this is the default action for the SAS/CONNECT Spawner. If `-inheritance` is set in your parms file, you will see the following error when starting the SAS/CONNECT Spawner task:*

```
ERROR: An unknown command line option inheritance was specified. Review
the doc for resolution.
```

Remove `-inheritance` from your parms file and restart your SAS/CONNECT Spawner.

At a local host, the TCP/IP access method is used to connect to the remote host named RMTHOST. This must be either the host name of the z/OS node, or a macro variable that contains that host name, where the spawner program is running. The `USER=` option in the `SIGNON` statement prompts the user for a user ID and password when connecting to RMTHOST, where the z/OS program named spawner runs:

```
options comamid=tcp;
signon rmthost.spawner user=_prompt_ ;
```

SAS/CONNECT to the Same Multi-Process Machine on z/OS

This sign-on type was formerly documented as "MP CONNECT." "Multi-Process (MP) CONNECT" exploits a local host's multi-processor capability, as well as multi-processors across a network, by allowing parallel processing of self-contained tasks and the coordination of all the results into the original SAS session. SAS/CONNECT accomplishes multi-processing by means of asynchronous rsubmits.

When MP CONNECT processing needs to be done on the same host as the local host, a SASCMD sign-on might be used to initiate one or more "remote" SAS sessions. For z/OS hosts, the SASCMD sign-on uses the cross-memory services (XMS) access method. The `SASCMD=` option is used for specifying options and values that are passed to a dynamically created DMR session. For details on usage of the MP CONNECT feature, refer to the *SAS/CONNECT 9.4 User's Guide*.

System Configuration for SAS/CONNECT to Same Multi-Processor Machine

To implement the XMS access method, which is used by this type of sign-on, complete the following steps:

STEP 1: Install the SASVXMS Load Module.

This procedure is described on page 107107 of "Chapter 10 — Post-Installation Configuration for SAS/SHARE." The SAS/CONNECT to same multi-processor machine and SAS/SHARE both require the installation of the SASVXMS module. During the installation of SAS/SHARE, you must rename the SASVXMS0 load module to SASVXMS (removing the 0).

STEP 2: Define an Anchor Point.

This procedure is described on page 108108 of "Chapter 10 – Post-Installation Configuration for SAS/SHARE."

STEP 3: Install the SAS 9.4 Foundation SVC Routine.

This procedure is described in "Installing the SAS 9.4 SVC Routine" on page 3434.

Considerations for Using XMS with SAS/CONNECT to Same Multi-Processor Machine

System administrators should note that the remote (DMR) SAS sessions spawned by MP CONNECT will be connected to their local client z/OS session exclusively with the cross-memory access method.

Like XMS SHARE servers, these DMR server sessions will have their z/OS ASID marked "temporarily non-reusable" by the operating system when their address space terminates. The DMR address space will be terminated in response to the signoff command. The ASID will remain non-reusable until the client address space is terminated -- when the client TSO session logs off, started task ends or batch initiator is drained. Customers should therefore follow IBM recommendations by not spawning DMR sessions from long-running address spaces.

In addition, customers should guard against spawning excessive numbers of DMR sessions from a single client session. A complete discussion of how z/OS manages cross memory ASIDs can be found in section 3.9 of *z/OS V1R9.0 MVS Extended Addressability Guide (SA22-7614-05)*. SAS XMS servers, both SHARE and DMR, create non-system LXs for purposes of that discussion.

Note also that the spawning of XMS DMR session utilizes z/OS UNIX System Services, including the BPX1FRK interface (UNIX fork support). If these services are not present in the host z/OS system and usable by the client session, the DMR server creation fails.

Chapter 6 – Post-Installation Configuration for SAS/IntrNet

This chapter has information for your SAS/IntrNet installation. It will help you install, configure, and test your SAS/IntrNet components.

The procedures for installing SAS software using the SAS Deployment Wizard are described in other documentation and not available from this chapter. Furthermore, the installation of your web server is your responsibility and not described in SAS documentation.

When the SAS/IntrNet software has been installed, configured and tested using the procedures described in this chapter, review the latest version of the SAS/IntrNet product documentation online at `http://support.sas.com/documentation/onlinedoc/IntrNet/index.html`.

SAS/IntrNet software has the following prerequisite on z/OS: SAS/IntrNet requires that SAS be configured for communication with TCP/IP. Make sure that the steps in "System Configuration for Using SAS with TCP/IP" on page 1818 have been completed.

Overview

All SAS/IntrNet installations are made up of two components:

- The SAS/IntrNet server (also referred to as the Application Server). This is where SAS Foundation is installed.
- CGI Tools (also referred to as the Broker). This is where the broker.cfg file and its supporting files are installed.

When you install SAS/IntrNet, select one of the two installation configurations:

- Type A - The SAS/IntrNet server and CGI Tools components are both installed on the same system machine. The web server **must** be installed before starting the SAS installation.
- Type B -The SAS/IntrNet server component is installed on one system machine and the CGI Tools component is installed on a different system machine. The web server **must** be installed on the CGI Tools system machine prior to installing CGI Tools.

Type A and Type B require different installation steps:

Type A Installation Steps
Confirm that the web server software (IIS, Apache etc.) is on the same system machine as your SAS/IntrNet software.
Install your SAS products, including "CGI Tools for the Web Server."
Test the Broker

Type B Installation Steps
Confirm that the web server software (IIS, Apache, etc.) is on the system machine that you will install the CGI Tools to.
On your application server system machine, start your SAS installation. Uncheck "CGI Tools for the Web Server" in the "Select Products to Install" menu. On your web server system machine, start your SAS installation. Uncheck all products except "CGI Tools for the Web Server" in the "Select Products to Install" menu. You can optionally check the IntrNet Monitor or Connect Drivers
Test the Broker

Type A Installation Steps
Configure a Socket Service
Start the Socket Service
Test the Socket Service

Type B Installation Steps
Configure a Socket Service
Start the Socket Service
Test the Socket Service

The steps are described more thoroughly in the sections that follow.

Installing and Configuring SAS/IntrNet

Install Your Web Server Software

Refer to your web server's documentation for its installation procedures.

Install Your SAS Software

Refer to the step-by-step z/OS installation instructions on SAS Install Center to perform your SAS software installation.

If you are performing a Type A installation (as described in the Overview, above), confirm that your web server software is installed before starting your SAS software installation.

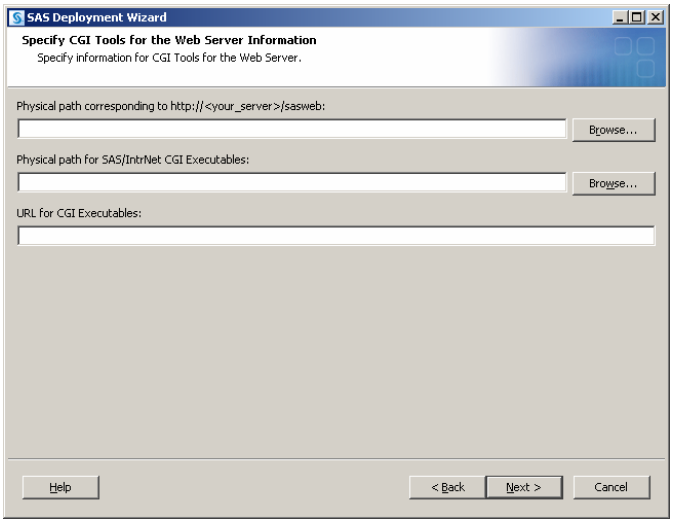
If you are performing a Type B installation, do **both** of the following:

- Install the SAS software on the SAS System side.
- Start the SAS software installation on the web server and select **CGI Tools for the Web Server** in the Select Products to Install dialog box.
  SAS/IntrNet Monitor and SAS/CONNECT Driver for Java are optional selections. Clear all other options.

CGI Tools Installation

The following screens appear for CGI Tools for the Web Server for all installations. Click **Help** on any dialog box for information about the fields.

