Configuration Guide for SAS® 9.4 Foundation for UNIX Environments

Copyright Notice

Table of Contents

Chapter 1 – Introduction

Audience

This document is intended for the SAS Installation Representative, designated as the person who is responsible for installing and maintaining SAS software for UNIX systems at your site. It has been updated for SAS 9.4M9 (TS1M9).

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

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 (the exact products vary by customer). Information about the configuration of middle-tier and client-side products is available from other sources, including the SAS Deployment Wizard and any documentation to which it might point you.

The server-side configuration instructions contained in this document are for the configuration of a generic SAS server.

- 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: Application Server Administration Guide* located at `http://support.sas.com/documentation/configuration/index.html`.
- To configure your server as an OLAP Server, refer also to *SAS 9.4 Intelligence Platform: Application Server Administration Guide*, at the same location.
- To configure your server as a Metadata Server, refer to the instructions in the *SAS 9.4 Intelligence Platform: System Administration Guide,* also at the same location.

Conventions Used in This Guide

This document conforms to the following conventions:

`Courier`	Courier type indicates commands, directory paths, file names, menu items, Internet addresses, etc.
Italics	Italic type indicates documentation references or key notes.
Bold	Bold type indicates important text or concepts.
UPPERCASE	Uppercase type indicates variable and option settings.
Dollar sign $ Number sign #	A dollar sign $ or number sign # at the beginning of an example indicates a sample UNIX command line.

Contacting SAS

If you need to contact SAS, refer to the following web page for contact information: `http://www.sas.com/en_us/contact.html`.

Accessing Release Documentation

The latest versions of the release documentation are available from the Install Center web page, http://support.sas.com/installcenter.

Additional Resources

Configuring Hadoop-Related Software

Important: *Starting with SAS 9.4M9, the procedures for deploying SAS/ACCESS Interface to Hadoop and SAS Data Loader for Hadoop have changed. Several manual steps are required in order to configure the drivers and run a configuration script. Full details are available in the SAS 9.4 Hadoop Configuration Guide for Base SAS and SAS/ACCESS.*

With SAS 9.4M9 and later, the default location for data drivers has changed to be `$SASHOME/AccessClients/9.4/DataDrivers` (or, if `$SASHOME` is not defined, in `/usr/local/SAS/AccessClients/9.4/DataDrivers`). As a result, if your program specifies the `CLASSPATH= LIBNAME` option, it must be either that default directory or one of its subdirectories.

Additional changes with SAS 9.4M9 and later affect the locations of the required JAR files. Take the following steps:

1. Create the following directories in the root location:
   - `/data-drivers/hadoop`
   - `/data-drivers/jdbc`

   These directories enable SAS to provide enhanced security for Hadoop deployments.
2. Copy the JAR files that were stored in *`SASConfig`*`/Lev1/HadoopServer/lib/` in previous releases of SAS 9.4 into the new `/data-drivers/hadoop` directory.
3. Copy the HiveJDBC42.jar file into the new `/data-drivers/jdbc` directory.
4. Update the path for the `SAS_HADOOP_JAR_PATH` variable in the sasv9_usermods.cfg file. Here is an example:

   ```
   -SET SAS_HADOOP_JAR_PATH "/data-drivers/hadoop/:/data-
   drivers/hadoop/spark/:/data-drivers/hadoop/hive_warehouse_connector/"
   ```

For information about configuring the SAS Data Loader for Hadoop offering and its associated software order, SAS In-Database Technologies for Hadoop, see "Configuring and Administering SAS In-Database Technologies for Hadoop and SAS Data Loader for Hadoop" on page 71.

Configuring your I/O Subsystem

Adequate resourcing is required in order to ensure that SAS applications achieve the sustained I/O bandwidth that SAS requires for optimal performance. SAS recommends the white paper titled *Best Practices for Configuring your I/O Subsystem for SAS® 9 Applications*, located at http://support.sas.com/resources/papers/proceedings15/SAS1501-2015.pdf.

Troubleshooting System Performance Problems

For a list of papers that are useful for troubleshooting system performance problems, see http://support.sas.com/kb/42/197.html

Chapter 2 – Restricted Options

SAS 9.4 Foundation options can be "restricted" by a site administrator so that once they are set by the administrator; they cannot be changed by a user. An option can be restricted globally, by group, or by user. To restrict an option, it must be added to the appropriate SAS 9.4 Foundation configuration file. The file permissions must be set by the administrator to prevent users from updating it.

SAS Foundation option files are processed in the following order: global, group, and user. If an option is specified in multiple files, the last occurrence is used.

Global Restrictions

Create the file `!SASROOT/misc/rstropts/rsasv9.cfg` and add options to this file in the normal config file format.

Group Restrictions

Create a file of the following format:

`!SASROOT/misc/rstropts/groups/`*`group-name`*`_rsasv9.cfg`

and add options to this file in the normal configuration file format.

Example:
For user `smith` in the group `staff`: the file name would be `staff_rsasv9.cfg`.

User Restrictions

Create a file of the following format:

`!SASROOT/misc/rstropts/users/`*`user-ID`*`_rsasv9.cfg`

and add options to this file in the normal configuration file format.

Example:
For user `smith`, the file name is `smith_rsasv9.cfg`.

Additional Information

To verify that an option has been set correctly, follow this example:

1. Assume that the option `-EMAILSYS=SMTP` was specified in one of the restricted configuration files.
2. Submit the following code:

   ```
   proc options restrict; run;
   ```

   The SAS log should then show a message that resembles the following:

   ```
   Option Value Information For SAS Option EMAILSYS
        Option Value: SMTP
        Option Scope: SAS Session
        How option value set:  Site Administrator Restricted
   ```

The following describes the process when a user attempts to change the option value.

1. Assume that the option -NOTHREADS was specified in one of the restricted configuration files.
2. Submit the following code:

   ```
   options THREADS;
   ```

   The SAS log should then show a message that resembles the following:

   ```
     options THREADS;
               -----
                36
   WARNING 36-12: SAS option THREADS is restricted by your Site
   Administrator and cannot be updated.
   ```

Only one Group Restrictions File is read during SAS processing. The effective groupid of the SAS process that is running is used to determine which Group Restrictions File to use.

- If the effective user ID of the SAS process that is running does not have a corresponding entry in the /etc/passwd file or POSIX attributes in LDAP, only the global restricted option and the group-restricted options files are read.
- If the effective group ID of the SAS process that is running lacks a corresponding entry in the /etc/group file or POSIX group attributes in LDAP, only the global restricted option and the user-restricted options files are read.

By default, the MetadataServer.sh script sets the value for the SAS system option MEMSIZE to MAX. Be aware that setting MEMSIZE as a restricted option might cause the metadata server process to fail with memory-related errors. For more information, see Usage Note 43280: Setting MEMSIZE as a restricted option overrides the MEMSIZE setting used by the metadata server.

Chapter 3 –Configuration for User Authentication and Identification

Overview

UNIX user security is more than just authentication. User identification is also performed when user credentials are validated. Unlike Windows, UNIX uses an integer value, called the UID, to identify users. Ownership of system resources is then assigned by associating a particular UID with a system resource. User identification determines the UID for a particular username.

When user credentials are validated, UNIX systems search a user database for an entry that contains the same username. Traditionally, the user database is a plain file in the file system, but newer security environments may store this in a binary database, or on a server on the network. Most UNIX systems support several other storage methods than the traditional file. Once the user entry is found, the password can be retrieved and matched against an encrypted version of the user-provided password (authentication), and the UID is retrieved for the user (identification).

SAS for UNIX systems validates users in the same manner. The user name provides an index into a user database, from which the user is identified and authenticated. Typically, superuser permissions are required to read the user database. Because running the entire SAS process with superuser permissions is undesirable – users would have access to files they do not own – an external utility, named sasauth (found in `!SASROOT/utilities/bin`), is used by default to perform authentication. The sasauth process runs `setuid` to root, so that it has the appropriate permissions to access the user database.

Authentication Utility for Large Linux Deployments

An additional authentication utility has been added to SAS 9.4M6 and later on Linux: sas-services-daemon. This service is a multi-threaded alternative to sasauth that improves the performance of SAS 9.4 environments under conditions of heavy usage. It is recommended for SAS deployments with a high volume of end-users.

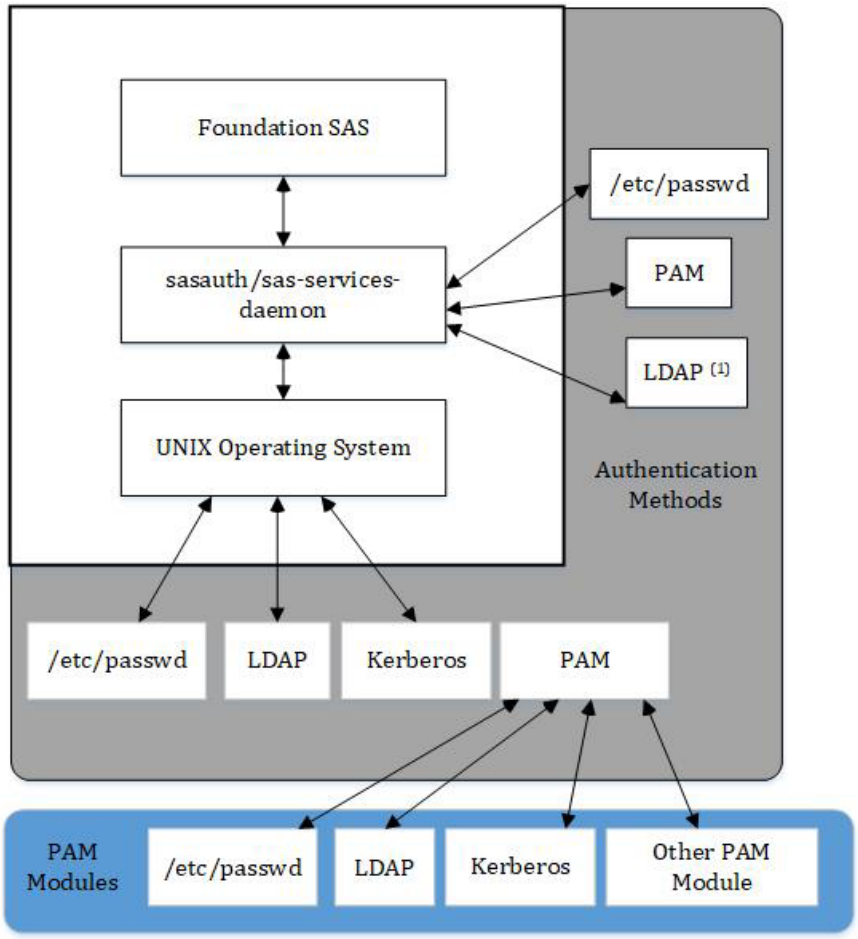
Like sasauth, sas-services-daemon is a software agent that implements UNIX authentication. It is not enabled by default.

Authentication Databases

Authentication databases can be stored in several places. The traditional form is as a text file, `/etc/passwd`, with encrypted passwords stored in `/etc/shadow`. Newer forms use a client/server architecture to provide network-wide authentication, as in NIS+ and LDAP.

For each of these forms, the operating system or application that performs user credential validation must implement the necessary functionality to access the database. Because each form has a different application interface, it is very difficult to support all authentication forms. PAM, or pluggable authentication modules, is a standard library for performing user authentication (but not identification). PAM uses “modules” or libraries, to access multiple authentication forms. A system administrator can select the appropriate authentication based on security requirements. Most UNIX systems support PAM in addition to the native operating system authentication.

The following figure shows possible authentication flows.



SAS strongly recommends configuring the base operating system to use the required authentication or identification form that matches local requirements. For example, if the SAS server is installed at a site with a central LDAP repository, the operating system should be configured as an LDAP client for the central repository.

(1) The sas-services-daemon utility does not directly support LDAP. If LDAP is a requirement, SAS recommends that you use the PAM LDAP module.

Supported Authentication Methods

The sasauth utility supports system authentication (such as `/etc/passwd`), LDAP repositories, and PAM for authentication. Many sites deploy PAM because it is a very flexible and widely accepted authentication mechanism.

The sas-services-daemon utility also supports system authentication (such as `/etc/passwd`) and PAM for authentication. However, sas-services-daemon does not support LDAP unless you use PAM and configure it for pam_ldap. For more information, see `https://linux.die.net/man/5/pam_ldap`.

In a PAM authentication environment, modules can be obtained for custom authentication mechanisms, such as smart cards, and added to the system without direct application support.

But PAM's lack of user identification is problematic for use with sasauth and sas-services-daemon. The PAM programming libraries will only authenticate a user/password combination. The UID, which is needed by SAS, is not returned. Therefore, sasauth and sas-services-daemon use the standard UNIX authentication calls to obtain the UID, meaning that the system must also be configured to access the same user information as PAM.

The following sections describe configuration steps to use sasauth for end-user authentication. For more information about sas-services-daemon, skip to "Running and Configuring sas-services-daemon" on page 18.

Configuring User Authentication Using sasauth

Certain SAS products and features employ functionality that require SAS to check user ID authentication and file access authorizations. This in turn necessitates that selected files within your SAS installation have setuid permissions and be owned by root.

Note: *When you run SAS Deployment Wizard, if you select* ***Run setuid.sh with my ID and password*** *on the "Specify your preference for automatic script execution" dialog box, you do not need to perform the manual steps in this section.*

Configuring user authentication is required for all users of SAS software. You can perform this task by issuing the following commands at the UNIX command prompt.

```
$ su root
# cd !SASROOT/utilities/bin
# mv setuid/* .
# chown root elssrv sasauth sasperm
# chmod 4755 elssrv sasauth sasperm
# exit
```

Configuring sasauth

The sasauth method provides three levels of logging, and user retry lockout, where a user is not allowed to authenticate for a certain period of time after a set number of invalid authentication attempts are made. All of these features are configured in a file, `!SASROOT/utilities/bin/sasauth.conf`.

Note*:* *When you run the SAS Deployment Manager, if you select the* ***Use Pam Authentication*** *check box, then you do not need to perform the manual updates described in this section.*

The sasauth configuration file is a text file consisting of name/value pairs for configuring behavior, one per line. Names and values are case-sensitive. A "#" character is used for comments, which extend to the end of the line.

Supported names and values are listed below.

Name: methods

The methods setting specifies what user validation methods should be used. At least one should be specified, though multiple values may be specified, separated by spaces. Authentication then follows the listed methods in order from left to right; each method is attempted until the user identity is found.

Values	Usage
`pw`	Use system authentication, typically `/etc/passwd` - `/etc/shadow` authentication. On some hosts, this also includes protected password databases or OS-provided enhanced security.
`pam`	Use PAM for authentication. The operating system's user security functions are also used to determine the user's UID and GID. PAM must be configured properly for sasauth, as described in "Configuring PAM Authentication for Use with sasauth" below.
`ldap`	Use LDAP queries for authentication. See "Using the sasauth LDAP Authentication Method" below.
`ext`	Use a custom authentication mechanism. This mechanism is built using the sasauth customization kit, which can be found in `!SASROOT/utilities/src/auth`.

Name: debugLog

Name: accessLog

Name: errorLog

These settings specify the pathnames for the sasauth logs. The sasauth method provides three logs:

- error log – Contains error messages.
- access log – Contains transaction information for each user validation request: user name, validation method used, and validation result.
- debug log – Contains verbose debugging information. Useful when troubleshooting initial configuration.

The value should contain the full path for the log file. Log files whose paths are unspecified are not generated, with the exception of the error log, which is sent to syslog instead.

Log file paths may not include system directories (such as `/dev`, `/usr`, `/etc`). If a log path contains a system directory, sasauth does not create the log and writes a message to the error log or syslog.

For example:

```
#debugLog=
accessLog=/tmp/sasauth.log
#errorLog=
```

configures sasauth to have no debug log, to use `/tmp/sasauth.log` as the access log, and to use syslog for the error log.

Note: *You might need to configure your system's syslog facility to see sasauth messages. Refer to your system documentation for details.*

Name: logOwner

Specifies the numeric UID of the owner of the sasauth log files. Defaults to root because sasauth runs as root. Use this setting to allow a user other than root to read the sasauth log files.

Name: debugNoPasswords

When set to *true*, passwords are not written to the log file. Defaults to `true`.

Name: maxtries

Name: maxtriesPeriod

Name: maxtriesWait

These settings configure sasauth's `maxtries` configuration. The sasauth method does not authenticate a user after a maximum number of attempts are made in a given period of time. The user must then wait for a specified time before additional authentication requests are validated. When `maxtries` is activated, information about failures is logged to the access log. The setting `maxtries` is the maximum number of attempts that can be made. The `maxtriesPeriod` setting specifies the number of seconds after which repeated attempts exceeding the `maxtries` count are not authenticated. The `maxtriesWait` setting specifies the number of seconds the user must wait before the `maxtries` count is reset and validation requests are then permitted.