Customary entries are documented following each dialog shown below. Customize the entries for your environment.



The following are examples of common entries for popular web servers. Customize your entries according to your own web server environment. These fields will tell SAS where your web server software is located.

Physical path corresponding to `http://<your_server>/sasweb`:

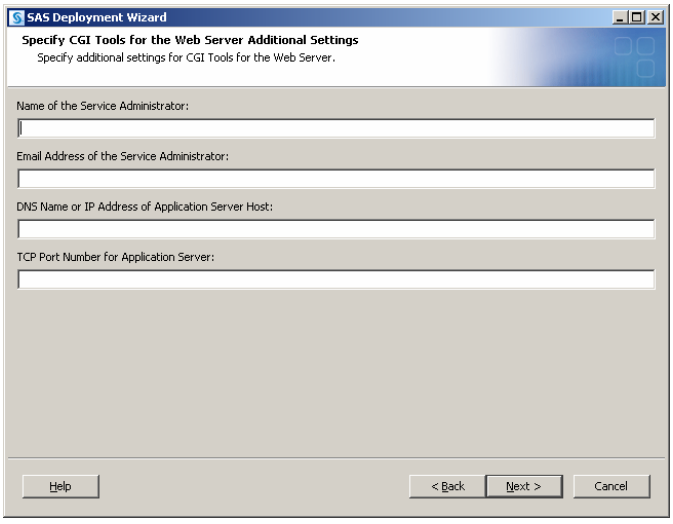
IIS:	`C:\Inetpub\wwwroot\sasweb`
Apache (Windows):	`C:\program files\Apache Software Foundation\Apache2.2\htdocs\sasweb`
Apache (UNIX):	`/usr/local/apache2/htdocs/sasweb`

Physical path for SAS/IntrNet CGI Executables:

IIS:	`C:\Inetpub\scripts`
Apache (Windows):	`C:\program files\Apache Software Foundation\Apache2.2\cgi-bin`
Apache (UNIX):	`/usr/local/apache2/cgi-bin`

URL for CGI Executables:

IIS:	`http://<web_servername>/scripts`
	Example: `http://abcserver.comp.com/scripts`
Apache (Windows):	`http:// <web_servername>/cgi-bin`
	Example: `http://abcserver.comp.com/cgi-bin`
Apache (UNIX):	`http://<web_servername>/cgi-bin`
	Example: `http://abcserver.comp.com/cgi-bin`



Note that your entries for this dialog box are added to the broker.cfg file, a text file that can be edited after the installation has completed.

Name of the Service Administrator: (Optional) Specify the name of the administrator (for example, John Doe).

Email Address of the Service Administrator: (Optional) Specify the email address of the administrator (for example, NetAdmin@comp.com).

DNS Name or IP Address of Application Server Host: Specify the DNS name or IP address of the application server host where SAS Foundation is located.

TCP Port Number for Application Server: The default port is *5001*, but you can use any valid available port on your system in the range 256 – 65535.

Installing CGI Tools and SAS Foundation on Machines with Different Operating Systems

Your SAS 9.4 Foundation operating system might be different than the operating system for CGI Tools. For example, SAS 9.4 Foundation might be installed on a z/OS machine and your CGI Tools might be installed on a UNIX host. The CGI Tools installation from SAS Deployment Wizard detects the destination operating system and install the appropriate operating system-specific software.

There are two methods to make the SAS Software Depot available to the installer on the destination CGI Tools machine. The method you select depends on the facilities available at your site. To access a SAS Software Depot on the destination CGI Tools machine, do one of the following:

1. Launch the setup from a SAS Software Depot residing on a remote system. You might need to use a cross-platform file access method, such as NFS or SAMBA, to connect the two systems.
2. Create media from an existing depot using SAS Deployment Wizard and use it on the host machine. This process is described in detail in the *SAS Deployment Wizard and SAS Deployment Manager 9.4: User's Guide*.

***Note**: SAS/IntrNet operation requires TCP/IP connectivity between the SAS Foundation system and the CGI Tools system regardless of the operating systems where these components are installed.*

Test the Web Server

To determine whether the web server is running, launch the web server's browser and enter `http://localhost`. A web page is returned if the web server is running.

If a web page does not display, you must debug or reinstall your web server before continuing.

Enabling the SAS/GRAPH Java Applets Documentation

If you want to enable online help for the SAS/GRAPH Java Applets, you must modify your web server configuration. Add the following directives to your `httpd.conf` file:

```
AddEncoding     .ascii          8bit          1.0   # stored in ASCII
Pass   /sasweb/graph/doc/*/*.htm <install-path>/sasweb/graph/doc/*/*.htm.ascii
```

Test the Application Broker

To verify that CGI Tools was installed correctly and can access the `broker.cfg` file, point your web browser to the following URL:

Windows:

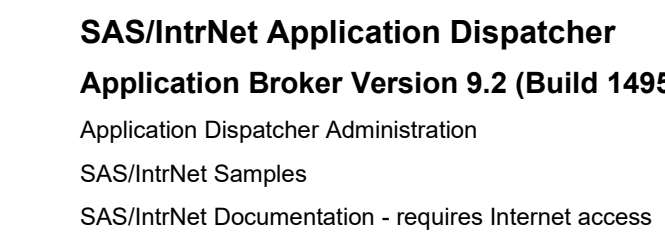
IIS: `http://<web_servername>/scripts/broker.exe`

Apache: `http://<web_servername>/cgi-bin/broker.exe`

Other hosts:

```
http://your_webserver/cgi-bin/broker
```

Replace `your_webserver` with the name of the web server. The URL path might also need to be changed if you installed CGI Tools to a different directory. You should see a web page that resembles the following:



If you do not receive this page, you must debug your web server installation before continuing. Verify that your web server is enabled for CGI execution in the directory where you installed the Application Broker (`broker.exe` and `broker.cfg` files). This directory was determined by what was entered for **Physical path for SAS/IntrNet CGI Executables** during the CGI Tools installation.

Configure a Socket Service

A **default** Application Dispatcher service must be defined to run the sample programs included with SAS/IntrNet. An Application Dispatcher service is a collection of one or more server tasks that will execute SAS/IntrNet programs. The default service described below is a simple server service with one server task that may be used to execute samples and is a convenient way to begin developing your own SAS/IntrNet applications. It is not required, but it is recommended for most installations.

A TCP/IP port (number) or service definition (name) must be reserved for your default service before you create the service. Note that TCP/IP ports or service definitions are not the same as Application Dispatcher service definitions. A single Application Dispatcher service may use one, many, or no TCP/IP ports or service definitions. For the default service definition you will need just one TCP/IP port. Consult your system administrator or check your TCP/IP services definition file (`ETC.SERVICES` and also `/etc/services`) to find an available port.

Services are configured by a batch job found the &prefix.CNTL(INETCFG) data set that was created during the first step of your SAS installation. To create a default service:

1. Edit the parameter file, member INETEDTP in the &prefix.CNTL data set, as described in the following steps. The INETEDTP member contains the parameters necessary for creating a service. Editing instructions are provided in the comments in INETEDTP, and the default values should be changed to the values required for the service being created.
2. Specify the name of the Application Dispatcher service that you are creating. Locate the line containing `ISVC=` and verify the service name is `DEFAULT`.
3. Specify the default service is a socket service. Uncomment the `%SOCKETTYP` line containing `ISVCTYP=`. Make sure the `%POOLTYP` line is commented out by placing an asterisk (*) in the first column.
4. Specify the TCP/IP port number or service name for the server. Change the value `5001` to the correct port number or service name for your server.
5. Review and update any other remaining parameters if necessary.

`ADMINPW=YOURPASSWORD`
Uncomment and supply password to enable administrator password for this service. Do not leave a blank between ADMINPW= and the supplied password. An administrator password is not required and can be added later if you chose to skip this step.

`I$SAMPLE=&prefix.SAMPLE`
Location of SAS/IntrNet sample program PDS.