For example, with the following settings, sasauth will stop authenticating a user for 5 minutes if 5 invalid attempts are made in 1 minute:

```
maxtries=5
maxtriesPeriod=60
maxtriesWait=300
```

To disable `maxtries`, remove all three settings from the configuration file by commenting them out.

Name: DENY_EMPTY_PASSWORDS If TRUE, fail the authentication if the user enters no password. When not set, sasauth follows the UNIX standard of allowing users with no password to authenticate successfully. Defaults to FALSE for backward compatibility with previous versions of sasauth. Used in all sasauth authentication modules.

AIX Only: If using the pw(password) authentication method and AIX_REPORT_RESULT is TRUE, a failed authentication for the user will be reported to the operating system. On AIX systems, use this setting to avoid hangs with LAM and Kerberos due to a bug in the AIX Kerberos module where the Kerberos module assumes that the process is an interactive process (which sasauth is not), and writes to STDOUT.

AIX-specific Options for Password Validation

The following options for AIX and instruct sasauth to use AIX-specific system calls when validating credentials using the "pw" authentication method.

Name: AIX_LOGIN_CHECK

If `TRUE`, check `S_LOGINCHK` flag when authenticating. If unspecified, default value is `TRUE`. The option must be explicitly set to `FALSE` to bypass the check.

Setting this value to true allows system administrators to block access to SAS servers and services for users by changing the value for `S_LOGINCHK`.

Setting this value to `false`, which bypasses the check, allows system administrators to turn off interactive logins but still allow users to utilize SAS servers and services.

Name: AIX_REPORT_RESULT

If `TRUE`, sasauth will report the result of authentications via the "pw" authentication method to the operating system. Defaults to `FALSE`.

Setting this value to true allows SAS authentications to be tracked in the `/etc/security/lastlogin` database. Information for SAS authentications is shown with the TTY name *SAS*.

Name: AIX_USE_AUTHENTICATE

If `TRUE`, sasauth "pw" authentication method uses the AIX system subroutine `authenticate()` to validate user credentials instead of the traditional UNIX algorithm for user validation.

Note: *The AIX* `authenticate()` *routine does not distinguish between an expired password and a bad password. When using the authenticate routine, sasauth never returns "account expired" for expired accounts/passwords. Users always get a more generic "authentication failed" message.*

Name: AIX_USE_LOGINRESTRICTIONS If TRUE, use the AIX subroutine `loginrestrictions()` to determine if a user is allowed to log in or if the user's account has expired. If FALSE, sasauth performs account checks itself. AIX_LOGIN_CHECK is ignored when using `loginrestrictions()`.

AIX_LOGIN_CHECK is ignored when using `loginrestrictions()`

Name: AIX_LOGIN_AUTHMODE Specifies access mode for the `loginrestrictions()` check. Values:

LOCAL - authentication is the same as a local login.

REMOTE - authentication is a remote login, such as for a Telnet session.

The sasauth utility can appear to AIX as a local agent or a remote agent. Some sites configure SAS as a remote agent and then disallow local logins in AIX, prohibiting users from logging into the SAS server directly.

Defaults to "LOCAL".

Name: AIX_AUTH_TTY Specifies the name used by sasauth for TTY when performing authentication and reporting user authentication result. The default value is "SAS". Administrators can use this limit access to SAS via LAM because LAM allows administrators to define access rules based on a TTY name.

Configuring PAM Authentication for Use with sasauth

PAM is architected such that applications must be registered in order to use authentication services. For sasauth to perform authentication, entries must be made in the PAM configuration that describe what authentication services are used when sasauth performs an authentication, specifically the "account" and "auth" module types. (Session and password modules are not supported).

PAM supports applications that run in both 32-bit and 64-bit environments. Modules used with sasauth must match the binary format of the sasauth program. For SAS 9.4 on UNIX platforms, sasauth is a 64-bit binary, and PAM modules must be 64-bit libraries. The standard system modules are usually provided in both 32-bit and 64-bit versions, with each set stored in a separate directory. The `pam.conf` file then contains pathnames that are either relative (Solaris and AIX) or contain a symbolic variable (HP-UX) that allows the correct format to be loaded depending on the format of sasauth.

On HP-UX, Solaris, and AIX systems, the PAM configuration is stored in `/etc/pam.conf`. For sasauth authentication to succeed, entries should be added of the following form:

```
service-name module-type control-flag module-path options
```

For example, these entries enable sasauth to authenticate on Solaris:

```
sasauth auth requisite      pam_authtok_get.so.1
sasauth auth required       pam_dhkeys.so.1
sasauth auth required       pam_unix_auth.so.1
sasauth account required    pam_unix_account.so.1
```

To authenticate on HP-UX:

```
Sasauth account required    /usr/lib/security/$ISA/libpam_unix.so.1
Sasauth auth required       /usr/lib/security/$ISA/libpam_unix.so.1
```

If the system uses an authentication service other than the UNIX password files (such as LDAP or Kerberos), then the entries will have to define what service to use. The manual page for `/etc/pam.d` will help determine these entries.

On Solaris, if LDAP is used, PAM should also be configured to communicate with the directory server via the `ldapclient(1m)` command. Refer to the ldapclient man page for more information.

AIX systems do not ship with PAM activated. For instructions about activating PAM on AIX, refer to the appropriate page on the IBM website:

- AIX 5.3:
  http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=%2Fcom.ibm.aix.security%2Fdoc%2Fsecurity%2Fplugauthmod.htm
- AIX 6.1:
  http://pic.dhe.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.security/doc/security/plugauthmod.htm
- AIX 7.1:
  http://pic.dhe.ibm.com/infocenter/aix/v7r1/topic/com.ibm.aix.security/doc/security/plugauthmod.htm

On Linux systems, the directory `/etc/pam.d` contains a file for each program authorized to use PAM. The name of the configuration matches the name of the process making authentication requests. For sasauth, the configuration file is `/etc/pam.d/sasauth`.

The configuration file contains entries in the following form:

```
module-type control-flag module-path options
```

For example, `/etc/pam.d/sasauth` might contain the following:

```
#%PAM-1.0
auth    required        pam_unix2.so    nullok
account required        pam_acct.so
include password-auth
```

Using the sasauth LDAP Authentication Method

The sasauth LDAP authentication method ("`method=ldap`" in the configuration file) provides a direct connection from sasauth to an LDAP database for authentication. Connections from sasauth to LDAP servers will be encrypted if specified in the sasauth configuration file. The sasauth method queries user attributes from the database and then authenticates the user based on the returned attributes. This method also queries the LDAP database to determine secondary group attributes for the user being authenticated.

LDAP repositories that are used for UNIX authentication (including sasauth) must include UNIX/Posix user attributes (such as UID) in the database. Without this information, the LDAP database cannot be used with UNIX. LDAP databases should conform to the RFC 2307 standard for including UNIX user attributes in an LDAP database, but other schemas can be configured.

The sasauth method requires the following user attributes, listed using their RFC 2307 names:

- `uid` - User name
- `uidnumber` - Numeric UID
- `gidnumber` - Numeric group number of the user's primary group
- `userpassword` - Encrypted form of user's password. sasauth supports crypt, SHA, and SSHA forms.
- `shadowLastChange` - Date of last password change
- `shadowMax` - Maximum age of password before change is required
- `shadowExpire` - Expiration date of account.

Note that password expiration is not processed by sasauth if the password expiration attributes are not found in the database.

The sasauth method also requires the following group attributes, listed using their RFC 2307 names:

- `group` - Group name
- `gidNumber` - Numeric ID of the group
- `memberUid` - User name for an account that is in the group

The `memberUid` attribute is repeated for each member of the group.

Configuring the sasauth LDAP Authentication Method

Once the LDAP method is added to the list of authentication methods for sasauth (see "Name: methods" above), additional settings must be configured for LDAP in `sasauth.conf`. The names and values are listed below.

Name: LDAP_HOST

Name: LDAP_PORT

Name: LDAP_SSL_HOST_PORT

These correspond to host name, port number, and LDAPS port number of the LDAP server. `LDAP_PORT` and `LDAP_SSL_HOST_PORT` can be omitted. In that case, sasauth uses the standard LDAP port number, and it uses `LDAP_SSL_HOST_PORT` instead of `LDAP_PORT` if encrypted communications are active. (See the setting "`LDAP_BIND_SECURITY`" below.)

NAME: LDAP_HOST_LIST

Specifies a list of LDAP hosts to use. Entries in the list are separated by spaces, and are of the form *`hostname:portnumber`*. Port number can be omitted to use the standard port number or standard LDAPS port number. For example:

```
LDAP_HOST_LIST=host1 host2.mycompany.com:3000
```

Hosts are queried from left to right. Hosts are not used if a network connection cannot be made. If a connection is successful, then that host is used for LDAP queries.

Name: LDAP_AUTH_METHOD

Name: LDAP_HOST_DN

Name: LDAP_HOST_PW

Name: LDAP_GROUP_METHOD

The sasauth method authenticates user credentials by using bind or match. For bind, sasauth will bind to the server with the user's credentials. If the bind fails, the user is not authenticated. By binding to the server using the user's credentials, the LDAP server performs all of the authentication (including applying security rules not supported by sasauth), but sasauth cannot determine the specific cause of failed logins. Users will not know why authentication failed when using bind for authentication.

To use bind authentication, set `LDAP_AUTH_METHOD` to the value `BIND` (case-sensitive) in the configuration file.

For match, the user's encrypted password and expiration information are queried from the database and matched with the provided credentials. A mismatch or expiration causes the authentication to fail.

To use match authentication, set `LDAP_AUTH_METHOD` to the value `MATCH`, and set `LDAP_HOST_DN` and `LDAP_HOST_PW` to the user and password for an administrator user. An admin user is required because LDAP will not return the encrypted password to a non-administrative user. The `sasauth.conf` file will now contain password information, so make sure that it is readable only by root (for example, run `chmod 400 sasauth.conf` from the shell).

`LDAP_GROUP_METHOD` controls how sasauth binds when querying secondary group memberships (the LDAP equivalent of reading `/etc/group`) from the LDAP server. When set to `USER`, sasauth binds using the user's credentials. When set to `HOST`, sasauth uses the credentials specified for `LDAP_HOST_DN` and `LDAP_HOST_PW` when binding to the LDAP server. Use the "`HOST`" setting if users lack sufficient access privileges to read group membership information.

Name: LDAP_BIND_SECURITY

Specifies the security/encryption used when binding to the server. Use the value "simple" for standard LDAP authentication. Use the value SSL for encrypted communications using LDAPS. Defaults to *simple*.

When set to SSL, the system administrator must install security certificates and configure sasauth to use them. See "Installing and Configuring LDAPS Certificates" below.

Name: LDAP_SEARCHBASE

Name: LDAP_USERBASE

These settings provide the search criteria used by sasauth when constructing queries to retrieve user identification. For example:

```
LDAP_SEARCHBASE="DC=MYGROUP,DC=MYCOMPANY,DC=COM"
LDAP_USERBASE="ou=People"
```

Set to values appropriate for your organization. Your LDAP administrator can assist with determining these values.

Name: LDAP_USERFILTER

Specifies a filter clause used while authenticating that limits access to SAS servers and services.

For example:

```
LDAP_USERFILTER="(gidNumber=100)"
```

would result in an LDAP query that returns no results for users not in group 100, limiting access to only users in group 100.

Name: LDAP_IGNORE_USERNAME

When set to `TRUE`, sasauth will ignore domain specifications in usernames and pass them to the LDAP server unmodified. When unset, the domain is extracted and an extra OU clause is added containing the domain.

For example:

```
fred@purchasing or purchasing\fred
```

results in:

`msSFU30Name=fred,ou=purchasing,dc=company,dc=com` (option unset)

`msSFU30Name=fred@purchasing,dc=company,dc=com` (option set)

This setting is helpful when working with Active Directory as your LDAP database.

Name: LDAP_SCHEMA

Specifies the schema that the server uses. Select from the following:

- LDAP_SCHEMA=RFC2307 - for RFC 2307
- `LDAP_SCHEMA=AD2` - for Active Directory with Services for UNIX (SFU) 2
- `LDAP_SCHEMA=AD3` - for Active Directory with Services for UNIX (SFU) 3
- `LDAP_SCHEMA=OTHER` - for a manual configuration. Follow instructions in the configuration file when using this value.

Installing and Configuring LDAPS Certificates

When using LDAPS, security certificates must be installed on the system. The sasauth method uses the standard system SSL libraries, so certificates are installed using operating system utilities. General instructions for installing certificates for each UNIX environment are provided below. LDAP servers may require more than one certificate, usually a "root" certificate for your site and a server certificate for the LDAP server itself.

Note: *The following examples use a certificate in binary (.cer) format.*

When installing certificates, order is usually specific. First install your "root" certificate and then additional certificates for the servers LDAP will be accessing.

Solaris and HP-UX Certificates

These hosts use the certutil utility to import certificates. It has the path:

`/usr/sfw/bin/certutil` (Solaris)

`/opt/ldapux/contrib/bin/certutil` (HP-UX)

The certutil utility reads the certificates and adds them to the certificate database. The certificate database is usually located in the following directories:

`/var/ldap` (Solaris)

`/etc/opt/ldapux` (HP-UX)

Your system administrator can place the certificate database in an alternate location, but leaving them in the standard location will make them available to other applications on the system that use the system's version of the LDAP libraries.

Install the certificates as follows. Root permissions are required.

1. Create the certificate directory if it does not exist.
   `mkdir /var/ldap` (Solaris)
   `mkdir /etc/opt/ldapux` (HP-UX)
2. Import the certificates. The `-n certutil` option specifies the name of the certificate, and should match the name encoded inside the certificate.

   ```
   certutil -A -a -i rootcertificate.cer -n "Root CA" -t "CT" -d
   /var/ldap (Solaris)
   certutil -A -a -i server.cer -n "ldapserver" -t "CT" -d
   /var/ldap (Solaris)

   certutil -A -a -i rootcertificate.cer -n "Root CA" -t "CT" -d
   /etc/opt/ldap (HP-UX)
   certutil -A -a -i server.cer -n "ldapserver" -t "CT" -d
   /etc/opt/ldap (HP-UX)
   ```

3. Modify the permissions of the certificates so all users can read them.

   ```
   chmod 644 /var/ldap          (Solaris)
   chmod 644 /var/ldap/*.db

   chmod 644 /etc/opt/ldapux    (HP-UX)
   chmod 644 /etc/opt/ldapux/*.db
   ```

4. Validate the certificates using the list option (-l) of certutil.

```
certutil -L -d /var/ldap     (Solaris)
certutil -L -d /etc/opt/ldapux (HP-UX)
```

Once the certificates are installed, sasauth.conf can be changed to match the installed certificates. Certificate settings are included at the end of the `sasauth.conf` file. Solaris and HP-UX require the following settings:

Name: LDAP_SSL_CERTIFICATE_FILE

Specifies the path/file name for the certificate database. On HP-UX and Solaris, this should specify the directory that contains the certificate files; otherwise, sasauth gets "Bad database" errors when initializing SSL.

Name: LDAP_SSL_STRENGTH

Specifies how the certificates are validated. Select from the following options:

- `LDAP_SSL_STRENGTH=CERT` – Accepts the server's certificate only if certificate authority is trusted.
- `LDAP_SSL_STRENGTH=WEAK` – Accepts the server's certificate without validating certificate authority.
- `LDAP_SSL_STRENGTH=CNCHECK` – Same as `CERT`, but matches the CN attribute with the server's DNS name. With this value set, `LDAP_HOST_LIST` may not be used.

The value "`CERT`" is used in most cases.

AIX Certificates

AIX certificate management tools are provided in the IBM Global Security Kit (GSKit). Use of the kit is documented in section 4.3.1, "Configuring SSL," of the IBM Redbook titled "Integrating AIX into Heterogeneous LDAP Environments." The Redbook is available at `http://www.redbooks.ibm.com/redbooks/pdfs/sg247165.pdf`.

GSKit provides the command "`gsk7cmd`" to create and maintain SSL certificates. (A graphical tool, `gsk7ikm`, can also be used. The examples that follow use the command-line tool.) If the utility is not available on your system, then you need to install the requisite packages as described in the Redbook.

Certificate files (also called key files or the key database) are commonly created in `/etc/security/ldap`, but that directory also contains many other files used by the AIX LDAP client software. Your administrator may want to add use `/etc/security/ldap/keys` instead.

Install the certificates as follows. Root permissions are required.