`I$SAMPSIO=&prefix.SAMPSIO`
Location of SAS/IntrNet sample SAS Library.

`INETENTRY=SAS`
Set to installation default entry.

`INETWORK=500,200`
Primary and secondary allocations for the PROC WORK Data set.

6. Save and close INETEDTP.
7. Edit the INETCFG job to verify the job header information and the name of the service you are defining. The service name in the JCL should match the value you supplied for ISVC in INETEDTP. If you make changes, be sure to save them.
8. Submit the INETCFG job for processing. The INETCFG job will submit another job (INETCFGA). Verify that both jobs complete with a return code of 0. If the jobs complete successfully, the data sets and members necessary for running your service.

   If the INETCFG job fails, examine the messages and SYSPRINT output for error messages. If the following message is received:

   ```
   ERROR: THIS REPLACEMENT CAUSES RESULT TO EXCEED OUTPUT LRECL
   ```

   You might have supplied a pathname in one of the INETEDTP parameters that is too long. Try shortening this pathname and rerun INETCFG.

   ***Note**: Before running INETCFG again, any data sets created by the previous failure of INETCFG must be deleted. To find these data sets, look at the &prefix determined by the original SAS installation.*

   For example, if SAS was installed with the &prefix name SYS.SAS and the failed INETCFG was trying to create the DEFAULT service, delete all data sets beginning with the name &prefix SYS.SAS.WEB.DEFAULT before running INETCFG again.
9. The configuration utility creates a server root in a partitioned data set (PDS) named `&prefix.WEB.DEFAULT`, where `&prefix` is the data set `&prefix` that was supplied during the SAS installation. The PDS contains any JCL procedures and server startup code required for starting the service.

   Verify that the following members have been created:

APSTRTn
contains the JCL necessary to run the corresponding @APSTXn member as a started task. This member should be moved to a started task library and enable it as started tasks.

@APSTXn
contains the SAS code that invokes the server. The JCL calls this file in the corresponding APSTRTn member. These SAS programs must remain in the PDS where they were created.

In addition to the server root PDS, the configuration utility creates an empty PDS named `&prefix.WEB.DEFAULT.TDIR`. The default service will use a member of this PDS named TBLIB1 as its scratch SAS data library.

10. The permissions for the data sets created above must be modified so the server can write to them as necessary. To modify the permissions, create a special security profile that applies to all the data sets in this service (`&prefix.WEB.DEFAULT.*`). The security system profile should also grant write access to the user ID of the Application Server.

11. **If you installed CGI Tools for the Web Server on your z/OS system, complete this step.** The Application Broker must know about this service so that you can access it. Edit the Application Broker configuration file (named `broker.cfg`) on your web server and customize the default service definition block for your installation. The service definition block for a socket service might look like this:

```
# This service contains one server (port 5001) on yourserv.yyy.com.
SocketService default "Reuse existing session"
   ServiceAdmin "Your Name"
   ServiceAdminMail "yourname@yyy.com"
   Server yourserv.yyy.com
   Port 5001
   FullDuplex   True
```

Starting the Socket Service

As stated previously, the `APSTRTn` file for the default Application Dispatcher service should be moved from the file `&prefix.WEB.DEFAULT` to a started task library and enabled as a started task. To start the default service, issue a START command from the system console.

After initialization, the Application Server will pause. This indicates the server has begun waiting for Application Dispatcher requests from the Application Broker, and everything is functioning properly.

Testing the Socket Service

To make sure that the service was installed and started correctly, take the following steps:

1. In a web browser, navigate to this URL:

   Windows:

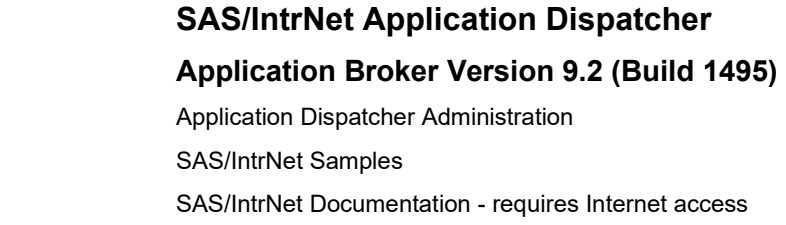
   IIS: `http://web_servername/scripts/broker.exe`

   Apache: `http://web_servername/cgi-bin/broker.exe`

   Other hosts:

   `http://web_servername /cgi-bin/broker`

   Replace *web_servername* with the host name of the web server. The URL path might also need to be changed if you installed CGI Tools to a different directory. You should see a web page *similar to* the following:

2. Click on the **Application Dispatcher Administration** link to see if the Application Broker can read the `broker.cfg` file. The Application Dispatcher Services web page should open.
3. Verify connectivity between the Application Server and the web server. Click on the **Application Dispatcher Administration** link and then click on the **ping** link under **SocketService default** heading. If the ping is successful, you should see:

   ```
   Ping. The Application Server <hostname>:<port_number> is functioning
   properly.
   ```

4. To complete installation testing, type this URL in your browser address line:

   Windows:

   IIS:

   ```
   http://web_servername/scripts/broker.exe?_service=default&_program=sampl
   e.webhello.sas
   ```

   Apache:

   ```
   http://web_servername/cgi-
   bin/broker.exe?_service=default&_program=sample.webhello.sas
   ```

   Other hosts:

   ```
   http://web_servername/cgi-bin/broker?_service=default&_program=
   sample.webhello.sas
   ```

   You should see the string "Hello World!" in large bold type in your browser. If you do not, add the debug option to create a log:

   Windows:

   IIS:

   ```
   http://web_servername/scripts/broker.exe?_service=default&_program=sampl
   e.webhello.sas&_debug=131
   ```

   Apache:

   ```
   http://web_servername/cgi-
   bin/broker.exe?_service=default&_program=sample.webhello.sas&_debug=131
   ```

   Other hosts:

   ```
   http://web_servername/cgi-
   bin/broker?_service=default&_program=sample.webhello.sas&_debug=131
   ```

Save the log screen in the browser for SAS Technical Support.

Configure Additional Services

This chapter only describes how to configure a simple default Application Dispatcher service. There are many reasons you might want to configure additional services, including segregating applications by security or performance requirements and implementing more scalable servers.

See the "Using Services" section of the SAS/IntrNet Application Dispatcher documentation at http://support.sas.com/documentation/onlinedoc/intrnet/index.html for information on configuring additional services, using the Load Manager, and adding pool services.

Chapter 7 – Post-Installation Configuration for Encryption

Important: *Starting with SAS 9.4M8, SAS Foundation servers use the cryptographic libraries provided and installed on the operating system to provide encryption for data at rest and data in motion.*

SAS/SECURE is included with Base SAS and provides strong encryption support. SAS/SECURE provides client components that you can use to create external (non-SAS 9.4) client applications that communicate with a SAS server in a secure environment. To enable encryption between an external client and a SAS server with SAS/SECURE, you must install the SAS/SECURE client components on the client machine.

Starting with SAS 9.4M8, SAS uses IBM System SSL and GSK APIs to provide cryptographic library support for data in motion. Previously, SAS provided OpenSSL libraries for use with z/OS. System SSL is configured separately. See System SSL Encryption on page 100.

Because z/OS uses the System SSL cryptographic libraries, specific SAS-supported cipher suites might not be available. When this issue is encountered, use a cipher suite that is supported by both SAS and System SSL. For more information, see Cryptographic Library Support Starting with SAS 9.4M8.

Encryption for SAS/CONNECT Server and SAS/CONNECT Spawner might also require additional configuration. For more information, see Setting Up TLS for SAS/CONNECT on page 80.

With SAS 9.4M8 and later, Pervasive Encryption is supported for encrypting data at rest. It is configured separately. See Pervasive Encryption on page 98.

For more information about encryption for SAS 9.4 Foundation, see *Encryption in SAS 9.4*.

Changes to System Options

Some SAS system options related to security are no longer supported, starting with SAS 9.4M8. For a full list of system options that were deprecated with SAS 9.4M8 and later, see "SAS System Options Deprecated for Use with IBM z/OS System SSL in SAS 9.4M8" in *Encryption in SAS 9.4*.

See "SAS System Options Used for IBM z/OS System SSL with SAS 9.4M8" in the same guide for a list of options that can be used to provide TLS encryption for z/OS in SAS 9.4M8 and later.

SAS/SECURE Client for Windows

In SAS 9.4M0 - SAS 9.4M7, the secwin.exe executable installs the files that are necessary for the IOM Bridge for COM to use the CryptoAPI algorithms. It also contains a TAR and ZIP file that are used to develop Java clients that use the encryption support.

SAS/SECURE Client for Java

The SAS/SECURE client for Java provides encryption support for Java applications. You can incorporate this support into applications that are written using the following components:

- SAS/SHARE driver for JDBC
- SAS/CONNECT driver for Java
- IOM Bridge for Java

FIPS Support

The Federal Information Processing Systems (FIPS) standard defines the security requirements for cryptographic modules. Starting with SAS 9.4M9, SAS supports FIPS 140-3 on z/OS V3R2 when TLS 1.3 is enabled.

FIPS support is provided as part of IBM System SSL. SAS uses IBM System SSL for encryption of z/OS data in motion. FIPS 140-2 is supported on IBM z/OS V3R2 and earlier using System SSL. You must enable FIPS on the IBM z/OS operating system. Then use the `ENCRYPTFIPS` SAS system option to place SAS code into FIPS mode.

`ENCRYPTFIPS` is a SAS system option that can be added to SAS/CONNECT spawners, IOM spawners, and Base SAS so that the encryption of communications is compliant with FIPS 140-2 and later. The `ENCRYPTFIPS` option is required for FIPS-compliant encryption. In addition, the `NETENCRALG` SAS system option must be set to `SSL`.

Note: *Starting with SAS 9.4M9, SAS has deprecated the* `NETENCRYPTALGORITHM` *(*`NETENCRALG`*) system option values* `AES`*,* `DES`*,* `RC2`*,* `RC4`*, and* `TripleDES`*. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to KB0041538. SAS strongly recommends changing the value to* `SSL` *to specify the use of the TLS protocol.*

For more information about FIPS and configuration to enable encryption in general, refer to *Encryption in SAS 9.4*.

Enabling TLS

Starting with SAS 9.4M8, TLS 1.2 is supported for SAS 9.4 Foundation on z/OS using IBM System SSL. And starting with SAS 9.4M9, TLS 1.3 is supported.

If TLS 1.3 is available and configured on the system, the `SSLMINPROTOCOL=` system option now uses TLS 1.3 if TLS 1.3 is set as the minimum protocol. The SAS `NETENCRYPTALGORITHM` system option must be set to `SSL`. SAS attempts to use TLS 1.3 and falls back to TLS 1.2 if support for TLS 1.3 is not available on the system.

Some additional configuration is required. For more information, see the following sections of *Encryption in SAS 9.4*:

- Configure IBM System SSL to Support TLS 1.3 for z/OS (Starting with SAS 9.4M9)
- Configure System SSL to Support TLS 1.2 for z/OS (Starting with SAS 9.4M8)

Pervasive Encryption

Starting with SAS 9.4M8, support for the IBM Z Pervasive Encryption feature is available. This support is not unlimited: it applies to SAS direct access bound libraries (via EXCP, using sequential basic and sequential large format data sets) and sequential access libraries (such as TAPE), using both the older CCW-based channel programs and the newer zHPF channel programs. Pervasive Encryption support is available only for the 31-bit z/OS SAS Foundation offering, not for the 64-bit offering.

Pervasive Encryption makes it easier to manage encryption keys and passwords for data at rest. Existing encryption methods that use `encrypt=` and `encryptkey=` options encrypt only stored member data and do not encrypt the directory information that is required to manage the library layouts. Pervasive Encryption encrypts 100% of the data in the library. It also enables system administrators to control and track encryption of both SAS and non-SAS stored data sets and libraries using the same approach and the same system tools.

Previously, SAS encryption required users to know and provide keys and passwords to access encrypted SAS data. By contrast, Pervasive Encryption keys are system-protected resources (in RACF, ICSF, and CPACF). Key values are stored in electromagnetically protected hardware. End users are provided with access to the associated label names in order to access the data. Data sets that are protected by Pervasive Encryption are tied to the current host machine. They cannot be transferred by, for example, removing disk drives, without the accompanying keys.

Configuring Pervasive Encryption

Enabling Pervasive Encryption involves multiple steps in the operating system. Refer to the following documents from IBM for descriptions of the necessary tasks to complete:

- Introducing IBM z/OS Data Set Encryption
- What is pervasive encryption for IBM Z?
- IBM Z Pervasive Encryption

Within your SAS code, additional changes are required. Applying pervasive encryption requires you to set one of the following options:

- LIBNAME statement `DSKEYLBL=` option
  The encryption key label can be provided on the `libname` statement using the `DSKEYLBL=<LABEL_NAME>` option. For example, to access an existing library that is already set up for pervasive encryption:

  ```
  libname test BASE ".encrypt.saslib" DISP=MOD, DSKEYLBL=DSKEY1;
  ```

  Here is an example for creating a new encrypted SAS bound library:

  ```
  libname test BASE ".encrypt.saslib" DISP=(NEW,CATLG) UNIT=(RIO,1)
  SPACE=(TRK,(5000,5000)) BLKSIZE=4096 DATACLAS=STD
  MGMTCLAS=STD STORCLAS=STD DSNTYPE=BASIC
  DSKEYLBL=DSKEY1;
  ```

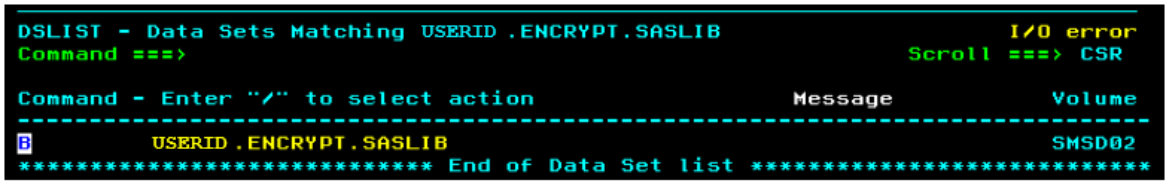
  ***Note**: Pervasive Encryption using EXCP requires the data set to exist on SMS managed volumes.*

- `DLDSKEYLBL=` SAS option
  This is a global SAS option that enables you to set a data set encryption key label, which is provided on every `libname` statement if a `DSKEYLBL=` is not already present. Set this option to apply wholesale encryption on all new SAS libraries with minimal key creation efforts.

For more information, see *Encryption in SAS 9.4*.

Troubleshooting Pervasive Encryption Configuration

Pervasively encrypted data sets have a special bit turned on that verifies that the accessing program can read the file using EXCP. Therefore, if you try to browse or edit the file from a non-capable program, for example using ISPF browse or edit, the system reports an I/O error that resembles the following:



If you attempt to access a SAS library that is Pervasive Encryption-enabled and do not have RACF access to the underlying encryption key label, you will see an error that resembles the following:

```
IEC143I 213-85, IGG19BX, USERID, LOGON, SYS00083,
USERID, TESTCCW.ENCLIB,
RC=X'00000008', RSH=X'00003E84'
ICH408I USER(USERID) GROUP(GROUPID) NAME
KEYNAME CL(CSFKEYS)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ) ACCESS ALLOWED(NONE)
```

System SSL Encryption

Starting with SAS 9.4M8, encryption for data in motion is provided for z/OS using IBM System SSL. IBM System SSL supports FIPS 140-2. It provides all the capabilities of OpenSSL and is fully integrated with the z/OS security services (SAF) that have been in use for many years (such as RACF, ACF/2, and Top-Secret). SAS uses the Global Security Kit (GSK) as the API. The System SSL gskkyman shell program supports the management of certificates that are stored in a key database file or in a z/OS PKCS#12 file. RADCERT can also be used to generate new certificates. IBM provides documentation at the following website: Cryptographic Services System Secure Sockets Layer Programming.