1. Create the directory if it does not exist.

```
mkdir /etc/security/ldap
```

2. Create a key database. The `-pw` option is the password for the certificate database. Select a password appropriate for your site.

```
gsk7cmd -keydb -create -db /etc/security/ldap/key.kdb -pw
ls93key -type cms
```

3. Import your certificates. The `-label` option is a symbolic name used to identify the certificate in the database. Select a name that uniquely identifies the certificate.

```
gsk7cmd –cert –add –db /etc/security/ldap/key.kdb –pw ls93key
–file rootcertificate.cer –format ascii –label "Root CA"
–trust enable
gsk7cmd –cert –add –db /etc/security/ldap/key.kdb –pw ls93key
–file server.cer –format ascii –label "ldap server" –trust
enable
```

4. Test your database by listing the contents. You should see all of the system certificates and the certificates you just added.

```
gsk7cmd –cert –list CA –db /etc/security/ldap/key.kdb –pw
ls93key
```

5. Validate the permissions for the new certificates. Check the directory that contains the certificates and the files themselves. All users should have read permission for the files and read/execute permissions for the directory.

```
ls –l /etc/security
ls –l /etc/security/ldap
```

Once the certificates are installed, `sasauth.conf` can be changed to match the installed certificates. Certificate settings are at the end of the `sasauth.conf` file. AIX requires the following settings.

Name: LDAP_SSL_CERTIFICATE_FILE

Specifies the path/filename for the certificate database. On AIX, this is the full path for the `key.kdb`, as was used with the `gsk7cmd` utility.

Name: LDAP_SSL_CERTIFICATE_NAME

The name/alias of the certificate to be used when connecting with the LDAP server, usually your root certificate. This should be the name specified in the certificate. You can determine the name of the certificate by executing the command:

```
gsk7cmd –cert –details –db /etc/security/ldap/key.kdb –pw ls93key
–label "Root CA"
```

The name is found in the "Subject:" field of the command output.

Name: LDAP_SSL_CERTIFICATE_PASSWORD

The password for the certificate file, as specified when using the gsk7cmd above. Quotes are not necessary.

Note: *Because the sasauth configuration file now contains a password, check the permissions on the sasauth.conf file. It should only be readable by root.*

Example: sasauth.conf Settings for LDAPS

The following are the necessary settings for sasauth to use encrypted communications with a Sun Directory Server on AIX without including an LDAP bind password. Only members of group 112 will be able to connect.

```
methods=ldap
LDAP_HOST=ldap.company.com
LDAP_AUTH_METHOD=BIND
LDAP_GROUP_METHOD=USER
LDAP_BIND_SECURITY=SSL
```

```
LDAP_SEARCHBASE="DC=group,DC=company,DC=com"
LDAP_USERBASE="ou=People"
LDAP_USERFILTER="(gidNumber=112)"
LDAP_SCHEMA=RFC2307
LDAP_SSL_CERTIFICATE_FILE=/etc/security/ldap/key.kdb
LDAP_SSL_CERTIFICATE_NAME="Root CA"
LDAP_SSL_CERTIFICATE_PASSWORD=ls93key
```

AIX: Using System LDAP Authentication with sasauth

IBM does not provide an LDAP module for PAM. The open source package OpenLDAP can be used to build an LDAP module, but this is not recommended for production environments since it is not a solution supported by IBM. Instead, sites that need LDAP authentication should configure the AIX system for LDAP authentication. Refer to the IBM Redbook *Integrating AIX into Heterogeneous LDAP Environments* for instructions on how to configure AIX as an LDAP client.

Solaris: LDAP and Numeric User Names

The Solaris LDAP client does not treat numeric user names as user names. Instead, Solaris assumes that a user name that is numeric is actually a UID, and converts the user name directly to the UID instead of querying the LDAP database. Since Solaris recommends that user names begin with an alphabetic character, this is unlikely to change. If your site uses Solaris as an LDAP client, then user names in LDAP cannot be numeric.

Customizing Authentication and Identification with sasauth

The sasauth method can be configured to perform authentication in a site-specific manner. The UNIX Authentication Application Programming Interface (API) is a set of predefined routines that provide user authentication, identification, and permissions verification for SAS when running in UNIX environments. The source files provide the ability to add site-specific behavior to the authentication/identification/permissions validations process. Administrators who need to implement custom behaviors should contact SAS Technical Support.

Configuring sas-services-daemon

The sasauth utility runs by default on SAS 9.4M6 and later in UNIX environments. However, you can instead enable the sas-services-daemon utility by setting the `SAS_USE_SASSD=` environment variable to 1 in the level_env_usermods.sh file. Here is an example:

```
export SAS_USE_SASSD=1
```

In addition, if you want to enable sas-services-daemon in a SAS Grid Manager environment, you must set the value of the `SAS_USE_SASSD=` environment variable to 1 in the `Lev`*`N`*`/Grid/sgmg_usermods.sh` script.

***Note**: This additional step applies only to a SAS Grid Manager environment (with SAS 9.4M6 or later). It does not apply to a SAS Grid Manager for Platform environment or to a SAS Grid Manager for Hadoop environment.*

For all SAS grid environments, you must unset (disable) the `SAS_USE_SASSD=` environment variable in the WorkspaceServer_usermods.sh script. Add the following command to WorkspaceServer_usermods.sh to disable this variable:

```
unset SAS_USE_SASSD
```

Configure sas-services-daemon by editing the following file:

```
!SASROOT/utilities/bin/sas-services-daemon.conf
```

This file has the same format and options as sasauth.conf. It must be saved in the same directory as sas-services-daemon.

The following options are unique to sas-services-daemon:

- `-numThreads` – The number of threads to use.
  Default: 4 threads. **Range**: 1 to 32 threads.

  Typically, this should be set to the number of cores on the server. The optimal number might vary from site to site. You might try a test of the system with 20 threads configured.

  Important *Configuring this option properly for your environment is critical for grid deployments.*

- `-socketQueue` – Queue length of sockets waiting to be accepted.

  Default: 100. **Maximum**: 500. If the maximum is exceeded, connections are refused.

You must restart sas-services-daemon in order to apply these configuration changes.

Manual startup instructions are provided in Starting sas-services-daemon on page 20.

Changing the Authentication Method

The default authentication method for SAS 9.4 on UNIX is via the host operating system. Most sites that use host operating system authentication deploy a shadow password file configuration. An administrator can change the authentication method. The file sas-services-daemon.conf is used by sas-services-daemon to manage the authentication requirements. The sas-services-daemon.conf file must be saved in the same directory as sas-services-daemon.

To change the authentication method, change the line that contains `methods=pw` in the sas-services-daemon.conf file. Here are the available methods:

`methods=pw`

Specifies the use of standard `/etc/pwd` **or** `/etc/shadow` authentication.

`methods=pam`

Specifies the use of PAM for authentication. The password database is also used to determine the user's UID and GID.

Configuring PAM Authentication for Use with sas-services-daemon

PAM is architected such that applications must be registered in order to use authentication services. For sas-services-daemon to perform authentication, entries must be made in the PAM configuration that describe what authentication services are used when sas-services-daemon performs an authentication, specifically the "account" and "auth" module types. (Session and password modules are not supported).

On Linux systems, the directory `/etc/pam.d` contains a file for each program authorized to use PAM. The name of the configuration matches the name of the process making authentication requests. For sas-services-daemon, the configuration file is `/etc/pam.d/sas-services-daemon`. The sas-services-daemon is a 64-bit binary, and PAM modules must be 64-bit libraries.

The configuration file contains entries in the following form:

```
module-type control-flag module-path options
```

For example, `/etc/pam.d/sas-services-daemon` might contain the following:

```
#%PAM-1.0
auth required pam_unix2.so nullok
account required pam_acct.so
include password-auth
```

Starting sas-services-daemon

Start sas-services-daemon when the system starts up, before any SAS processes or SAS servers have started up. The daemon must run under the root user ID. Unlike sasauth, sas-services-daemon does not have to be owned by the root user, nor does it require you to set the setuid bit.

On UNIX, you can use "service" (System V) commands or "systemctl" commands to start, stop, or restart it.

SAS has provided example init scripts in the `SASROOT/utilities/bin` directory:

- sas-services-daemon.init for System V
- sas-services-daemon.service for Systemd

Start the Daemon Using System V

Copy sas-service-daemon.init to `/etc/init.d/sas-services-daemon`.

Edit sas-services.daemon.init and modify the following line to reflect the actual installation location for sas-services-daemon:

```
cmd="SASHOME/SASFoundation/9.4/utilities/bin/sas-services-daemon"
```

To start: `sudo /sbin/service sas-services-daemon start`

To stop: `sudo /sbin/service sas-services-daemon stop`

To restart: `sudo /sbin/service sas-services-daemon restart`

To check status: `sudo /sbin/service sas-services-daemon status`

To start on boot: `sudo /sbin/chkconfig --add sas-services-daemon`

Manage the Daemon Using systemd

Copy sas-services-daemon.service to `/etc/systemd/system/sas-services-daemon.service`.

Edit sas-services.daemon.service and modify the following line to reflect the actual installation location for `sas-services-daemon.ExecStart=SASHOME/SASFoundation/9.4/utilities/bin/sas-services-daemon`

After sas-services-daemon.service has been copied to `/etc/systemd/system` and modified, you must reload the systemd manager configuration:

```
sudo systemctl daemon-reload
```

To start the daemon: `sudo systemctl start sas-services-daemon`

To stop it:	`sudo systemctl stop sas-services-daemon`
To restart it:	`sudo systemctl restart sas-services-daemon`
To check status:	`sudo systemctl status sas-services-daemon`
To start it on boot:	`sudo systemctl enable sas-services-daemon`

Chapter 4 – Configuring Integrated Windows Authentication

Integrated Windows Authentication (IWA) configures participating SAS servers to accept users who have successfully authenticated to their Windows desktop. It is primarily used for connections to the Metadata Server and the standard workspace server, but it is also supported for direct connections to an OLAP server (for example, from a data provider).

User name/password authentication takes place using PAM in the default IWA setup; therefore, PAM configuration is required. For more information, see the preceding chapter, "Chapter 3 – Configuration for User Authentication and Identification."

A number of benefits are associated with IWA:

- Bypasses the initial logon prompt.
- Accommodates logon mechanisms that are not password-based (such as smart cards or biometrics).
- No user credentials are transmitted.
- Uses the Kerberos protocol, which relies on exchanging tickets rather than passwords, and this process happens automatically without user knowledge.
- Clients can talk to Windows and UNIX servers (see the limitations listed below).

Some significant limitations should also be considered:

- All participating clients and servers must authenticate against the same Windows domain (or against domains that trust one another).
- Web applications cannot participate in this implementation of IWA. However, if you configure SAS Web Application Server for web authentication using IWA, your web applications can use IWA. SAS provides instructions for configuring IWA for SAS Web Application Server in the *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide*.
- If you use IWA for the Metadata Server, no cached credentials are available from an initial logon. For this reason, SAS recommends configuring IWA for the workspace server as an additional step.
- Desktop clients that run on UNIX (for example, SAS Management Console on UNIX) cannot participate in IWA.

If you want to use IWA on a UNIX platform, additional requirements apply:

- To use IWA on Linux or Solaris, you must purchase, install, and configure additional third-party software. One Identity Authentication Services 4.0.1.23 or later (formerly Quest Authentication Services), Heimdal Kerberos, and MIT Kerberos are supported.
- To use IWA on any other of the supported UNIX platforms, you must purchase, install, and configure One Identity Authentication Services 4.0.1.23 or later.
- With IWA on any UNIX platform, only Kerberos connections are supported (there is no support for NTLM on UNIX).

The use of Integrated Windows Authentication is optional.

Prerequisites for Integrated Windows Authentication on UNIX

To use IWA for a server on a UNIX host, you must complete the following steps:

1. Purchase, install, and configure a supported authentication package.

   Note: *With SAS 9.4M2 and later, the implementation of IWA on UNIX supports One Identity Authentication Services 4.0.1.23 (or later). This product was previously named Quest Authentication Services. The Linux and Solaris platforms also support Heimdal Kerberos and MIT Kerberos. The examples that follow discuss One Identity Authentication Services.*

2. Verify that your UNIX host has joined the Active Directory domain and is represented in Active Directory as a computer object.

3. Create a service account and corresponding keytab file. For example, on your UNIX host, from `/opt/quest/bin/vastool`, run the following command:

   ```
   vastool -u admin service create SAS/
   ```

   Here are some details:

- The command-line utility named vastool enables you to manage your One Identity Authentication Services deployment, information in Active Directory, keys, and Kerberos tickets.
- In the `-u` option, specify an Active Directory identity under which vastool can connect to Active Directory and create users. You will be prompted for the password.
- In this example, `SAS` is the service class name of the service account that is created. You must use this service class name in order to create the default service principle name (SPN) that clients expect.

  Note: The alternative, using a custom SPN, can be labor-intensive and error-prone.
- The command creates a service account in the default Computers container on the Active Directory domain. The account name is in a format such as *machine-service*. In this example, if the UNIX host is **machineA.unx.company.com**, then the service account name is **machineA-SAS**. A random password is generated for the account.
- The service has an associated user principal name (UPN) in the format *account-name@Kerberos-realm*. In this example, the Kerberos realm is **COMPANY.COM**, so the UPN is `machineA-SAS@COMPANY.COM`.
- The service also has an associated service principal name (SPN) in the format *service/machine@Kerberos-realm*. The machine value must be specified as a fully qualified domain name (FQDN). In this example, the SPN is **SAS/machineA.unx.company.com@COMPANY.COM**.
- The service has a corresponding Kerberos keytab file. For each type of encryption, the file includes two entries, one for the UPN and another for the SPN. Each key is derived from the service account's generated password, so for each encryption type the keys for the UPN and the SPN are identical. In our example, the vastool command generates a keytab file named `SAS.keytab` with contents that resemble the following:

```
Type                    Principal                                  Key
aes128-cts-hmac-sha1-96 machineA-SAS@COMPANY.COM                   ca17fd3d8...
aes128-cts-hmac-sha1-96 SAS/machineA.unx.company.com@COMPANY.COM   ca17fd3d8...
aes256-cts-hmac-sha1-96 machineA-SAS@COMPANY.COM                   01562e774...
```

```
aes256-cts-hmac-sha1-96  SAS/machineA.unx.company.com@COMPANY.COM  01562e774...
arcfour-hmac-md5         machineA-SAS@COMPANY.COM                  tht8qrg72...
arcfour-hmac-md5         SAS/machineA.unx.company.com@COMPANY.COM  tht8qrg72...
```

Note: *Not all of the encryption types that are listed in a keytab file are necessarily available or used in all contexts.*

- By default, the keytab file is named *service*`.keytab` and is located in `/etc/opt/quest/vas`. You can specify a different location, for example:

  ```
  vastool -u admin service create -k /etc/mypath/SAS.keytab SAS/
  ```

  For more information, consult the documentation about vastool from One Identity (formerly Quest).

4. Participating SAS processes on UNIX must be able to read the keytab file. In the standard configuration, those processes run under the SAS Installer (sas) account, so it is that UNIX identity that requires access to the keytab file.

CAUTION: *Anyone who can read a keytab file can use all of the keys that it contains. Make sure that the keytab file is not generally available.*

5. Set the Quest shared library path, based on the host you are working with:

   AIX

   Add the following code to the level_env.sh script to set the Quest library path environment variable. Specify locations as appropriate for your environment.

   ```
   SAS_QUEST_PATH="/opt/quest/lib"   # user defines this path for
   their AIX platform
       if [ -z "$LIBPATH" ];
       then
         LIBPATH="$SAS_QUEST_PATH"
       else
         LIBPATH="$LIBPATH:$SAS_QUEST_PATH"
       fi
       export LIBPATH
   ```

   HP-UX for the Itanium Processor Family

   Add the following code to the `level_env.sh` script to set the Quest library path environment variable. Specify locations as appropriate for your environment.

   ```
   SAS_QUEST_PATH="/opt/quest/lib"   # user defines this path for
   their HP-UX platform
       if [ -z "$LD_LIBRARY_PATH" ];
       then
         LD_LIBRARY_PATH="$SAS_QUEST_PATH"
       else
         LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SAS_QUEST_PATH"
       fi
       export LD_LIBRARY_PATH
   ```

   In addition, the path you specified must be added to the file `/etc/dld.sl.conf` to allow for correct dynamic linking of setuid root programs (like sasauth). If the `/etc/dld.sl.conf` file does not exist, it must be created. The file can be readable by all but must only be writeable by root or it will be ignored. See "`man dld.so`" for more information.

Linux for x64

Create a file in `/etc/ld.so.conf.d` called `vas.conf` and add the following line:

```
/opt/quest/lib64
```

Note that the added content is based on the default location for installation and can vary.

Run the `/sbin/ldconfig` to recreate the `/etc/ld.so.cache`. If you are using the sasauth utility, this step is required to enable sasauth to perform Kerberos authentication. It runs as root, and shared libraries must be in a trusted path and cannot be specified with `LD_LIBRARY_PATH`.

Solaris and Solaris for x64

Use the `crle` command to add the Quest library location to the search path of both the default and trusted search paths:

- Solaris: /opt/quest/lib/sparcv9
- Solaris for x64: `/opt/quest/lib/64`

The command would look like this, using Solaris for x64 as an example:

```
crle -64 -c /var/ld/64/ld.config -l
/lib/64:/usr/lib/64:/opt/quest/lib/64 -s
/lib/secure/64:/usr/lib/secure/64:/opt/quest/lib/64
```

After Configuring Your Deployment

1. Add the following lines in /…/Lev1/level_env_usermods.sh. Note that the path depends on where you placed the keytab file:

```
KRB5_KTNAME=/etc/opt/quest/vas/SAS.keytab
export KRB5_KTNAME
```

2. Restart the back-end servers to pick up the new environment variable. At this point, the back-end servers should be ready to accept Kerberos connections.

Optional: If you are using Kerberos-enabled NFSv4, Kerberos tickets must be cached in files according to the following pattern:

`krb5cc_<UID>_<xxxxxx>`

rather than in this format:

`tkt<xxxxxx>`.

To change the default cache name pattern:

1. Add the following lines in `/…/Lev1/ObjectSpawner/ObjectSpawner_usermods.sh` and, if you are using SAS Grid Manager, in `/…/Lev1/Grid/sgmg_usermods.sh`:

```
SASUSEDEFAULTCACHE=1
export SASUSEDEFAULTCACHE
```

2. Restart SAS Object Spawner and SAS Workload Orchestrator Server.

Logins for Users Who Participate in Integrated Windows Authentication

If you decide to configure IWA, make sure that user metadata definitions include logins with properly formatted user IDs. The format of the stored user IDs must match the format in which authenticated user IDs are returned to the target server. Failure to meet this requirement causes the user to have only the generic PUBLIC identity (which, by default, cannot log in to most applications).

In the standard configuration, the appropriate format varies as follows:

- If the target server is on Windows, the authenticated user ID is returned in qualified format, so the stored user ID should be qualified (for example, `WIN\joe` or `fred.smith@company.com`).
- If the target server is on UNIX, the authenticated user ID is returned in short format (it is not qualified), so the stored user ID should not be qualified (for example, `joe` or `fred.smith`).

If you need to align formats, use the `SASUSEKERBNAME` environment variable. For example, you might use this environment variable in either of the following circumstances:

- The metadata server is on Windows, the workspace server is on UNIX, both are using IWA, and you don't want to store two logins for each user.
- You need to distinguish between two different users, in two different Kerberos realms, who happen to have the same sAMAccountName name (for example, joe@US.COMPANY.COM and joe@EMEA.COMPANY.COM).

For more information, see "Windows User ID Formats" in the *SAS 9.4 Intelligence Platform: Security Administration Guide*.

Using Custom Service Principal Names

In the unusual circumstance where you need to use a SPN that differs from the standard, generated SPN, review the following information.

In a standard configuration on Windows, SAS servers automatically register their SPN as **`SAS/`***machine* (for example, **SAS/machineA.na.company.com**). Clients can construct the default SPN (because they know the format and machine name), so you don't have to explicitly provide the SPN.

If you need to use a custom SPN on UNIX, the SPN that is used must be listed in the keytab file. In addition to running setspn to set a custom SPN, and adjusting client connection profiles to use that custom SPN, you must generate a new keytab file that includes the new SPN. See Step 2 in "Prerequisites for Integrated Windows Authentication on UNIX" above.

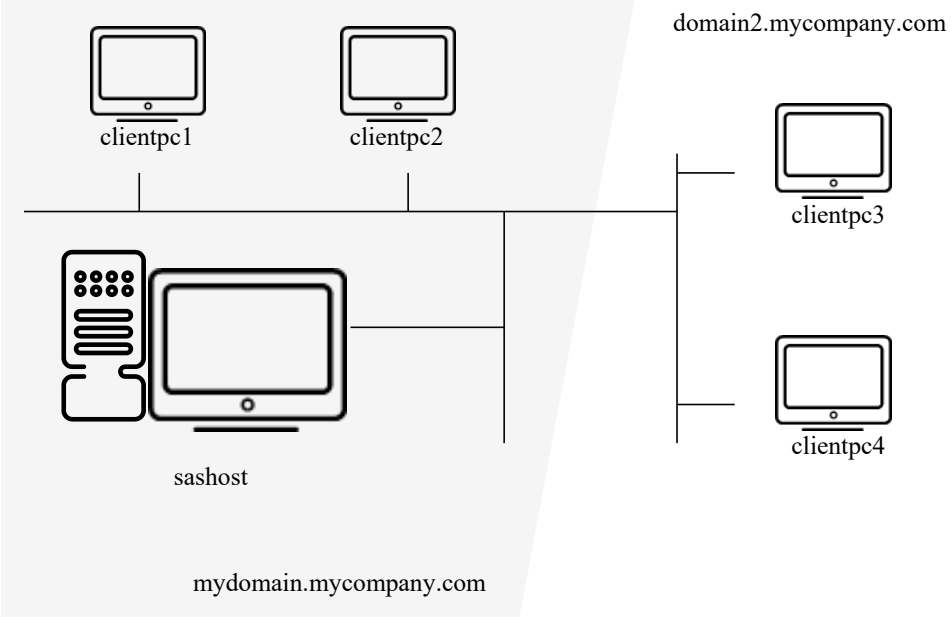
Additional Documentation

For more information about Integrated Windows Authorization, including detailed configuration information about different kinds of servers and recommended security protocols, refer to "How to Configure Integrated Windows Authentication" in the "Authentication Tasks" chapter of the *SAS Intelligence Platform: Security Administration Guide*.

Chapter 5 – Post-Installation Configuration for Remote Browsing

The SAS host might need to be configured appropriately for remote browsing. If one or more SAS desktop clients reside outside the DNS domain of the SAS host, the host must be configured with a hostname that contains the fully qualified domain name (FQDN) of the host.

For example. SAS is installed on the host `sashost.mycompany.com`, two client computers exist in the same domain (`clientpc1.mycompany.com` and `clientpc2.mycompany.com`), and two other clients exist in another domain (`clientpc3.domain2.mycompany.com` and `clientpc4.domain2.mycompany.com`).



If the system named `sas.mycompany.com` is not configured with a host name that is the FQDN for the system, clientpc1 and clientpc2 will be able to view HTML content from SAS, but clientpc3 and client4 will not. This situation occurs because URLs generated for the SAS host will not include the domain, as in `http://sashost:12345/output.html`.

Because clientpc1 and clientpc2 are in the same domain as SAS itself, their browser will build valid host names from their domain, `sashost.mycompany.com`. But clientpc3 and clientpc4, which are outside the domain of the SAS host, will use their domain names to construct a complete host name, which results in the invalid name `sashost.domain2.mycompany.com`.

By configuring the SAS host with the system's FQDN, URLs for HTML display are valid from all clients. From the example, the valid URL for all clients is `http://sashost.mycompany.com:12345/output.html`.

Configuring a Host with a Fully Qualified Domain Name

Note: *Superuser privileges are required to make this change.*

1. Edit `/etc/hosts`.
2. For the IP address of network interfaces for the host, add the FQDN as the first name in the list. For example (using IPv4 addresses), the following entry:

   ```
   10.4.86.62        sashost
   ```

   becomes

   ```
   10.4.86.62        sashost.mycompany.com sashost
   ```

Chapter 6 – Supporting 64 KB pages on AIX Machines

IBM pSeries servers running AIX 5.3 now support 64 KB pages as well as 4 KB pages. To allow SAS executables to take advantage of 64 KB pages, set and export the environment variables using the following commands:

```
$ LDR_CNTRL="DATAPSIZE=64K@TEXTPSIZE=64K@STACKPSIZE=64K@$LDR_CNTRL"
$ export LDR_CNTRL
```

Chapter 7 – Post-Installation Configuration for National Language Support (NLS)

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

Important: Before invoking a localized SAS 9.4 Foundation image from a UNIX shell, you must ensure that the UNIX locale environment variable `LANG` is set appropriately for the language of the SAS version you want to run. The exact values to set will vary depending on your operating system support. In addition, be sure to use the proper font for your language.

To list the locales supported on your operating system, enter the following command:

```
$ locale -a
```

For example, to invoke a Japanese version of SAS 9.4 Foundation in the HP-UX Korn shell environment, enter the following command:

```
$ LANG=ja_JP.SJIS; export LANG
```

For more information on setting locale environment variables, consult the documentation for your operating system. And if you have additional questions, consult the *SAS 9.4: National Language Support Reference Guide*.

Introduction

SAS Invocation Scripts

SAS is invoked by Bourne Shell scripts located in the `!SASROOT/bin` directory. A SAS invocation script is created for each language installed. The invocation scripts are named using the language codes of the installed language. For example, `sas_en` invokes the English version of SAS 9.4 Foundation. Below is a list of the valid languages and language codes.

Language	Code
Arabic	AR
Chinese (Simplified)	ZH
Chinese (Traditional)	ZT [EUCTW/BIG5]*
Danish	DA
Dutch	NL
French	FR
German	DE
Hebrew	IW
Hungarian	HU
Italian	IT
Japanese Primary Encoding	JA
Japanese Secondary Encoding	JA [EUC/SJIS]**
Korean	KO

Norwegian	NO
Polish	PL
Portuguese (Brazil)	PB
Portuguese	PT
Russian	RU
Spanish (Castilian)	ES
Swedish	SV
Turkish	TR

* For Chinese Traditional, EUCTW is the primary encoding on Solaris and the secondary encoding on AIX, HP-UX, and Linux. BIG5 is the primary encoding for AIX, HP-UX, and Linux and the secondary encoding for Solaris.

** EUC is the Japanese Secondary Encoding for HP-UX and AIX. SJIS is the Japanese Secondary Encoding for Solaris, and Linux.

SAS Configuration Files

SAS 9.4 Foundation creates a separate configuration file for each language installed (including English). These language-specific configuration files are `!SASROOT/nls/`*`lang`*`/sasv9.cfg` for each respective language. The configuration file `!SASROOT/sasv9.cfg` is also language-independent. This master configuration file in `!SASROOT` is used by all languages, in addition to the language-specific files in `!SASROOT/nls/`*`lang`*`/`.

In SAS 9.4, the `ENCODING` system option is set explicitly in the configuration file for single-byte languages and Unicode support. In the first three maintenance releases of SAS 9.4, encoding for DBCS was not set in the configuration file.

***Important**: Starting with SAS 9.4M4, the `ENCODING` option for DBCS is also set in the `sasv9.cfg` file. If you want to change the encoding setting, be sure to contact SAS Technical Support.*

Occasionally, other NLS options are also included in the configuration file.

Selecting a Locale during SAS Foundation Deployment

A dialog box was added to SAS Deployment Wizard in SAS 9.3 that enables the installer to select the locale to use for SAS Foundation. The locale that initially displays in this dialog box is the user's locale on the UNIX machine where SAS Foundation is being installed. If you prefer to use a different locale, you can select a locale from the dialog box.

The selected locale is used as the value of the `LOCALE` system option in the language-specific configuration file that matches the locale. If the selected locale matches a localization installed for the SAS Foundation image, the `!SASROOT/sas` symbolic link is set to the SAS invocation script for that localization. Otherwise, the `!SASROOT/sas` symbolic link is set to the appropriate English language script, which is:

`!SASROOT/bin/sas_dbcs` for any language that requires DBCS support, or

`!SASROOT/bin/sas_en` for all other languages.

For example, if the French localization is installed and the French (Canada) [`fr_CA`] locale is selected, `!SASROOT/sas` is a symbolic link to `!SASROOT/bin/sas_fr`.

Starting with SAS 9.4M4, the ENCODING system option is set explicitly in the configuration file for single-byte languages, double-byte languages, and Unicode support.

Chinese, Japanese, and Korean DBCS Support

This section explains how to specify Asian font catalogs and how to determine the localization used for Chinese locales.

Also, be aware that full-screen products are NOT supported in 9.4 SAS for the following UNIX platforms and languages:

- HP-UX IPF: Japanese, Korean, Simplified Chinese, and Traditional Chinese
- AIX: Korean, Simplified Chinese, and Traditional Chinese

Setting System Fonts with X Resource Files

SAS 9.4 Foundation may not have the correct font settings for your locale by default. To ensure that the correct fonts are defined for the SAS System, you must add them to your X Resource files.

Japanese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_CDE.ja - for the CDE environment
- ./Resource_LNX.ja - for Linux
- ./Resource_Sun.ja - for Solaris
- ./Resource_HP.ja - for HP-UX
- ./Resource_IBM.ja - for AIX
- ./Resource_ReflX.ja - for ReflectionX users

Simplified Chinese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_HP.zh - for HP-UX
- ./Resource_LNX.zh - for Linux
- ./Resource_Sun.zh - for Solaris

Traditional Chinese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_HP.zt - for HP-UX
- ./Resource_HP.zt.euc - for HP-UX
- ./Resource_LNX.zt - for Linux
- ./Resource_Sun.zt - for Solaris
- ./Resource_Sun.zt.big5 - for Solaris

Korean X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_HP.ko - for HP-UX
- ./Resource_LNX.ko - for Linux
- ./Resource_Sun.ko - for Solaris

To apply the X Resources in these template files, copy the appropriate template to one of the following locations, renaming it to SAS (in all uppercase):

- /usr/lib/X11/app-defaults (on most UNIX systems)
- /usr/openwin/lib/X11/app-defaults (on Solaris)
- $HOME (your home directory)

For example, on a Solaris system, you would use the following COPY command:

```
$ cp !SASROOT/X11/resource_files/Resource_CDE.ja  /usr/openwin/lib/X11/app-defaults/SAS
```

In the example, !SASROOT refers to the root directory of your SAS 9.4 Foundation installation.

For more information, refer to the *SAS 9.4: National Language Support Reference Guide*.

Asian Font Catalogs

For SAS 9.4, Simplified and Traditional Chinese have been added to SASHELP.FONTS.

Specifying the Font Catalog in the Configuration File for Traditional Chinese Fonts

When you run a Traditional Chinese localization, the configuration file contains the GFONT definition for the location of the ZT font catalog in the UNIX DBCS directory. However, when you run the English version with LOCALE=ZH_TW, you must either set GFONT in your SAS session or you must modify the DBCS configuration file to define the GFONT definition for the ZT catalog, as follows

```
-set gfontx !SASROOT/nls/zt/font-name
```

In this statement

x represents a value from 0-9

font-name represents the name of the font catalog you want to use.

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

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

```
libname gfontx !SASROOT/nls/zt/font-name
```

In this statement

x represents a value from 0-9

font-name represents the name of the font catalog you want to use.

Chinese Localizations

The installer has the option to install localizations for both Simplified Chinese and Traditional Chinese. Several Chinese-based locales are supported by SAS. In some cases, the localization selected for the locale may not be intuitive. The following table shows the language that SAS uses when you select one of the five Chinese locales. Note that the default language may be English.

Locale	Location of `sasv9.cfg` file	Language
Chinese (China) [zh_CN]	`!SASROOT/nls/zh`	Simplified Chinese
Chinese (Hong Kong) [zh_HK]	`!SASROOT/nls/zt`	Traditional Chinese
Chinese (Macau) [zh_MO]	`!SASROOT/nls/dbcs`	English
Chinese (Singapore) [zh_SG]	`!SASROOT/nls/dbcs`	English
Chinese (Taiwan) [zh_TW]	`!SASROOT/nls/zt`	Traditional Chinese

Invoking SAS for Secondary Encodings

If you selected Traditional Chinese or Japanese in the Language Selection dialog box, the localized content for both the primary and secondary encodings is installed for the language. If you want to invoke SAS using the secondary encoding, you can do so as follows:

- If Traditional Chinese is installed on AIX, HP-UX, or Linux, the script `!sasroot/bin/sas_zt.euc` will be created, and it should be used to run the secondary encoding.
- If Traditional Chinese is installed on Solaris, the script `!sasroot/bin/sas_zt.pcms` will be created, and it should be used to run the secondary encoding.
- If Japanese is installed on AIX or HP-UX, the script `!sasroot/bin/sas_ja.euc` will be created, and it should be used to run the secondary encoding.
- If Japanese is installed on Linux or Solaris, the script `!sasroot/bin/sas_ja.sjis` will be created, and it should be used to run the secondary encoding.