Some configuration steps are required in order to configure a SAS deployment for System SSL. The configuration steps are slightly different for setting up System SSL for z/OS 31-bit and for z/OS 64-bit. Be aware that z/OS 64-bit is supported only for a SAS Metadata Server deployment.

In previous releases, SAS included the Mozilla trusted CA bundle in the trustedcerts.pem file. SAS 9.4M8 and later running in z/OS can use only a key database file (.kdb) or a PKCS#12 file (.p12). PEM files are not supported. The SAS system option `SSLKEYRINGFILE=` supports .kdb and .p12 files. For more information, see Configuration Examples for Accessing System SSL Certificates on page 101.

Prior to SAS 9.4M8, you could use SAS Deployment Manager to add or remove TLS certificates from the truststore. SAS Deployment Manager cannot be used to add or remove certificates for SAS 9.4M8 (or later) clients and servers running in z/OS. However, all SAS Java applications, such as SAS Management Console, still expect to find certificates in `SASHOME`, so a trusted certificate must also be installed there (that is, in *`SASHOME`*`/SASSecurityCertificateFramework/1.1/cacerts/`). For more information about using the truststore with z/OS in SAS 9.4M8 and later, see SAS note 69954.

More information about System SSL for SAS 9.4 on 31-bit z/OS is provided in *Encryption in SAS*. The following sections describe the steps to set up the Metadata Server when System SSL is used for encryption on 64-bit z/OS.

For more information about SAS system options for System SSL, see SAS System Options Used for IBM z/OS System SSL with SAS 9.4M8 and Later.

Configuring the SAS Intelligence Platform Server Tier to Use System SSL

SAS servers require additional configuration in order to communicate with the SAS Metadata Server when System SSL is used for encryption on z/OS 64-bit.

System SSL encryption for 64-bit z/OS is provided by hot fixes. Refer to SAS Usage Note 70504: Support for the IBM "System SSL" for z64 for assistance with downloading and deploying the required hot fixes.

When you have added the hot fixes, see Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS in the *SAS 9.4 Security Administration Guide* for a full set of instructions for configuring SAS servers for System SSL on 64-bit z/OS.

SAS recommends consulting SAS System Options Used for IBM z/OS System SSL with SAS 9.4M8 and Later for information about the required system options.

Configure SAS Management Console

Now set up SAS Management Console to use TLS for connections to the Metadata Server. Run SAS Deployment Manager and add your CA certificate(s) to the SAS CA trust store.

When you have completed those steps, SAS Management Console should be ready to communicate with a running Metadata Server through an encrypted connection. Start SAS Management Console and connect to the Metadata Server port as you normally would.

Configuration Examples for Accessing System SSL Certificates

SAS and IBM support multiple methods of communicating with System SSL where the certificates are stored. The SAS option `SSLKEYRINGFILE=` determines which method is in use. The supported methods are as follows:

- A path name to a key data base (KDB file) that was generated by (for example) the gskkyman utility.
  If a key data base is used, you must provide the password. You can configure the SAS option `SSLKEYRINGSTASHFILE=` with the path to the stash file (STH file) created by gskkyman, which is the preferred method. As an alternative, you can provide the password itself using the SAS option `SSLKEYRINGPW=`.
- An SAF keyring name.
  The syntax is `SSLKEYRINGFILE=USERID/KEYRINGNAME`. If `"USERID/"` is not provided, the user ID of the running process is assumed.
- A z/OS PKCS#11 token name.
  This parameter must be specified as `"*TOKEN*/token_name"`.
- A PKCS#12 file.

Here is an example configuration using a key database (KDB file):

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="<path-to-KDB-file>.kdb"
SSLKEYRINGSTASHFILE="<path-to-STH-file>.sth" or
SSKLEYRINGPW="<KDB-password>"
SSLKEYRINGLABEL="<label-name-in-KDB>"
```

Here is an example configuration using an SAF KEYRING name:

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="<SAF-keyring-name>" or
SSLKEYRINGFILE="<user-ID/SAF-keyring-name>"
SSLKEYRINGLABLE="<label-name-from-keyring>"
```

Here is an example using a PKCS#11 token:

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="*TOKEN*/<token-name>"
SSLKEYRINGLABEL="label-name-from-token"
```

Here is an example using a PKCS#12 file (.P12):

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="<path-to-P12-file>.p12"
SSKLEYRINGPW="<KDB-password>"
SSLKEYRINGLABEL="<label-name-in-P12>"
```

Chapter 8 – Post-Installation Configuration for Workspace Server for Jobs with Large Utility File Requirements

The default configuration of your workspace server might not allocate sufficient utility space for jobs that use large temporary files. These jobs might include, for example, sorts, summarizations, creating cubes, and more.

For details on increasing your workspace server's utility file space, see the "SAS system option UTILLOC" section of the *SAS 9.4 Companion for z/OS*.

Chapter 9 – Post-Installation Configuration for Security

Security for SAS 9.4 Foundation running on z/OS is handled differently than SAS security for other operating systems. Some notable changes to security apply to deployments of SAS 9.4M8 and later. For information about changes to encryption requirements and configuration, see Chapter 7 – Post-Installation Configuration for Encryption.

SAS Security in a z/OS Environment

SAS 9.4 Foundation security for a deployment on z/OS is largely handled by IBM packages such as RACF. However, network security for SAS client and server communications must also be considered. Here are some examples:

- SAS/SHARE enables local and remote clients in a heterogeneous network to update SAS data concurrently. It also provides a low-overhead method for multiple remote clients to read local SAS data. For more information about SAS/SHARE, see Chapter 10 – Post-Installation Configuration for SAS/SHARE on page 105.
- SAS/CONNECT enables cooperative and distributed processing between z/OS and Windows, and between z/OS and UNIX. It supports TCP/IP and XMS (Cross-Memory Services) communications methods. TCP/IP and XMS enable local clients that are running SAS to communicate with SAS applications or programs that are running in remote environments. For more information, see Chapter 5 – Post-Installation Configuration for SAS/CONNECT on page 79.
- The SAS/CONNECT Spawner for z/OS, CNTSPAWN, allows any supported hosts (including Windows) with the machine name and port to connect to z/OS. CNTSPAWN is an executable that starts a spawner on any supported host. It allows any other hosts to send communications through it by specifying machine name, port, and credentials to start SAS server sessions on the host where the spawner is running.
- The SAS z/OS Object Spawner runs on a z/OS server that SAS/CONNECT clients can access and starts a SAS server session in response to a request from a client. The client uses TCP/IP to communicate first with the spawner, and then with the Object server.

  For more information, see Configuring and Starting the Object Spawner on z/OS.

SAS strongly recommends configuring TLS for network communications. With SAS 9.4M8 and later, IBM System SSL libraries are used for encrypting data in motion. See System SSL Encryption on page 100 for more information.

Deprecated Software

Prior to SAS 9.4M8, SAS/SESSION software provided an option that would enable terminal users who were connected to the Customer Information Control System (CICS) to communicate with SAS 9.4 Foundation in a z/OS environment. The user communicated with SAS 9.4 Foundation running in an APPC/MVS initiator, and SAS 9.4 Foundation used VTAM as the communication access method.

Starting with SAS 9.4M8, SAS/SESSION is not available from SAS. If you have an existing installation of SAS/SESSION in your environment and plan to upgrade or migrate to SAS 9.4M8 or later, SAS recommends that you first uninstall SAS/SESSION.

Multifactor Authentication

Starting with SAS 9.4M9, multifactor authentication (MFA) is supported for SAS deployments. MFA is supported for z/OS using IBM Z Multi-Factor Authentication (z/MFA). For the initial release of SAS 9.4M9, you can apply it to the SAS/CONNECT Spawner for z/OS, the SAS z/OS Object Spawner, and SAS/SHARE on z/OS.