Chapter 8 – Post-Installation Configuration for SAS/ACCESS

Before beginning your SAS/ACCESS software configuration, you should determine the following information about your DBMS:

- The version or release of the DBMS client shared libraries installed on your operating system. This is important due to potential incompatibilities between DBMS versions or releases.
- The location of the DBMS client shared libraries. This is important so that SAS/ACCESS software can be loaded at execution time.

Refer to the following sections for detailed DBMS-specific instructions on configuring your environment to interface with your SAS/ACCESS software.

ODBC Drivers on Linux

Multiple ODBC drivers might use the same libodbc.a archive. Due to a versioning compatibility issue with selected SAS solutions, OpenSSL libraries that are used by the ODBC drivers are installed in a different location on Linux machines. The two required OpenSSL libraries are libcrypto.so.3 and libssl.so.3. When SAS 9.4M8 hot fixes are applied or when you upgrade to SAS 9.4M9, the location of these libraries is changed from:

SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/9.4/Driver

to

SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/9.4/Driver/**lib64**

This directory change applies only to Linux operating systems. For more information, see SAS Note 67666.

Enabling Co-location of Multiple SAS/ACCESS Products on AIX

The installation of multiple SAS/ACCESS products on the same UNIX machine is supported, but additional configuration might be required. For example, you might install multiple SAS/ACCESS products that rely on ODBC drivers. Multiple ODBC drivers often rely on the same libodbc.a archive. This situation can occur when you have both a DataDirect Driver manager and a unixODBC driver manager. In such a case, the SAS/ACCESS engine finds only the driver manager that corresponds to the first occurrence of the archives in the library path environment variable definition.

The workaround is to concatenate the information from multiple driver managers. However, you must make sure that the operating-system library path is set before the default path for the PostgreSQL ODBC driver,
<SASHOME>/SASODBCDriversfortheWebInfrastructurePlatformDataServer/9.4/Driver.
This default client path is where two required libraries are stored: libssl.a and libcrypto.a. It is slightly different on Linux machines with a SAS 9.4M8 hot fix, as described previously.

The library path for this driver must be defined ahead of any other SAS/ACCESS library paths in the applicable library path environment variable definition:

- On Linux, Solaris for SPARC, and Solaris for x64, the environment variable is LD_LIBRARY_PATH

- On AIX, the environment variable is LIBPATH

You can add the unixODBC shared objects to the DataDirect archives by running the following commands:

```
chmod 777 libodbc.a
chmod 777 libodbcinst.a
ar -X64 -r libodbc.a libodbc.so.2
ar -X64 -r libodbcinst.a libodbcinst.so.2
chmod 555 libodbc.a
chmod 555 libodbcinst.a
```

SAS/ACCESS Interface to Amazon Redshift

The installation of SAS/ACCESS Interface to Amazon Redshift includes the required ODBC driver. It is saved to the following location during the installation of SAS/ACCESS Interface to Amazon Redshift:

SASHOME/AccessClients/9.4/Redshift

where *SASHOME* is the path to the SAS home directory.

This location is the ODBCHOME directory, which is used to set up the paths to the shared libraries as well as the odbc.ini file shown below. You must set the ODBCHOME environment variable to your ODBC home directory before setting the ODBCINI and shared library environment variables, as shown in the examples below.

The odbc.ini system information file contains a list of possible data sources to connect to your Amazon Redshift servers. You can configure data sources in this file or use the DSN-less connection method. Use a text editor to edit the odbc.ini file to configure data sources.

The general format of the odbc.ini file is illustrated below:

```
[ODBC Data Sources]
redshift=SAS ACCESS to Amazon Redshift

[ODBC]
InstallDir=<my install dir>
Trace=0
TraceDll=<my install dir>/lib/odbctrac.so
TraceFile=odbctrace.out

[redshift]
Description=SAS ACCESS to Amazon Redshift
Driver=<my install dir>/lib/S0rsft<file version>.so
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<db>
EnableDescribeParam=1
EncryptionMethod=0
ExtendedColumnMetadata=0
FailoverGranularity=0
```

```
FailoverMode=0
FailoverPreconnect=0
HostName=<Amazon Redshift host>
HostNameInCertificate=
IANAAppCodePage=
InitializationString=
KeepAlive=0
KeyPassword=
KeyStore=
KeyStorePassword=
LoadBalanceTimeout=
LoginTimeout=15
LogonID=
MaxCharSize=
MaxPoolSize=100
MaxVarcharSize=
MinPoolSize=0
Pooling=0
PortNumber=<Amazon Redshift server port>
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
```

Note that the <*driver version*> and <*file version*> specifications describe specific versions of the ODBC driver that is installed with SAS/ACCESS Interface to Amazon Redshift. The <*driver version*> that is referenced in your `odbc.ini` already contains the latest version of the driver that SAS ships. In addition, the <*file version*> contains a two-digit value that describes the version of the actual driver library. You do not need to update these two version specifications in your `odbc.ini` file.

Replace all occurrences of <*my install dir*> in the sample `odbc.ini` file with the installed location of the Amazon Redshift ODBC driver, *SASHOME*`/AccessClients/9.4/Redshift`. This directory must be the same directory that is specified by the `ODBCHOME` environment variable that you set earlier. You must also replace <*Amazon Redshift host*> with the IP address or named server of your Amazon Redshift server, and replace *Amazon Redshift server port* with the port where your Amazon Redshift server is listening (typically 5439). Finally, replace <*db*> with the name of your Amazon Redshift database.

In the above example, `redshift` corresponds to the configured data source name that is used in the `DSN=` option when assigning a LIBREF to the SAS/ACCESS Interface to Amazon Redshift.

A sample completed `odbc.ini` file is included below for your reference:

```
[ODBC Data Sources]
redshift=SAS ACCESS to Amazon Redshift

[ODBC]
InstallDir=/SASHome/AccessClients/9.4/Redshift
Trace=0
TraceDll=/SASHome/AccessClients/9.4/Redshift/lib/odbctrac.so
TraceFile=odbctrace.out
```

```
[redshift]
Driver=/SASHome/AccessClients/9.4/Redshift/lib/S0rsft27.so
Description=SAS ACCESS to Amazon Redshift
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database= sample
EnableDescribeParam=1
EncryptionMethod=0
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
HostName= company.domain.region.redshift.amazonaws.com
HostNameInCertificate=
IANAAppCodePage=
InitializationString=
KeepAlive=0
KeyPassword=
KeyStore=
KeyStorePassword=
LoadBalanceTimeout=
LoginTimeout=15
LogonID=
MaxCharSize=
MaxPoolSize=100
MaxVarcharSize=
MinPoolSize=0
Pooling=0
PortNumber=5439
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
```

After you configure your data sources, set the `ODBCINI` environment variable to the location and name of your `odbc.ini` file:

- For Bourne Shell:

  ```
  ODBCINI=$ODBCHOME/odbc.ini
  export ODBCINI
  ```

- For C Shell:

  ```
  setenv ODBCINI $ODBCHOME/odbc.ini
  ```

The required drivers that SAS includes with SAS/ACCESS Interface to Amazon Redshift are ODBC API-compliant shared libraries that are referred to in UNIX as shared objects. Include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $ODBCHOME/lib:${LIBPATH}`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv LIBPATH $ODBCHOME/lib:${LIBPATH}`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:${LD_LIBRARY_PATH}`

It is possible to invoke multiple ODBC-based SAS/ACCESS products in the same SAS session on UNIX. However, you must first define the driver names in a single `odbcinst.ini` configuration file. In addition, if you use DSNs in your SAS/ACCESS connections, you must define the data sources in a single `odbc.ini` configuration file. You cannot pass a delimited string of files for the `ODBCINSTINI` or `ODBCINI` environment variables. The requirement to use a single initialization file extends to any situation in which you are running multiple ODBC-based SAS/ACCESS products. For more information, see "SAS/ACCESS Interface to ODBC" on page 56.

SAS/ACCESS Interface to Aster

Installing and Configuring the ODBC Driver and Bulk Loader

Before you configure the ODBC driver, the bulk loader should be installed in *SASHOME*`/SASFoundation/9.4/` or another location that is in the `PATH` environment variable.

The `odbcinst.ini` system information file contains the driver definition to connect to your Aster server. You must configure the default Aster driver to use SAS/ACCESS Interface to Aster. A sample `odbcinst.ini` file may be included with Aster ODBC Driver. You will have to edit the sample file with a text editor to configure the driver.

The Aster ODBC Driver's main file has a name similar to *`libAsterDriver.so`*. To begin, find the directory that contains this file. The location is determined by your version of Aster ODBC Driver and your UNIX host. For more information, refer to your Aster ODBC Driver documentation.

The general format of the `odbcinst.ini` file is shown below:

```
[AsterDriver]
Driver=path-to/libAsterDriver.so
IconvEncoding=UCS-4LE
```

After you configure your driver, you must set the `ODBCSYSINI` environment variable to the location of your `odbcinst.ini`. Normally the file is in the `Setup` folder. Use the actual name of the directory.

- For Bourne Shell:

```
ODBCSYSINI=path-to-driver-install/Setup
export ODBCSYSINI
```

- For C Shell:

```
setenv ODBCSYSINI path-to-driver-install/Setup
```

Create `.aster.ini` in your home directory. You can copy it from Aster Database driver's Setup directory.

```
# cd /usr/local/lib/stage/clients-odbc-linux64/Setup
# cp aster.ini ~/.aster.ini
```

Note that you must rename the `aster.ini` file, adding a dot at the beginning of the file name.

Here is a sample `aster.ini`:

```
 [driver]
DriverManagerEncoding=UTF-8
DSILogging=0
ErrorMessagesPath=/Drivers/AsterDriver/ErrorMessages
ODBCInstLib=/Drivers/AsterDriver/DataDirect/lib/libodbcinst.so
```

Note that SAS requires DriverManagerEncoding to be UTF-8.

You can refer to Aster Database document for more information about client settings.

The `odbc.ini` system information file contains a list of possible data sources to connect to your Aster servers. Optionally, you can configure at least one data source to use the SAS/ACCESS Interface to Aster. A sample `odbc.ini` file may be included with the Aster ODBC Driver. You will have to edit the `odbc.ini` file with a text editor to configure the data sources. The general format of the `odbc.ini` file is shown below:

```
[ODBC Data Sources]
nCluster=AsterDriver

 [nCluster]
Driver=AsterDriver
DATABASE=beehive
SERVER=127.0.0.1
UID=beehive
PWD=beehive
   PORT=2406
```

After you configure your data sources, if `odbc.ini` is not located in the path that the `ODBCSYSINI` environment variable is set to, you must set the `ODBCINI` environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell:

```
ODBCINI=path-to/odbc.ini
export ODBCINI
```

- For C Shell:

```
setenv ODBCINI path-to/odbc.ini
```

Finally, you must include the full path to the shared libraries (dependencies of Aster ODBC Driver) in the shared library path. This allows them to be loaded dynamically at run time. Specific details of the configuration process are determined by your version of Aster ODBC Driver and your UNIX host. For more information, refer to your Aster ODBC Driver documentation.

AIX	
Bourne Shell	`$ LIBPATH=`*`path to driver install`*`/Libs:`*`path-to-driver-install`*`/DataDirect/lib:`*`path-to-driver-install`*`/unixODBC/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH =`*`path-to-driver-install`*`/Libs:`*`path-to-driver-install`*`/DataDirect/lib:`*`path-to-driver-install`*`/unixODBC:$LIBPATH`
Linux for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=`*`path-to-driver-install`*`/Libs:`*`path-to-driver-install`*`/DataDirect/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH `*`path to driver install`*`/Libs:`*`path-to-driver-install`*`/DataDirect/lib:`*`path-to-driver-install`*`/unixODBC/lib:${LD_LIBRARY_PATH}`

Solaris	
Bourne Shell	`$ LD_LIBRARY_PATH=`*`path-to-driver-install`*`/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH=`*`path-to-driver-install`*`/lib:${LD_LIBRARY_PATH}`

SAS/ACCESS Interface to DB2

The SAS/ACCESS Interface to DB2 executable uses shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables, and, if necessary, indicate the IBM Db2 version that you have installed at your site. You must also set the `INSTHOME` environment variable to your Db2 home directory before setting the environment variables as shown in the examples.

AIX	
Bourne Shell	`$ LIBPATH=$INSTHOME/lib64:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $INSTHOME/lib64:$LIBPATH`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$INSTHOME/lib64:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH $INSTHOME/lib64:$SHLIB_PATH`

Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$INSTHOME/lib64:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $INSTHOME/lib64:$LD_LIBRARY_PATH`

SAS/ACCESS Interface to Google BigQuery

In order to avoid connection errors when you connect to Google BigQuery with a `LIBNAME` statement, you can specify a value for the `GOMEMLIMIT=` environment variable. This variable must be set from the command line prior to starting your SAS session. Here is an example that shows how to set this variable.

```
export GOMEMLIMIT=250MiB
```

SAS/ACCESS Interface to Greenplum

Earlier maintenance releases of SAS 9.4 provided an option in the SAS Deployment Wizard to specify a location for the required ODBC driver as part of the initial installation of SAS/ACCESS Interface to Greenplum. The dialog box related to the ODBC driver has been removed. The required ODBC drivers are now installed automatically in the following location:

SASHOME`/AccessClients/9.4/Greenplum`

where *SASHOME* is the path to the SAS home directory.

This location is the `ODBCHOME` directory, which is used to set up the paths to the shared libraries as well as the `odbc.ini` file that is discussed below. You must set the `ODBCHOME` environment variable to your ODBC home directory before setting the `ODBCINI` and shared library environment variables as shown in the examples that follow.

The `odbc.ini` system information file contains a list of possible data sources to connect to your Greenplum servers. You must configure at least one data source in order to use the SAS/ACCESS Interface to Greenplum. Use a text editor to edit the `odbc.ini` file to configure the data sources.

The general format of the `odbc.ini` file is described below:

```
[ODBC Data Sources]
greenplum=SAS ACCESS to Greenplum

[ODBC]
InstallDir=<my install dir>
Trace=0
TraceDll=<my install dir>/lib/odbctrac.so
TraceFile=odbctrace.out

[greenplum]
Driver=<my install dir>/lib/S0gplm<file version>.so
Description=SAS ACCESS to Greenplum
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
```

```
Database=<db>
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=<Greenplum host>
InitializationString=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=<Greenplum server port>
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

Note that the <*driver version*> and <*file version*> specifications describe specific versions of the DataDirect Greenplum driver that is installed with SAS/ACCESS Interface to Greenplum. The <*driver version*> in your `odbc.ini` will already contain the latest version of the DataDirect driver that SAS ships. Also, the <*file version*> will contain a two-digit value that describes the version of the actual driver library. No updates to these two version specifications are required in your `odbc.ini` file.

Replace all occurrences of <*my install dir*> in the sample `odbc.ini` with the path and directory where you installed the Greenplum ODBC driver. This must be the same directory that is specified by the `ODBCHOME` environment variable, which you set earlier. You should also replace <*Greenplum host*> with the IP address or hostname of your Greenplum server, replace <*Greenplum server port*> with the port where your Greenplum server is listening (typically 5432), and replace <*db*> with the name of your Greenplum database.

In the above example, `greenplum` is the configured data source name that is used in the `DSN=` `option` when assigning a libref with SAS/ACCESS Interface to Greenplum. A sample completed `odbc.ini` file is shown below for your reference:

```
[ODBC Data Sources]
Greenplum=SAS ACCESS to Greenplum


[ODBC]
InstallDir=/TECHDBI/odbc/gpdrv
Trace=0
TraceDll=/TECHDBI/odbc/gpdrv/lib/odbctrac.so
TraceFile=/tmp/odbctrace.out


[greenplum]
```

```
Driver=/TECHDBI/odbc/gpdrv/lib/S0gplm701.so
Description=SAS ACCESS to Greenplum
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=sample
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=machine.unx.company.com
InitializationString=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=5432
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

After you configure your data sources, you must set the `ODBCINI` environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell:
  ```
  ODBCINI=$ODBCHOME/odbc.ini
  export ODBCINI
  ```
- For C Shell:
  ```
  setenv ODBCINI $ODBCHOME/odbc.ini
  ```

The DataDirect Greenplum ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $ODBCHOME/lib:${LIBPATH}`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH $ODBCHOME/lib:${SHLIB_PATH}`

Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:${LD_LIBRARY_PATH}`

Bulk Load

SAS/ACCESS can interface with the Greenplum Client Loader interface for loading large volumes of data. To perform bulk loading, the Greenplum Client Loader Package must be present on the system where you install SAS.

SAS recommends using the "gpfdist" protocol for bulk loading. For this protocol, you must set the `GPLOAD_HOME` environment variable to point to the location where the gpfdist utility allocates the files to be loaded into the Greenplum tables. Additional information about the bulk loading feature can be found in the *SAS/ACCESS 9.4 for Relational Databases: Reference* documentation.