For information about and procedures for configuring z/MFA for a SAS deployment on z/OS, see About IBM Z Multifactor Authentication (MFA) in the *SAS 9.4 Security Administration Guide*.

After IBM z/MFA has been enabled, minor changes to your SAS programs are required. For user sign-on with MFA, where you currently provide a user ID and password (to a SAS/CONNECT session, for example), in place of the password, you provide the MFA token.

Chapter 10 – Post-Installation Configuration for SAS/SHARE

Note: *For information on implementing and using SAS/SHARE software, refer to the SAS/CONNECT 9.4 User's Guide and the SAS/SHARE 9.4 User's Guide.*

Special Files for Use with SAS/SHARE Software

Customizing the Started Task JCL Procedure for a Server

STEP 1: Edit the **SHREDITP** member of the **CNTL** data set and specify the following parameter values.

- SASSNM= Supply your started task procedure name. This JCL procedure name and member name will be created in the procedure library, `&prefix.PROCLIB`
- SASSTP= Supply your stop task procedure name. This JCL procedure name and member name will be created in the procedure library, `&prefix.PROCLIB`
- SERVERID= Supply your default server ID
- PROC-DSN= Supply your PROCLIB

STEP 2: Edit and submit the SHRPOST job.

This job modifies the SAS/SHARE started task JCL procedure, member SHRPROC01 in the CNTL data set, with the SHREDITP parameter values from Step 1 above (SASSNM= and SASSTP=) and copies the procedure to the library specified with PROC-DSN.

STEP 3: Notify the server administrator that this file has been provided.

STEP 4: Refer to the *SAS/Share 9.4 User's Guide*, Appendix 2, "Creating the SAS/Share Server Environment," and Appendix 3, "Tuning Tips for Applications That Use SAS/Share Software."

Configuration File for a Server

Member SRVCNFG of the CNTL data set is provided as a default configuration file for a server's SAS execution. This member contains recommended SAS system option settings and is included in the CONFIG concatenation in the started task JCL procedure customized according to the procedure described in the previous section.

Customizing the SAS/SHARE Autocall Macros

Note: *This task is required.*

The installed SASSAML data set is an APPLSYS macro library used by the SAS/SHARE autocall macros. This library contains the required members DEFAULTS and SERVERID. These members contain instructions with examples in comment headers for adding entries to the tables used by the macros. To use this library, you must specify its name in the SHRMACS autocall macro.

STEP 1: Edit the SHRMACS autocall macro (required).

Member SHRMACS in the AUTOLIB data set must contain the correct data set reference for the APPLSYS macro library. Edit this member to change the name SAS.SASSAML to the correct installed &prefix.SASSAML data set name for your installation.

STEP 2: Notify the server administrator that this file has been provided.

Selecting Communications Access Methods to Use

***Note**: This task is required.*

STEP 1: Determine the access method to use.

Communication between a SAS/SHARE server and user is handled by a part of SAS called a communications access method. There are two communications access methods available for use with this release of SAS/SHARE software under z/OS:

- cross-memory services
- TCP/IP

To use the cross-memory services access method, a server and user must be running on the same z/OS system.

You can choose to use one access method exclusively, or you can choose one as your primary access method and others as secondary access methods. If you choose to define primary and secondary access methods, SAS/SHARE software will attempt to establish a user-to-server connection using the primary access method first. If that attempt fails, SAS/SHARE software will then attempt to establish the connection using each of the secondary access methods in turn.

You should choose the access methods you will use based on your site's requirements and restrictions. The cross-memory access method is the default and is faster than the other access methods. However, the cross-memory access method can only be used for within-system communication and requires installation of a module in an authorized link list library and definition of an inactive z/OS subsystem.

***Note**: SAS/SHARE 9.4 is not compatible with the V8 cross-memory module (normally SASVXMS). The V9 module is backward compatible with V8 Share so it can service both releases.*

STEP 2: Set SAS system options to specify selected access methods.

SAS 9.4 Foundation option `COMAMID=` specifies which access method SAS/SHARE software should use as the primary or only access method. SAS 9.4 Foundation option `COMAUX1=` specifies the secondary access method. These options are specified, typically in a SAS 9.4 Foundation configuration file, by the SAS/SHARE Software Consultant.

The following table shows the value of these options for each access method:

Access Method	COMAMID=/COMAUX1= Value
`cross-memory services`	XMS
`TCP/IP`	TCP

For a server, these two options have a related meaning; each access method specified by these options will be initialized when the server is started, making the server accessible to users via any of those access methods.

For example, for a server that is to be accessible only to users who use the cross-memory services access method, specify

```
COMAMID=XMS
COMAUX1=
```

For a server that is to be accessible to users who use either the cross-memory services access method or the TCP access method, specify

```
COMAMID=XMS
COMAUX1=TCP
```

Or

```
COMAMID=TCP
COMAUX1=XMS
```

For a user session, the access method specified by the `COMAMID=` option is the first one used to attempt to connect to a server.

Note that is not necessary to specify `COMAUX1=` if you do not want to specify a secondary access method.

To cause a user session to try the cross-memory services and TCP/IP access methods, in that order, specify

```
COMAMID=XMS
COMAUX1=TCP
```

System Configuration for the Cross-Memory Access Method

Installing the SASVXMS Load Module

Note: *This task is required.*

To use the cross-memory access method for communication between a SAS/SHARE server and user, you must copy the module `SASVXMS0` from the SAS load library data set into an authorized library. You must then rename this module `SASVXMS` (removing the 0). It is very important that you perform these two tasks in that order.

When SAS/SHARE loads the module `SASVXMS`, it must find that module to be marked authorized, re-entrant, and reusable, and to have been loaded from an authorized library.

The version of SASVXMS that was distributed with release 6 of SAS/SHARE can be used ONLY with version 6. If you still have version 6 of SAS/SHARE installed, be sure to follow the special instructions in Step 2.

STEP 1: Copy SASVXMS0 into an authorized link list library.

Copy the module `SASVXMS0` into any authorized library. In a production environment, SAS recommends you copy the `SASVXMS0` module into an authorized link list library. As an alternative, you can install this module into the link pack area. You can use any standard utility program to copy the module `SASVXMS0` from your `&prefix.LIBRARY` data set to your authorized library.

Note: *A user abend 984 will occur if the* `SASVXMS` *module is not installed in an authorized library or the library is in a STEPLIB concatenation where one of the libraries is not authorized.*

STEP 2: Rename SASVXMS0.

After copying SASVXMS0 into the appropriate library, you must rename it. You can use any standard utility to rename the module.

Rename SASVXMS0 to SASVXMS. Specify SAS 9.4 Foundation option `COMAMID=XMS`.

Note: *The XMS access method does not support communication between a Version 6 SAS session and a SAS/SHARE 9.4 server, nor does it support communication between a SAS 9.4 Foundation session and a Version 6 SAS/SHARE server.*

*If you want to run SAS/SHARE 9.4 and Version 6 SAS/SHARE software concurrently, you **MUST** rename the Version 6 SASVXMS0 module to SASVXMSn and set COMAMID=XMSn in Version 6.*

Defining an Anchor Point

Note*: This task is required.*

To use the default cross-memory access method for communication between a SAS/SHARE server and user, you must define an anchor point. The anchor point is a place in common memory that can be located by servers and users and used to store and retrieve cross-memory communication information.

Note: *If you have defined an anchor point for a previous release of SAS/SHARE, it is not necessary to repeat this step now.*

STEP 1: Define an inactive z/OS subsystem.

The anchor point is specified by defining an inactive z/OS subsystem. Defining an inactive subsystem causes z/OS to create a subsystem communications vector table (`SSCVT`) at IPL time. The `SSCVT` chain is in common memory and easily accessible to the cross-memory access method routines. The `SSCTSUSE` field of the `SSCVT` is available to these routines and is used as the anchor point for their control blocks.