SAS/ACCESS Interface to Hadoop

Important: *Starting with SAS 9.4M9, the procedures for deploying SAS/ACCESS Interface to Hadoop have changed. Several manual steps are required in order to configure the drivers and run a configuration script.*

For information about how to configure SAS/ACCESS Interface to Hadoop, see the *SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS* in the SAS Help Center.

SAS/ACCESS Interface to HAWQ

SAS/ACCESS Interface to HAWQ requires a 64-byte ODBC driver manager and an ODBC driver, which is included with the installation. The ODBC driver that is installed automatically is the Progress DataDirect ODBC Greenplum driver. It is saved to the following location during the installation of SAS/ACCESS Interface to HAWQ:

SASHOME`/AccessClients/9.4/Greenplum`

where *SASHOME* is the path to the SAS home directory.

This location is the `ODBCHOME` directory, which is used to set up the paths to the shared libraries as well as the `odbc.ini` file shown below. You must set the `ODBCHOME` environment variable to your ODBC home directory before setting the `ODBCINI` and shared library environment variables, as shown in the examples below.

The `odbc.ini` system information file contains a list of possible data sources to connect to your HAWQ servers. You must configure at least one data source in order to use SAS/ACCESS Interface to HAWQ. Use a text editor to edit the `odbc.ini` file to configure the data sources.

The general format of the `odbc.ini` file is described below:

```
[ODBC Data Sources]
greenplum=SAS ACCESS to Greenplum

[ODBC]
InstallDir=<my install dir>
```

```
Trace=0
TraceDll=<my install dir>/lib/odbctrac.so
TraceFile=odbctrace.out

[greenplum]
Driver=<my install dir>/lib/S0gplm<file version>.so
Description=SAS ACCESS to HAWQ
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<db>
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=<HAWQ host>
InitializationString=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=<HAWQ server port>
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

Note that the <*driver version*> and <*file version*> specifications describe specific versions of the Progress DataDirect Greenplum driver that is installed with SAS/ACCESS Interface to HAWQ. The <*driver version*> in your `odbc.ini` will already contain the latest version of the Progress DataDirect driver that SAS ships. Also, the <file version> will contain a two-digit value that describes the version of the actual driver library. You will not need to update these two version specifications in your `odbc.ini` file.

Replace all occurrences of <*my install dir*> in the sample `odbc.ini` with the path and directory where you installed the Greenplum ODBC driver. This must be the same directory that is specified by the `ODBCHOME` environment variable that you set earlier. You should also replace <*HAWQ host*> with the IP address or named server of your HAWQ server, replace <*HAWQ server port*> with the port where your HAWQ server is listening (typically 5432), and replace <*db*> with the name of your HAWQ database.

In the above example, `greenplum` is the name of the configured data source name that is used in the `DSN= option` when assigning a libref to the SAS/ACCESS Interface to HAWQ.

A sample completed `odbc.ini` is included below for your reference:

```
[ODBC Data Sources]
greenplum=SAS ACCESS to Greenplum

[ODBC]
InstallDir=/TECHDBI/odbc/gpdrv
Trace=0
TraceDll=/TECHDBI/odbc/gpdrv/lib/odbctrac.so
TraceFile=/tmp/odbctrace.out

[greenplum]
Driver=/TECHDBI/odbc/gpdrv/lib/S0gplm701.so
Description=SAS ACCESS to HAWQ
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=sample
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=machine.unx.company.com
InitializationString=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=5432
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

After you configure your data sources, you must set the `ODBCINI` environment variable to the location and name of your `odbc.ini`:

For Bourne Shell:

```
ODBCINI=$ODBCHOME/odbc.ini
export ODBCINI
```

For C Shell:

```
setenv ODBCINI $ODBCHOME/odbc.ini
```

The DataDirect Greenplum ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $ODBCHOME/lib:${LIBPATH}`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH $ODBCHOME/lib:${SHLIB_PATH}`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:${LD_LIBRARY_PATH}`

Bulk Load

SAS/ACCESS Interface to HAWQ can interface with the Greenplum Client Loader interface for loading large volumes of data. To perform bulk loading, the Greenplum Client Loader Package must be present on the system where you install SAS.

SAS recommends using the "gpfdist" protocol for bulk loading. To enable this protocol, you must set the `GPLOAD_HOME` environment variable to point to the location where the gpfdist utility allocates the files to be loaded into the HAWQ tables. Additional information about the bulk loading feature can be found in the *SAS/ACCESS 9.4 for Relational Databases: Reference* documentation.

SAS/ACCESS Interface to Impala

The Impala ODBC driver is an ODBC API-compliant shared library. You must include the full path to the shared library in the shared library path as shown below so that the Impala driver can be loaded dynamically at run time. In the table below the installation directory for the Impala ODBC driver is represented by `Impala_ODBC_driver_install_directory`.

In addition, the Impala ODBC driver requires that a third-party ODBC Driver Manager be installed as well. A current version of the unixODBC Driver Manager (freeware) is available for download from the unixODBC website at `http://www.unixodbc.org/download.html`. In the table below, the installation directory for the ODBC Driver Manager is represented by `$ODBCHOME`.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:`*`Impala_ODBC_driver_install_directory`*`/lib/64:$ LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $ODBCHOME/lib:`*`Impala_ODBC_driver_install_directory`*`/lib/64:${LIBPATH }`

Linux for x64	
Bourne Shell	$ LD_LIBRARY_PATH=$ODBCHOME/lib:*Impala_ODBC_driver_install_directory*/lib/64:$LD_LIBRARY_PATH $ export LD_LIBRARY_PATH
C Shell	$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:*Impala_ODBC_driver_install_directory*/lib/64:${LD_LIBRARY_PATH}

SAS/ACCESS Interface to Impala allows multiple ODBC drivers to be used to communicate with Impala.

The default driver used by SAS/ACCESS Interface to Impala is the Cloudera Impala ODBC driver. The Cloudera driver requires the `CLOUDERAIMPALAINI` environment variable to be set to the path and filename of the `.cloudera.impalaodbc.ini` configuration file. See the *Cloudera ODBC Driver for Impala Installation Guide* for more information.

To use an Impala ODBC driver from a different vendor in SAS/ACCESS Interface to Impala, you must set either the `SAS_IMPALA_DRIVER_VENDOR` environment variable or the `DRIVER_VENDOR` `libname` option as shown in the following examples:

Setting the environment variable to use the MapR Impala ODBC driver:

```
$  SAS_IMPALA_DRIVER_VENDOR=MAPR
$  export SAS_IMPALA_DRIVER_VENDOR
```

Setting the `libname` option to use the Progress DataDirect Impala ODBC driver:

```
libname implib impala server=impserver schema=default
DRIVER_VENDOR=DATADIRECT;
```

Currently the only valid values for the driver vendor are `DATADIRECT` and `MAPR`. For more information on selecting a different driver vendor, refer to the SAS/ACCESS Interface to Impala section of the *SAS/ACCESS 9.4 for Relational Databases Reference*.

The MapR Impala ODBC driver requires the `SIMBAINI` environment variable to be set to the path and filename of the `mapr.impalaodbc.ini` configuration file. See the *MapR Impala ODBC Driver Configuration Guide* for more information.

SAS/ACCESS Interface to Informix

SAS/ACCESS Interface to Informix uses an ODBC interface to access Informix.

An administrator must configure the Informix DB-Access utility if you want to perform bulk loading and bulk unloading. The Informix DB-Access utility must be running on Red Hat Enterprise Linux 7.9 or later.

Enable the utility by copying two libraries into the SDK installation library path:

```
cp /lib64/libncurses.so.5  /Informix-SDK-installation-directory/lib
cp /lib64/libtinfo.so.5  /Informix-SDK-installation-directory/lib
ls -l /Informix-SDK-installation-directory/lib
```

The two libraries should be available to Base SAS. The DB-Access utility requires authenticated credentials on the host.

The location of the DB-Access utility must also be set. You can use the BL_DBACCESS_PATH= data set option to set this location. Be sure to specify full path to the DB-Access utility. You must also set BULKLOAD=YES. For more information, see the *SAS/ACCESS 9.4 for Relational Databases: Reference*.

You might have to use a text editor to edit the odbc.ini file in your home directory to configure data sources. Some ODBC driver vendors might allow system administrators to maintain a centralized copy of this file by setting the environment variable ODBCINI. Refer to your ODBC driver vendor's documentation to find more specific information.

The ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables so that ODBC drivers can be loaded dynamically at run time. You must also set the InformixDIR environment variable to your Informix home directory before setting the environment variables, as shown in the following examples:

AIX	
Bourne Shell	`$ LIBPATH = $InformixDIR/lib/cli:$InformixDIR/lib/esql:$LIBPATH` `$ LIBPATH`
C Shell	`$ setenv LIBPATH` `$ InformixDIR/lib/cli:$InformixDIR/lib/esql:$LIBPATH`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$InformixDIR/lib/cli:$InformixDIR/lib/esql:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH` `$ InformixDIR/lib/cli:$InformixDIR/lib/esql:$SHLIB_PATH`
Linux for x64 and Solaris	
Bourne Shell	`$ LD_LIBRARY_PATH=$InformixDIR/lib/cli:$InformixDIR/lib/esql:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $InformixDIR/lib/cli:$InformixDIR/lib/esql:$LD_LIBRARY_PATH`

SAS/ACCESS Interface to Microsoft SQL Server

SAS/ACCESS Interface to Microsoft SQL Server requires a 64-bit ODBC driver manager and ODBC driver. These ODBC client components (from Progress DataDirect) are included with SAS/ACCESS Interface to Microsoft SQL Server. They are automatically installed during the installation of SAS/ACCESS software in the following location:

SASHOME/AccessClients/9.4/SQLServer

where *SASHOME* is the path to the SAS home directory.

This location is the ODBCHOME directory, which is used to set up the paths to the shared libraries as well as the odbc.ini file below. The directory described above becomes the ODBCHOME directory, which is used to set up the paths to the shared libraries as well as the odbc.ini file that is discussed below. You must set the ODBCHOME environment variable to your ODBC home directory before setting the ODBCINI and shared library environment variables, as shown in the examples below.

The odbc.ini system information file contains a list of possible data sources to connect to your Microsoft SQL Server instances. You must configure at least one data source in order to use SAS/ACCESS Interface to Microsoft SQL Server. A sample odbc.ini file is located in the ODBCHOME directory as odbc.ini.sample. Use a text editor to edit the odbc.ini file to configure the data sources. The general format of the odbc.ini file is shown below:

```
[ODBC Data Sources]
sqlserver=SAS Institute, Inc <driver version> SQL Server Wire
Protocol

[sqlserver]
Driver=<my install dir>/lib/S0sqls<file version>.so
Description= SAS Institute, Inc <driver version> SQL Server Wire
Protocol
AlternateServers=
AlwaysReportTriggerResults=0
AnsiNPW=1
ApplicationName=
ApplicationUsingThreads=1
AuthenticationMethod=1
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
BulkLoadFieldDelimiter=
BulkLoadOptions=2
BulkLoadRecordDelimiter=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<database_name>
EnableBulkLoad=0
EnableQuotedIdentifiers=0
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchTSWTZasTimestamp=0
FetchTWFSasTime=1
GSSClient=native
HostName=<SQL_Server_host>
HostNameInCertificate=
InitializationString=
Language=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
PacketSize=-1
Password=
Pooling=0
PortNumber=<SQL_Server_server_port>
QueryTimeout=0
ReportCodePageConversionErrors=0
SnapshotSerializable=0
```

```
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
WorkStationID=
XMLDescribeType=-10

[ODBC]
InstallDir=<my install dir>
Trace=0
TraceDll=<my install dir>/lib/ S0trc<file version>.so
TraceFile=odbctrace.out
```

Note that the <*driver version*> and <*file version*> specifications describe specific versions of the DataDirect Microsoft SQL Server driver that is installed with SAS/ACCESS Interface to Microsoft SQL Server. The <*driver version*> in your `odbc.ini` already contains the latest version of the DataDirect driver that SAS ships. In addition, the <*file version*> contains a two-digit value that describes the version of the actual driver library. These two version specifications do not require updates in your `odbc.ini`.

Replace all occurrences of <*my install dir*> in the sample `odbc.ini` with the path and directory name that you specified during the SAS/ACCESS Software Configuration for Microsoft SQL Server. This must be the same directory that is specified by the `ODBCHOME` environment variable that you set earlier.

You should also replace <*SQLServer host*> with the IP address or named server of your SQL Server machine, replace <*SQLServer server port*> with the port number where your SQL Server is listening (typically 1433), and replace <*database_name*> with the name of your SQL Server database.

In the above example, `sqlserver` is the name of the configured data source name (DSN) that is used in the `DSN=` option when assigning a libref with SAS/ACCESS Interface to Microsoft SQL Server.

If you use a DSN, or if you connect to Microsoft Azure SQL Database, add the option `EnableScrollableCursors=3` to your DSN configuration in the odbc.ini file, or include it in the string used to make a `COMPLETE`, `PROMPT`, or `NOPROMPT` connection to SAS. A sample completed odbc.ini is shown below for reference:

```
[ODBC Data Sources]
sqlserver= SAS Institute, Inc 7.1 SQL Server Wire Protocol

[sqlserver]
Driver=/install/sas/driver/lib/S0sqls27.so
Description= SAS Institute, Inc 7.1 SQL Server Wire Protocol
AlternateServers=
AlwaysReportTriggerResults=0
AnsiNPW=1
ApplicationName=
ApplicationUsingThreads=1
AuthenticationMethod=1
BulkBinaryThreshold=32
BulkCharacterThreshold=-1
BulkLoadBatchSize=1024
BulkLoadFieldDelimiter=
BulkLoadOptions=2
```

```
BulkLoadRecordDelimiter=
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=users
EnableBulkLoad=0
EnableScrollableCursors=3
EnableQuotedIdentifiers=1
EncryptionMethod=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchTSWTZasTimestamp=0
FetchTWFSasTime=1
GSSClient=native
HostName=sqlserver.myhost.com
HostNameInCertificate=
InitializationString=
Language=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
PacketSize=-1
Password=
Pooling=0
PortNumber=1433
QueryTimeout=0
ReportCodePageConversionErrors=0
SnapshotSerializable=0
TrustStore=
TrustStorePassword=
ValidateServerCertificate=1
WorkStationID=
XMLDescribeType=-10
[ODBC]
InstallDir=/install/sas/driver
Trace=0
TraceDll=/install/sas/driver/lib/S0trc27.so
TraceFile=odbctrace.out
```

The bulk-loading feature is available with recent versions of the DataDirect ODBC SQL Server driver. Bulk loading is initiated with the following entry in `odbc.ini`:

```
EnableBulkLoad=1
```

Bulk loading is available on Solaris, HP UX for Itanium, AIX, and Linux platforms. You can download the driver from the Downloads tab in KB article 41127.

After you configure your data sources, you must set the `ODBCINI` environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell:
```
ODBCINI=$ODBCHOME/odbc.ini
export ODBCINI
```
- For C Shell:
```
setenv ODBCINI $ODBCHOME/odbc.ini
```

The DataDirect Microsoft SQL Server ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH  $ODBCHOME/lib:${LIBPATH}`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH  $ODBCHOME/lib:${SHLIB_PATH}`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH  $ODBCHOME/lib:${LD_LIBRARY_PATH}`

SAS/ACCESS Interface to MongoDB

The SAS/ACCESS Interface to MongoDB executable uses the MongoDB C API client libraries, referred to in UNIX as shared objects. You must add the location of the MongoDB C shared libraries to the shared library path environment variable specific to your operating system.

Modify the shared library variable based on the shell that you are using, according to the table below. The following table assumes that the `$MONGOC_LIBDIR` environment variable names the directory that contains the MongoDB C client libraries.

Linux for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$MONGOC_LIBDIR:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $MONGOC_LIBDIR:$LD_LIBRARY_PATH`

If the environment variables are not set correctly, you will see error messages that resemble the following when connecting to MongoDB from SAS*:*

```
ERROR: CLI error trying to establish connection: Unable to load
extension: (tkemongo) : Extension Load Failure: OS Error: -1

(libmongoc-1.0.so.0: cannot open shared object file: No such file or
directory): Could not load tkemongo Extension.

ERROR: Error in the LIBNAME statement.
```

SAS/ACCESS Interface to MySQL

The SAS/ACCESS Interface to MySQL executable uses the MySQL C API client library (libmysqlclient, a client library for C development). Be sure to obtain this library before attempting to use SAS/ACCESS Interface to MySQL. You can download the latest client library available from the MySQL website.