Although you define a subsystem to z/OS, it will never be considered active and will provide no system services because the `SSCTSSVT` field of the `SSCVT` will never be non-zero.

You can define the inactive subsystem by adding an entry to any of the following:

- the IEFJSSNT member of SYS1.LINKLIB
- an IEFSSNxx member of SYS1.PARMLIB.

Consult z/OS system initialization and tuning documentation for the details of each alternative.

Regardless of the method you select, you must include the subsystem name, and you must **not** specify an initialization routine name. Use the name `SAS0` unless it conflicts with conventions at your site.

STEP 2: Set SAS 9.4 Foundation option SUBSYSID= to specify the inactive subsystem you defined.

The name you specify for the inactive subsystem defined as the anchor point for the cross-memory access method must also be specified as the value of SAS 9.4 Foundation option `SUBSYSID=`. This option is specified, typically in a SAS 9.4 Foundation configuration file, by the SAS/SHARE Software Consultant. This option is described in *SAS/SHARE 9.4 User's Guide*.

System Configuration for TCP/IP

Note: *The TCP communications access method within SAS/SHARE requires that you configure SAS 9.4 Foundation for communication with TCP/IP. For this reason, ensure that you have reviewed and completed the steps in "System Configuration for Using SAS with TCP/IP" on page 1818.*

Specify SAS 9.4 Foundation Option TCPSEC=_SECURE_ for the Server Execution

Note: *This task is required.*

Specify the SAS 9.4 option TCPSEC=_SECURE_ in `&prefix.CNTL(SRVCNFG)`, described in "Configuration File for a Server" on page 105. This will cause the TCP/IP access method to require users to supply a valid user ID and password for the z/OS system where the server is running in order to connect to the server.

Testing the SAS/SHARE Server Startup

After bringing up your SAS/SHARE server, from a client SAS session execute the following code:

```
PROC  OPERATE SERVER=server-ID SAPW=oapw;

      DISPLAY LIBRARY _ALL_  ;
      DISPLAY USER _ALL_ ;
      DISPLAY SERVER  server-ID ;

RUN;
```

Note: *The server-ID must be identical to the service name that is specified in the SERVICES file. For details, see "The Services File" on page 2929.*

Note: *The operator password oapw is provided at invocation of the SAS/SHARE server. If `OAPW=` was not provided at invocation of the SAS/SHARE server, remove the "`SAPW=`" option from the example.*

Client-Side Components

SAS/SHARE software includes client components that are used outside of your SAS installation. These components are available from the SAS 9.4 Software Download site and are described below:

SAS/SHARE Data Provider

The SAS/SHARE data provider enables you to access, update, and manipulate SAS data using OLE DB- and ADO-compliant applications on Windows platforms.

SAS ODBC Driver

The SAS ODBC driver enables you to access, update, and manipulate SAS data from ODBC-compliant applications on Windows platforms.

SAS/SHARE Driver for JDBC

The SAS/SHARE driver for JDBC enables you to write applets, applications, and servlets that access and update SAS data. The Java Tools package that includes the SAS/SHARE driver for JDBC also includes the SAS/CONNECT driver for Java. If you are writing Java programs using these interfaces, you might also want to use the tunnel feature. This optional feature can be used with the Java applets you write to solve some common configuration problems.

SAS/SHARE SQL Library for C

The SAS SQL Library for C provides an application programming interface (API) that enables your applications to send SQL queries and statements through a SAS/SHARE server to data on remote hosts.

Special Consideration for the SECPROFILE System Option

There are two installation requirements that must be met before the SECPROFILE system option can be used with the TCP access method:

The SAS SVC routine must be installed, and must be at Level 8 or greater. See "Installing the SAS 9.4 Foundation SVC Routine" on page 3434 for details on the SAS SVC.

Note: *These SVC routine Levels shipped with the following SAS releases:*

- *SAS 9.4 – SVC Level 10*
- *SAS 9.3 – SVC Level 9*
- *SAS 9.1.3 and SAS 9.2 – SVC Level 8*
- *SAS 8 – SVC Level 7*

The RACF security administrator must activate the PTKTDATA class, and define at least one PTKTDATA profile for use by SAS/SHARE. If the client and server are on different systems, these steps must be done on both systems, and the profile definitions must be identical on both systems.

Note: *Provided that the Level 8 SVC Routine is installed (on both client and server systems, if they are different), SAS 9.4 Foundation clients can use the SECPROFILE option to connect to SAS Version 6, 7, or 8 SAS/SHARE servers without a password. One restriction applies: the SECPROFILE option does not exist in SAS versions prior to SAS 9.4 Foundation, and as a result, the PTKTDATA profile name can only be the RACF default name. For MVS batch jobs, this is typically the characters "MVS" prefixed to the SMF system identifier of the MVS system. However, it can be changed by an ICHRIX01exit.*

Chapter 11 – Post-Installation Configuration for SAS/TOOLKIT

The SAS/TOOLKIT installation CNTL data set for each language contains sample JCL members that include in-stream cataloged procedures. To make it easier for your users to take advantage of the SAS/TOOLKIT product, you can customize these cataloged procedures for your site and install them in your standard cataloged procedure library.

The member APROC in the &prefix.TLKT.ASM.CNTL library contains the definitions of the cataloged procedures UWASM and UWLINKA. These are needed by SAS/TOOLKIT assembler users.

The member CPROC in the &prefix.TLKT.C.CNTL library contains the definitions of the cataloged procedures UWC, UWCLINK, and UWLINKC. These are needed by SAS/TOOLKIT SAS/C users.

In all cases, the procedures are defined as in-stream procedures. You must remove the PEND line as you add each member to the cataloged procedure library at your site. Also, follow the notes in the cataloged procedure text to see where you need to change data set names, entry points, etc. Any lowercase data set names should be changed to the correct data set names for your installation.

Note: *Some early installations of SAS 9.4 may see data set names beginning with &prefix.TOOLKIT (as opposed to &prefix.TLKT.) These &prefix.TOOLKIT names may be used as is and APROC and CPROC will use the longer names.*

SAS/TOOLKIT software is installed on a z/OS along with multiple libraries. Here is the list as they appear when installed at SAS. Your names will differ except for the last two levels, which should be the same:

```
TDI.MVA.V940.TLKT.ASM.CNTL
TDI.MVA.V940.TLKT.ASM.LOAD
TDI.MVA.V940.TLKT.ASM.MACLIB
TDI.MVA.V940.TLKT.ASM.OBJ
TDI.MVA.V940.TLKT.ASM.SRC

TDI.MVA.V940.TLKT.C.CNTL
TDI.MVA.V940.TLKT.C.LOAD
TDI.MVA.V940.TLKT.C.MACLIB
TDI.MVA.V940.TLKT.C.OBJ
TDI.MVA.V940.TLKT.C.SRC

TDI.MVA.V940.TLKT.PLI.CNTL
TDI.MVA.V940.TLKT.PLI.LOAD
TDI.MVA.V940.TLKT.PLI.MACLIB
TDI.MVA.V940.TLKT.PLI.OBJ
TDI.MVA.V940.TLKT.PLI.SRC

TDI.MVA.V940.TLKT.GLOBAL.BRG
TDI.MVA.V940.TLKT.GLOBAL.GRM
TDI.MVA.V940.TLKT.GLOBAL.OBJ
TDI.MVA.V940.TLKT.GLOBAL.TESTS
```

The first three sets of libraries are for assembly language, the SAS/C compiler, and the PL/I compiler, respectively. A description of the contents of each library follows:

- CNTL contains sample JCL for building the examples in each language.
- SRC contains the sample source code in the proper language.
- OBJ contains the resultant objects from compiling the source.
- MACLIB contains the libraries that are accessed via the `%INCLUDE statement` (PL/I), `#include statement` (C), or used as macros (ASM).
- LOAD contains the resultant load modules.

The GLOBAL libraries contain the parts that are not language-sensitive. A description of the contents of each library follows:

- BRG contains a load module library of bridge routines that are linked into any SAS/TOOLKIT application.
- GRM contains the grammar definitions used by SAS/TOOLKIT procedures.
- OBJ contains the global objects that are linked into any SAS/TOOLKIT application.
- TESTS contains the test programs used to confirm that the samples work properly.

The SAS/C compiler supported in SAS 9.4 is SAS/C 7.50x. The PL/I compiler supported in SAS 9.4 is the IBM Enterprise PL/I compiler for z/OS Version 3.

Chapter 12 – Post-Installation Configuration for SAS System Forms

As the SAS Consultant for your site, you have the ability to customize all SAS forms for your operating system. Associated with each form is a list of available printers. Review this information if users at your site require a site customized print form for use in windowing environments, SAS/FSP, SAS/AF, or SAS/ASSIST.

Note: *You must have SAS/AF Software licensed to modify the site form.*

(Optional) Customizing the Printer Selection List

Whenever you create a SAS 9.4 Foundation form, a list of printers is displayed. You can modify this list to reflect only those printers available for your site. Information on changing the printer selection list has been included in a help file within the `SASHELP` library. To find out more about customizing this printer list, issue the following command from the windowing environment command line:

```
af c=sashelp.base.pdevice.cbt
```

This command displays a series of help screens that provide instructions for adding, deleting, and modifying entries in the PDEVICE Catalog.

Chapter 13 – Post-Installation Configuration for System Services Scheduler

Note: *If you do not have any currently scheduled flows, or if you prefer to re-schedule your existing flows, this section is not necessary.*

In order to address an upcoming change in how IBM stores information about jobs, the SCHGETRC Rexx exec that is used by scheduled jobs now invokes a load module that is also named SCHGETRC. The SCHGETRC load module must be copied (not moved) into a linklist library so that the exec can run it for existing jobs. The JCL template now includes a STEPLIB to the SAS library, so newly scheduled jobs will run it from there.

If you use the Operating System Services Scheduler and have existing scheduled flows that you would like to continue to run (without re-scheduling), you must copy the SCHGETRC load module from the SAS library to a linklist library.

Chapter 14 – Licensing SAS

Starting with SAS 9.4, there are two sets of instructions for updating the SAS license (SETINIT) information, which consists of current data about your software licensing agreement. These instructions provide information to enable your basic or planned deployments.

A few changes to these instructions were introduced in the first quarter of 2025. When it is time to renew your SAS 9.4 license, you receive the set of instructions appropriate for your installation with your SAS Software Order Email (SOE).

These instructions can also be found in the SAS Install Center at the following locations:

- For basic deployments, see https://support.sas.com/documentation/installcenter/en/ikmvsbasicri/66597/PDF/default/setinit_basic.pdf.
- For planned deployments, see https://support.sas.com/documentation/installcenter/en/ikmvsplannedri/66598/PDF/default/setinit_planned.pdf.

If you have additional questions about your software license, visit the License Assistance page.

Chapter 15 – Logging Directly on to the SAS System

z/OS sites can substitute SAS 9.4 Foundation for the standard TSO terminal monitor program. Sites can insulate users from the TSO environment by automatically invoking SAS or a SAS application when users log on. Because SAS is running as its own terminal monitor program, TSO commands are not accessible to users.

This technique is intended for z/OS sites that want to restrict interactive user access to the TSO environment or shield novice users from having to learn how to work in the mainframe environment. Sites that use this technique also save some memory.

This chapter describes how to install and use the direct logon procedure and provides an example. It also discusses the differences between logging on to SAS using the windowing environment, using a windowing application, as well as the possibility of using the direct logon process with SAS/CONNECT software.

In most circumstances, only system administrators need to read this chapter. If you are not a system administrator and are interested in logging directly into SAS, see your SAS Installation Representative.

Installing the Direct Logon Procedure

When users log onto the system, a JCL stream called a logon procedure (logon proc) is automatically executed. Normally, the logon procedure activates the TSO terminal monitor program that sends the TSO READY prompt to the display when the logon process is complete. To make SAS 9.4 the logon environment, replace this procedure with a logon procedure that activates SAS as the terminal monitor program.

To use this capability, you must still start TSO. Any user logging on to SAS must have a valid TSO user ID. However, invoking SAS 9.4 Foundation directly results in a reduction of approximately 50K in working set size for each user.

Complete the following steps to allow users to log directly onto SAS:

- Create a logon procedure that is used by all users directly logging on to SAS.
- Install the logon procedure into your site's logon procedure library.
- Specify the SAS logon procedure as each user's logon procedure.

The logon procedure that you create is similar to the standard SAS cataloged procedure. This procedure is in the PROCxx member of your site's control installation data set (CNTLDSN).

Example Logon Procedure

Example JCL for a logon procedure can be found in `&prefix.PROCLIB`.

After making any site-specific modifications, copy the PROC into a valid site logon procedure library. To enable users to access the SAS direct logon facility, modify their user IDs to use the modified logon procedure.

Using Direct Logon

You can use the direct logon technique to log onto the windowing environment of SAS, or you can choose to log directly onto a windowing application. You can even use this technique in combination with SAS/CONNECT software to log directly onto SAS on the mainframe from your workstation.

The purpose of combining the direct logon technique with SAS/CONNECT software is to prevent users that connect to the mainframe from having access to the TSO environment. For information on using SAS/CONNECT, refer to the *SAS/CONNECT 9.4 User's Guide*.

Logging onto the SAS Display Manager System

Use the example logon previously described. If you need to allocate special files for each user, such as individual SASUSER files, you must create a separate logon procedure for each user because of a system restriction.

Unless you are using the Amdahl Logon Pre-prompt Exit Version 2.7.5, you do not have any control over supplying customized SAS system options, configuration files, or the dynamic allocation of a user's SASUSER data set to a single logon procedure.

Logging on to a Windowing Application

To log directly on to a windowing application, specify an autoexec file for the application. To do this, add a `SASEXEC` statement that supplies the data set name of the file containing the autoexec code to the example logon procedure. The `SASEXEC` statement has the following form:

```
//SASEXEC DD DISP=SHR,DSN=autoexec-file
```

For more information on modifying the way in which SAS is invoked, see Chapter 1, "Initializing and Configuring SAS Software," in the *SAS 9.4 Companion for z/OS*.

Restrictions

Using SAS as the logon environment implies certain restrictions. For example, because SAS is the terminal monitor program, users cannot execute TSO commands or access TSO facilities such as ISPF from their SAS sessions. Nor can users issue the TSO or X command from their SAS sessions to gain access to the TSO environment.

However, the SAS windowing environment contains environment-dependent statements, windows, and a full-function editor that perform many of the same utilities available in ISPF. These services are available to users that log directly onto SAS. Users can dynamically allocate any files they are authorized to access using `LIBNAME` and `FILENAME` statements. They can also use the `INCLUDE` command to include external files and members of partitioned data sets into SAS editor windows. For more information about these and other operating-system-dependent language features, see the *SAS 9.4 National Language Support (NLS) Reference*.

If the logon procedure you have provided includes a `DDname` assigned to the internal reader, users are able to submit batch jobs from within their SAS sessions.

Accounting Considerations

Substituting SAS for the standard IBM terminal monitor program affects records produced by SMF and TSO/MON. SMF type 30, 34, and 35 records have the SAS entry name in the program name field rather than IKJEFT01 or ADFMDF03. Type 32 (TSO command) records are not produced.

If you are using LEGENT Corporation's TSO/MON product, TSO/MON system records contain complete resource usage, transaction, and response time information, but no command information. TSO/MON command detail records are not produced.

SAS is the leader in business analytics software and services, and the largest independent vendor in the business intelligence market. Through innovative solutions, SAS helps customers at more than 70,000 sites improve performance and deliver value by making better decisions faster. Since 1976 SAS has been giving customers around the world THE POWER TO KNOW®.