You must add the location of the MySQL shared libraries to the shared library path environment variable for your operating system. Modify the shared library variable based on the host and shell you are using, according to the table below.

Some versions of the MySQL C client library might have dependencies on other libraries (such as OpenSSL). Make sure these dependencies are also added to the shared library path environment variable if required. Consult the MySQL documentation for additional information.

The following table assumes that the `$MYSQL_LIBDIR` environment variable points to the directory that contains the MySQL client libraries:

AIX	
Bourne Shell	`$ LIBPATH=$MYSQL_LIBDIR:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $MYSQL_LIBDIR:$LIBPATH`
Linux for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$MYSQL_LIBDIR:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $MYSQL_LIBDIR:$LD_LIBRARY_PATH`
Solaris and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$MYSQL_LIBDIR:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $MYSQL_LIBDIR:$LD_LIBRARY_PATH`

If you receive the following error message, double-check your library path environment variable:

```
ERROR: The SAS/ACCESS Interface to MYSQL cannot be loaded. The
libmysqlclient code appendage could not be loaded.
ERROR: Error in the LIBNAME statement.
```

SAS/ACCESS Interface to Netezza

The IBM Netezza ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib64:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH  $ODBCHOME/lib64:${LIBPATH}`

HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib64:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH  $ODBCHOME/lib64:${SHLIB_PATH}`
Linux for x64 and Solaris	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib64:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH  $ODBCHOME/lib64:${LD_LIBRARY_PATH}`

SAS/ACCESS Interface to ODBC

You may have to edit the `odbc.ini` file in your home directory with a text editor to configure data sources. Some ODBC driver vendors may allow system administrators to maintain a centralized copy by setting the environment variable `ODBCINI`. Refer to your ODBC driver's vendor documentation to find more specific information.

If you want to use SAS/ACCESS Interface to ODBC to connect to an ODBC driver that is compatible with the unixODBC driver manager, the DataDirect driver manager is also supported. However, DataDirect requires a modification to the `odbc.ini` file in order to support UTF-8 encoding. Add the following line (in boldface type) to your `odbc.ini` file:

```
[sample]
Driver=/data01/home/db2v10/sqllib/lib64/libdb2o.so
Server=[server-name]
DriverUnicodeType=1
```

The ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables so that ODBC drivers can be loaded dynamically at run time. You must also set the `ODBCHOME` environment variable to your ODBC home directory before setting the environment variables as shown in the following examples.

AIX	
Bourne Shell	`$ LIBPATH=$ODBCHOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH  $ODBCHOME/lib:${LIBPATH}`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH  $ODBCHOME/lib:${SHLIB_PATH}`
Linux for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:$LD_LIBRARY_PATH`

Solaris and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH  $ODBCHOME/lib:${LD_LIBRARY_PATH}`

It is possible to invoke multiple ODBC-based SAS/ACCESS products in the same SAS session on UNIX. However, you must first define the driver names in a single `odbcinst.ini` configuration file. Also, if you decide to use DSNs in your SAS/ACCESS connections, the data sources must be defined in a single `odbc.ini` configuration file. You cannot pass a delimited string of files for the `ODBCINSTINI` or `ODBCINI` environment variables. The requirement to use a single initialization file extends to any situation in which you are running multiple ODBC-based SAS/ACCESS products.

You can create a single `odbcinst.ini` or `odbc.ini` by combining the contents of the individual files. These files are located in *`SASHOME`*`/AccessClients/9.4/`*`SAS/ACCESS-product-name`* for each of the ODBC-based SAS/ACCESS products that you have.

SAS/ACCESS Interface to Oracle

In order to use SAS/ACCESS Interface to Oracle, you must install and configure the Oracle client libraries. Refer to the Oracle documentation for instructions. Typically, two steps are required:

- Set the `ORACLE_HOME` environment variable.
  ***Note**: This step is not required if you are installing the Oracle instant client.*
- Set the shared library path variable (the name of this variable is operating system-dependent) so that it points to the location of the Oracle shared libraries. This step is required because SAS/ACCESS Interface to Oracle uses Oracle shared libraries and must be able to locate them at your site.

The following table provides examples for the various operating systems:

AIX	
Bourne Shell	`$ LIBPATH=$ORACLE_HOME/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH=$ORACLE_HOME/lib:$LIBPATH`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=$ORACLE_HOME/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH $ORACLE_HOME/lib:$SHLIB_PATH`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ORACLE_HOME/lib:$LD_LIBRARY_PATH`

If the environment variables are not set correctly, you will see error messages similar to the following when connecting to Oracle from SAS:

```
ERROR: Unable to load oracle client (libclntsh.so)
```

```
ERROR: Error in the LIBNAME statement.
```

All Oracle client installations require symbolic links in the installation directory. SAS/ACCESS Interface to Oracle uses those links to access the correct Oracle client at run time. Most Oracle client installers will automatically create the correct symbolic links. However, depending on the Oracle client installer you are using, those links might be missing (as is typically the case when using the Oracle instant client libraries). In that case, your Oracle administrator will have to create the missing symbolic links manually.

The following soft links are required:

- A soft link from libclntsh.so to the actual version of the Oracle client that you installed.
- If you have installed an Oracle 11g client, you will need a soft link from libclntsh.so.11.1 to libclntsh.so
- If you have installed an Oracle 12c client or later, you will need a soft link from libclntsh.so.12.1 to libclntsh.so

Here is an example of the soft links that are required in an Oracle 19c installation:

```
mybox> ls -al libclntsh.so*
lrwxrwxrwx 1 oracle dba       17 Jun 26  2024 libclntsh.so -> libclntsh.so.19.1
lrwxrwxrwx 1 oracle dba       12 Jun 26  2024 libclntsh.so.12.1 -> libclntsh.so
-rwxr-xr-x 1 oracle dba 79960736 Jun 26  2024 libclntsh.so.19.1
```

***Note**: You might have more soft links than the ones shown here. While extra soft links are not required for SAS to operate, they might be required for other Oracle tools.*

SAS/ACCESS Interface to PC Files

Use of the PCFILES libname requires SAS PC Files Server. You can run SAS PC Files Server as a Windows service or as an application listening to the PCFILES libname. SAS PC Files Server provides data encryption and authentication feature with default port number 9621, which can be configured through the PC Files application console. Note that the 32-bit or 64-bit SAS PC Files Server requires installation of the same "bitness" (32- or 64-bit) as the Microsoft ACE driver installed on the same Windows machine.

When you install SAS PC Files Server on Windows, check the option to **Start Service Now and Automatically when Windows Starts** to run SAS PC Files Server as a Windows Service in the background. This default setting both starts the service and sets the service startup type to `automatic`. To run SAS PC Files Server as a desktop application, ensure that this option is cleared.

***Note:** Only one instance of SAS PC Files Server can run at a given time. You cannot run SAS PC Files Server as a Windows Service and an Application Server at the same time.*

Both server mode and service mode store the server configuration settings in the Windows Registry. If you change the settings while in server mode, the server running service mode will be affected after it is restarted. The default configuration is sufficient for most installations.

To change the configuration options, run the server in server mode:

1. In Windows Services (**Start -> Run** and enter services.msc), stop the SAS PC Files Server service.
2. Start the PC Files Server in server mode by clicking **Start -> SAS -> PC Files Server**.

3. Modify the options that you want to change on the PC Files Server application console.
4. Click the **Shutdown** button to stop the PC Files Server in server mode.
5. Access Windows Services again. Start the SAS PC Files Server service.

SAS/ACCESS Interface to the PI System

Base SAS is required for the installation of SAS/ACCESS Interface to the PI System.

SAS/ACCESS Interface to the PI System uses the PI System Web API, which is HTTPS-based and RESTful. No PI System client software is required to be installed on the machine where SAS is running. However, the PI System Web API (PI Web API 2015-R2 or later) must be installed and activated on the host machine where the user connects.

To test your connection, issue the following commands:

```
LIBNAME x PISYSTEM
   HOST=WebAPI-host
   SERVER=data-server;
```

Replace "*data-server*" with the name of the PI System Data Archive server.

```
LIBNAME x PIAF
   HOST=WebAPI-host
   SERVER=asset-server
   DATABASE=asset-database;
```

Replace "*asset-database*" with the name of the PI System Asset Framework server.

Multi-byte Character Sets

The PI System Web API host, which runs on Windows, performs some internal transcoding using the Windows System Locale. Therefore, in order to use multi-byte character sets, such as Asian characters, you must set the correct Windows System Locale on the Web API host in the Control Panel (select **Region and Language** -> **Administrative** -> **Change System Locale**). A restart is required when this setting is changed.

In addition, you must start SAS using the correct locale or encoding and must also use the following setting in your SAS session:

```
options validvarname=any validmemname=extend;
```

Multi-byte characters can be used for PI System tag descriptors (labels) and string data values. However, the use of multi-byte characters as tag names has limitations. The PI System Asset Framework fully supports multi-bye characters. For more information, refer to the *SAS/ACCESS Interface to the PI System Reference Guide* at https://go.documentation.sas.com/?cdcId=pgmsascdc&cdcVersion=9.4_3.4&docsetId=acpi&docsetTarget=titlepage.htm.

Host Name and Port Number

To connect through the Web API, you must specify the host name on the `LIBNAME` statement. The port number of the Web API host is optional. This information is specified in addition to the database server, which may be running on a different machine. Use the `LIBNAME` options

`HOST="name"` and `PORT=number`. If `PORT=` is omitted, it defaults to 443, which is the standard port number for HTTPS connections.

Time Zone Settings

The Web API returns all timestamps in UTC (GMT) time, which SAS must convert to local time. By default, SAS uses the current time zone offset as per system settings. In locales that observe daylight savings time, this offset might vary by one hour, depending on the time of year. As an example, Eastern Time in the US is UTC-5hrs in the winter and UTC-4hrs in the summer. This disparity causes timestamps to be off by one hour for historical data that was collected during a time of year that differs from the time SAS is run. As a specific example, when SAS is run in December, a timestamp from May is off by one hour.

In order to handle this issue, SAS must be run with the correct `-timezone` option:

```
sas -timezone "America/New_York" other options;
```

Note that this option must be issued on the SAS command line. It cannot be supplied by means of the 'option set' feature within SAS or in `autoexec.sas`. Refer to "Specifying Time Zones in SAS" in the *SAS 9.4 National Language Support (NLS): Reference Guide* for the correct setting for your location. You must provide the full time-zone ID, similar to the example above. Do not supply 'EST', 'EDT', or 'ET'.

TLS (SSL) Certificate

HTTPS requires a TLS (formerly SSL, Secure Sockets Layer) certificate to authenticate with the host. Refer to the PI System Web API documentation for information about how to create this certificate. The resulting certificate file, with a `.pem` extension, must be made available to SAS using an SSL option. You can set this option at startup using the SAS command line, or you can set it within SAS using the "option set" feature (or from `autoexec.sas`). Each Web API host requires a unique certificate**, signed by a root certificate**. This root certificate path must be supplied to SAS. The syntax is as follows:

- On the SAS command line, add the following:
  ```
  -set SSLCALISTLOC "/usr/mydir/root.pem"
  ```
- Within SAS (or in `autoexec.sas`), use the following:
  ```
  options set=SSLCALISTLOC "/usr/mydir/root.pem";
  ```

Kerberos-only Web API Configuration

The PI System Web API Windows host can be configured for several authentication methods. (These methods are distinct from the SSL certificate, which is always required for the Web API). If the Web API host is configured to allow only Kerberos-authenticated connections, any SAS client connecting to that host through the Web API must properly set up Kerberos (`kinit`) and define an environment variable pointing to the resulting `krb5.conf` file, as in the following example:

```
export KRB5_CONFIG=/usr/mydir/krb5.conf
```

Refer to the PI System documentation from OSIsoft for more information about the `kinit` and principal string to use.

Also, when connecting to a Kerberos-only Web API host (HOST=) and to a server (SERVER=) that is different from the host, the host must be configured for "delegation in Active Directory". This setting enables the Kerberos credentials that are provided to the host to be proxied to the server. The PI System documentation from OSIsoft contains more information.

SAS/ACCESS Interface to PostgreSQL

SAS/ACCESS Interface to PostgreSQL uses theODBC driver for PostgreSQL that is included with Base SAS. The PostgreSQL ODBC driver is installed in the following location:

```
<SASHOME>/SASODBCDriversfortheWebInfrastructurePlatformDataServer/
9.4/Driver
```

where <SASHOME> is the path to the SAS home directory.

Note: *When SAS 9.4M8 hot fixes are applied or when you upgrade to SAS 9.4M9, the location of these libraries on Linux machines is changed. For more information, see* *ODBC Drivers on Linux on page 35.*

When SAS/ACCESS Interface to PostgreSQL code is submitted, this ODBC driver will automatically be referenced. As a result, it is not necessary to set environment variables to point to this ODBC driver.

With SAS/ACCESS Interface to PostgreSQL, you can either provide connection specifics in your code or reference a DSN. If the DSN method is used, you must create a serviceable ODBC.INI. You might also need to set the ODBCINI environment variable to point to it.

For more information, refer to the SAS/ACCESS Interface to PostgreSQL chapter in *SAS/ACCESS 9.4 for Relational Databases: Reference* and the documentation provided by PostgreSQL.

SAS/ACCESS Interface to R/3

SAS/ACCESS Interface to R/3 software 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 Salesforce

SAS/ACCESS Interface to Salesforce uses the Salesforce SOAP API. Therefore, API access must be enabled for the profiles of any users who will be connecting to Salesforce from SAS.

To enable API access:

1. Visit the Settings area of the Salesforce Administration website.
2. Under the **Administration** heading in the side bar, find the **Users** section and click **Profiles**.
3. Click **New** to create a new profile, or click **Edit** if you want to modify an existing user profile.
4. Verify that the **API Enable**d permission is checked for the selected profile.

Repeat these steps for the user accounts of all users who will make connections to Salesforce from SAS.

SAS/ACCESS Interface to SAP ASE

For users of SAP ASE Open Client 15, in order to correctly copy the SAP ASE libraries for use with SAS/ACCESS Interface to SAP ASE, you must have read/write authority for `$SYBASE/OCS-15_0/lib` and `$SYBASE/OCS-15_0/devlib` to run `$SYBASE/OCS-15_0/scripts/lnsyblib`. Instructions for copying the SAP libraries are in the header comments in the `lnsyblib` file.

Installing SAP ASE Procedure

In SAS 9.4, the administrator or user must install two SAP ASE (Sybase)-stored procedures on the target SAP server. These files are available in a compressed TGZ archive for download from the SAS Support site at https://support.sas.com/downloads/package.htm?pid=2458.

The archive includes the following files:

- `sas-spcp.txt` is a text file containing instructions on how to do the installation.
- `sas-spdf.txt` is the first of two actual stored procedure scripts for CTLIB 12.5x users
- `sas-spdf_15.txt` is the first of two actual stored procedure scripts for CTLIB 15 users
- `sassp2df.txt` is the second of two stored procedure scripts for CTLIB 12.5x users
- `sassp2df_15.txt` is the second of two stored procedure script for CTLIB 15 users.

The process uses two SAP ASE (Sybase) facilities, `defncopy` and `isql`.

Adding Shared Libraries

The SAS/ACCESS Interface to SAP ASE executable uses shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables and, if necessary, indicate the SAP ASE (Sybase) version that you have installed at your site. You must also set the SAP ASE environment variable to your SAP ASE home directory before setting the environment variables as shown in the following examples:

AIX	
Bourne Shell	`$ LIBPATH=$SYBASE/lib:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH $SYBASE/lib:$LIBPATH`
HP-UX for the Itanium Processor Family	
Bourne Shell	`$ SHLIB_PATH=$SYBASE/lib:/lib:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH $SYBASE/lib:/lib:$SHLIB_PATH`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=$SYBASE/lib:/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $SYBASE/lib:/lib:$LD_LIBRARY_PATH`

SAS/ACCESS Interface to SAP HANA

SAS/ACCESS Interface to SAP HANA uses an ODBC interface to access SAP HANA.

SAS/ACCESS Interface to SAP HANA requires the 64-bit ODBC driver for SAP HANA.

These are the prerequisites for configuration of SAS/ACCESS Interface to SAP HANA:

- You have downloaded the SAP HANA client software from SAP Service Marketplace and installed and configured the ODBC driver.
- You have included the full path to the shared library in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

For more information about how to obtain the software, see the *SAP HANA Master Guide* on the following website: https://help.sap.com/docs/SAP_HANA_PLATFORM

For information about how to install and configure the ODBC driver, refer to the *SAP HANA Client Installation and Update Guide*, available from the same website.

The following table assumes the SAP HANA client libraries are installed in `/usr/sap/hdbclient`:

AIX	
Bourne Shell	`$ LIBPATH=/usr/sap/hdbclient:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH /usr/sap/hdbclient:$LIBPATH`
Linux for Intel Architecture and Solaris	
Bourne Shell	`$ LD_LIBRARY_PATH=/usr/sap/hdbclient:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH /usr/sap/hdbclient:$LD_LIBRARY_PATH`
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	`$ SHLIB_PATH=/usr/sap/hdbclient:$SHLIB_PATH` `$ export SHLIB_PATH`
C Shell	`$ setenv SHLIB_PATH /usr/sap/hdbclient:$SHLIB_PATH`

If you receive the following error message, verify your library path environment variable:

```
ERROR: Could not load path/sashna (35 images loaded)
ERROR: libodbcHDB.so: cannot open shared object file: No such file
or directory
ERROR: The SAS/ACCESS Interface to SAPHANA cannot be loaded. The
SASHNA code appendage could not be loaded.
ERROR: Error in the LIBNAME statement.
```

SAS/ACCESS Interface to SAP HANA can use data sources that are defined in the `odbc.ini` file to identify the SAP HANA server.

The general format of the `odbc.ini` file is shown below:

```
[ODBC Data Source]
SERVERNODE=hana_host:hana_port
```

Here is an example:

```
[SAPHANADSN]
SERVERNODE=hanasrv1.mycompany.com:30015
```

You must set the `ODBCINI` environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell:

```
ODBCINI=path-to/odbc.ini
export ODBCINI
```

- For C Shell:

```
setenv ODBCINI path-to/odbc.ini
```

Bulk Load

The bulk load feature for SAS/ACCESS Interface to SAP HANA uses SFTP to transfer files to the SAP HANA server.

If you want to use the bulk load feature, you must configure SFTP for accessing the SAP HANA server from the SAS server.

SAS/ACCESS Interface to SAP IQ

You must first install the SAP IQ client software (formerly Sybase IQ) available from SAP. After the installation is complete, you must run a script to set up your environment, including the path to the shared library.

This script is located in the installation directory for your SAP IQ client software and is named for your version of SAP IQ. For SAP IQ version 16.0, the script name is `IQ-16_0.sh`.

SAS/ACCESS Interface to Snowflake

The configuration requirements for SAS/ACCESS Interface to Snowflake depend on the maintenance level of SAS 9.4 Foundation. Guidance related to the new multifactor authentication requirement from Snowflake is provided in Multifactor Authentication below.

Multifactor Authentication (SAS 9.4M6 and Later)

Snowflake in 2025 has announced plans to require multifactor authentication (MFA) on any deployment that is using basic user name/password authentication.

By default, SAS/ACCESS Interface to Snowflake operates as designed and defers MFA authentication to Snowflake. However, for programmatic or batch access to data stored in Snowflake, SAS 9.4 does not support the use of the Snowflake MFA credentials. SAS has determined that Snowflake RSA key-pair authentication represents an optimal configuration for these purposes. For SAS 9.4M6 and later users who are attempting to access Snowflake databases, the recommended workflow is to modify SAS code to replace user name/password authentication with RSA key pairs, as described in the Snowflake document titled Key-pair authentication and key-pair rotation.

The Snowflake database administrator can follow the instructions in the Snowflake document titled Configuring key-pair authentication in order to set up key pairs for users.

When user accounts have been configured to use RSA keys, and when users have access to the private key file and the key password, the key information can be included in the `LIBNAME` statements for Snowflake. SAS recommends setting permissions on the RSA key file and your SAS program to limit access to these files.

For more information, see Authentication to Snowflake in the *SAS 9.4 and SAS Viya 3.5 Programming Guide*.

For SAS 9.4M6 and SAS 9.4M7

In SAS 9.4M7, SAS/ACCESS Interface to Snowflake requires the ODBC driver for Snowflake, which is an ODBC API-compliant shared library, to be installed. You must include the full path to the shared library in the shared library path so that the Snowflake driver can be loaded dynamically at run time. In the table below, the installation directory for the Snowflake ODBC driver is represented by the variable *Snowflake-ODBC-driver-install-directory*.

In addition, the Snowflake ODBC driver requires you to install a third-party ODBC driver manager. For information about installing the Snowflake ODBC driver and the required ODBC driver manager, see Installing and Configuring the ODBC driver for Linux in the Snowflake Developer Guide.

In the table below, the installation directory for the ODBC driver manager is represented by `$ODBCHOME`.

Linux for X64	
Bourne Shell	`$ LD_LIBRARY_PATH=$ODBCHOME/lib:`*`Snowflake-ODBC-driver-install-directory`*`/lib:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH $ODBCHOME/lib:`*`Snowflake-ODBC-driver-install-directory`*`/lib:${LD_LIBR ARY_PATH}`

For SAS 9.4M8 and Later

Starting with SAS 9.4M8, the installation of SAS/ACCESS Interface to Snowflake includes a Snowflake ODBC driver component. It is automatically installed during the installation of SAS/ACCESS software in the following location:

```
SASHOME/AccessClients/9.4/Snowflake
```

where *SASHOME* is the path to the SAS home directory.

SAS/ACCESS Interface to Spark

For information about how to configure SAS/ACCESS Interface to Spark, see the *SAS 9.4 Hadoop Configuration Guide for Base SAS and SAS/ACCESS*.

SAS/ACCESS Interface to Teradata

Access to Shared Libraries

The SAS/ACCESS Interface to Teradata executable uses shared libraries, referred to in UNIX as shared objects. These shared objects typically reside in `/opt/teradata/client/`. You must add the location of the shared libraries to one of the system environment variables.

AIX	
Bourne Shell	`$ LIBPATH=`*`TERADATA-CLIENT-LOCATION`*`:$LIBPATH` `$ export LIBPATH`
C Shell	`$ setenv LIBPATH `*`TERADATA-CLIENT-LOCATION`*`:$LIBPATH`
HP-UX for the Itanium Processor Family	

Bourne Shell	`$ SHLIB_PATH=`*`TERADATA-CLIENT-LOCATION`*`:$SHLIB_PATH`
	`$ export SHLIB_PATH`
	`$ LD_PRELOAD=/usr/lib/hpux64/libpthread.so.1`
	`$ export LD_PRELOAD`

C Shell	`$ setenv SHLIB_PATH `*`TERADATA-CLIENT-LOCATION`*`:$SHLIB_PATH` `$ setenv LD_PRELOAD /usr/lib/hpux64/libpthread.so.1`
Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	`$ LD_LIBRARY_PATH=`*`TERADATA-CLIENT-LOCATION`*`:$LD_LIBRARY_PATH` `$ export LD_LIBRARY_PATH`
C Shell	`$ setenv LD_LIBRARY_PATH `*`TERADATA-CLIENT-LOCATION`*`:$LD_LIBRARY_PATH`

Teradata Parallel Transporter

SAS/ACCESS supports the Teradata Parallel Transporter API for loading data using MultiLoad (`TPT UPDATE` operator), FastLoad (`TPT LOAD` operator) and multi-statement inserts (`TPT STREAM` operator). The API also supports reading data using FastExport (`TPT EXPORT` operator).

Note: *Starting with SAS 9.4M9, TPT API is now the* ***default*** *and* ***the only supported*** *method that is used for all Teradata utility processing in SAS/ACCESS Interface to Teradata.*

To determine whether TPT is installed correctly, perform a quick test within SAS. Issue a Teradata libref, and then invoke `fastload` with a single row:

```
/* quick test to see if TPT is installed correctly */
/* TPT = YES b default                             */

Libname x teradata server=mydbc user=dtest pw=xxxxx ;

Data x.new (fastload=yes) ; x=99; run;

NOTE: The data set X.new has 1 observations and 1 variables
NOTE: Teradata connection: TPT Fastload has inserted 1 row(s).
NOTE: Data statement used(Total process time):

real time      4.39
cpu time 0.29
```

TPT or Legacy Teradata Utility (FastExport, FastLoad, and Multiload)

Important *Starting with SAS 9.4M9, TPT API is the **default** and the **only supported** method that is used for all Teradata utility processing. Documentation that describes the use of the Legacy Teradata Utility remains for users who have not upgraded to SAS 9.4M9. See Legacy Teradata Utility Configuration, below, for more information.*

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.
- Use of 64-bit libraries on 64-bit operating systems.
- 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

Important *Starting with SAS 9.4M9, TPT API is the **default** and the **only supported** method that is used for all Teradata utility processing. Documentation that describes the use of the Legacy Teradata Utility 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.

As needed, modify your library path environment variable to include the directory containing `sasaxsm.sl` (HP-UX) or `sasaxsm.so` (Linux, Solaris, and AIX). These shared objects are delivered in the `$SASROOT/sasexe` directory. You can copy these modules to other locations, but ensure that the directory where you copy them is in the appropriate shared library path environment variable.

The library path variables are as follows:

- Solaris and Linux: `LD_LIBRARY_PATH`
- HP-UX: `SHLIB_PATH`
- AIX: `LIBPATH`

Also, make sure that the directory for the Teradata FastExport utility, fexp, is included in the `PATH` environment variable. This utility is usually installed in the `/usr/bin` directory.

The FastExport Utility is not required by SAS 9.4 Foundation; SAS/ACCESS reads large tables efficiently without it. For further information, see the `DBSLICEPARM` option in your SAS/ACCESS Interface to Teradata documentation. Contact Teradata if you want to obtain the Teradata FastExport Utility.

MultiLoad

Important *Starting with SAS 9.4M9, TPT API is the **default** and the **only supported** method that is used for all Teradata utility processing.*

SAS/ACCESS can interface with MultiLoad for loading large volumes of data. To perform Multi-Loading, the Teradata MultiLoad Utility must be present on the system where you install SAS.

As needed, modify your library path environment variable to include the directory containing the shared objects `sasmlam.sl` and `sasmlne.sl` (HP-UX) or `sasmlam.so` and `sasmlne.so` (Linux, Solaris, HP-UX for the Itanium Processor Family and AIX). These shared objects are delivered in the `$SASROOT/sasexe` directory. You can copy these modules to other locations, but ensure that the directory into which you copy them is in the appropriate shared library path environment variable.

The library path variables are as follows:

- On Solaris and Linux: `LD_LIBRARY_PATH`
- On HP-UX for the Itanium Processor Family: `SHLIB_PATH`
- On AIX: `LIBPATH`

Also, make sure that the directory for the Teradata MultiLoad utility, mload, is included in the `PATH` environment variable. This utility is usually installed in the `/usr/bin` directory.

The MultiLoad Utility is not required; SAS/ACCESS provides other options for loading tables. For more information, see the `MULTISTMT` option in your SAS/ACCESS Interface to Teradata documentation. Contact Teradata to obtain the Teradata MultiLoad Utility.

(Optional) Working with Sample Tables

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.

SAS/ACCESS Interface to Vertica

To configure SAS/ACCESS Interface to Vertica, take the following steps:

1. Install a Vertica ODBC driver for UNIX.

   Note: *SAS/ACCESS Interface to Vertica requires an ODBC driver for Vertica. This driver is typically included with a Vertica client installation. SAS recommends installing the most recent version of the driver.*

2. Install an ODBC driver manager, such as UnixODB.
3. Create the `vertica.ini` file.

   Note: *For additional information, refer to Vertica documentation online.*

   - `DriverManagerEncoding`: Always use UTF-8 for UNIX.
   - `ODBCInstLib`: The absolute path to the file containing the ODBC installer library (ODBCInst). For UnixODBC: `libodbcinst.so`
   - `ErrorMessagesPath`: The absolute path to the parent directory that contains the Vertica client driver's localized error message files.

Here is an example:

```
[Driver]
DriverManagerEncoding=UTF-8
ODBCInstLib= /dbi/unixodbc/lib/libodbcinst.so
ErrorMessagesPath= /dbi/vertica/6.1/lib64
```

For Vertica 7.1, the ErrorMessagesPath is slightly different:

```
ErrorMessagesPath=/dbi/vertica/7.1
```

4. Set the environment variable `VERTICAINI` to the location of `vertica.ini`:

   Note: *These examples assume that you saved `vertica.ini` in `/dbi/vertica`.*

   - For Bourne Shell:

     ```
     export VERTICAINI=/dbi/vertica/vertica.ini
     ```

   - For C Shell:

     ```
     setenv VERTICAINI /dbi/vertica/vertica.ini
     ```

5. Set the environment variable `LD_LIBRARY_PATH` to contain the path of the Vertica ODBC driver.

   - For Bourne Shell:

     ```
     export LD_LIBRARY_PATH=/dbi/vertica/6.1/lib64:$LD_LIBRARY_PATH
     ```

   - For C Shell:

     ```
     setenv LD_LIBRARY_PATH /dbi/vertica/6.1/lib64:${LD_LIBRARY_PATH}
     ```

6. (Optional) You can provide connection information in your SAS code, or you can configure DSNs in an `odbc.ini` file similar to the following example:

   Note: *These examples assume that you saved `vertica.ini` in `/dbi/vertica`.*

   ```
   [VTest]
   Description = Vertica DSN
   Driver = /dbi/vertica/6.1/lib64/libverticaodbc.so
   Database = xxx
   Servername = xxx.xxx.xxx.xxx
   UID = xxx
   PWD = xxx
   Port = 5433
   ConnSettings =
   Locale = en_GB
   ```

   Set the `ODBCINI` environment variable to the location of `odbc.ini`

   - For Bourne Shell:

     ```
     export ODBCINI=/dbi/vertica/odbc.ini
     ```

   - For C Shell:

     ```
     setenv ODBCINI /dbi/vertica/odbc.ini
     ```

SAS/ACCESS Interface to Yellowbrick

SAS/ACCESS Interface to Yellowbrick utilizes ODBC driver for PostgreSQL that is included with Base SAS. The PostgreSQL ODBC driver is installed in the following location:

```
<SASHOME>/SASODBCDriversfortheWebInfrastructurePlatformDataServer/
9.4/Driver
```

where <SASHOME> is the path to the SAS home directory.

***Note**: When SAS 9.4M8 hot fixes are applied or when you upgrade to SAS 9.4M9, the location of these libraries on Linux machines is changed. For more information, see ODBC Drivers on Linux on page 35.*

For DSN-less connections, no additional configuration is required.

For DSN connections you must create a serviceable odbc.ini file. You might also need to set the ODBCINI environment variable to point to it.

Here is an example of an odbc.ini file that supports DSN:

```
[postgresql_data_source_name]
Driver=SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/
9.4/Driver/psqlodbcw.so
ServerName=localhost or hostname or IP address
username=user-name
password=password
database=database
port=5432
```

You might also need to create or configure the odbcinst.ini file. Here is an example:

```
[ODBC Drivers]
PostgreSQL=Installed
[PostgreSQL]
Description=ODBC for PostgreSQL
Driver=SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/
9.4/Driver/psqlodbcw.so
```

For more information, refer to the SAS/ACCESS Interface to Yellowbrick chapter in *SAS/ACCESS 9.4 for Relational Databases: Reference* and the documentation provided by Yellowbrick.

Chapter 9: Configuring and Administering SAS In-Database Products

Deployment of SAS In-Database products, including the SAS Embedded Process, requires detailed configuration and administration steps.

For the SAS Embedded Process, follow the steps in the Software Order Email (SOE) that contains only one or more SAS Embedded Processes.

For all other components, follow the steps in your SOE and QuickStart Guide to perform the initial deployment. Then refer to the chapter of the *SAS In-Database Products: Administrator's Guide* for your particular database. This chapter contains instructions on how to install and configure the in-database deployment packages for your specific database.

These in-database deployment packages are required to support the following actions:

- using the SAS Scoring Accelerator
- using the SAS In-Database Code Accelerator
- using the SAS Data Loader for Hadoop
- reading and writing data to a Hadoop Distributed File System in parallel for High-Performance Analytics
- publishing formats
- using the `SAS_PUT` function
- using any other software that requires the SAS Embedded Process

When you have completed the instructions described there, your software is ready for use.

The *SAS In-Database Products: Administrator's Guide* and other documentation can be accessed from the SAS In-Database Technologies Learn & Support Page.

Configuring and Administering SAS Data Loader for Hadoop

Important: *Starting with SAS 9.4M9, the procedures for deploying SAS Data Loader for Hadoop have changed. Several manual steps are required in order to configure the drivers and run a configuration script. Full details are available in the SAS 9.4 Hadoop Configuration Guide for Base SAS and SAS/ACCESS.*

SAS Data Loader for Hadoop is deployed in a two-part process that uses two SOE messages.

The first part of this process deploys the SAS In-Database Technologies for Hadoop components. This collection of technologies includes the following products and components:

- SAS/ACCESS Interface to Hadoop
- SAS Embedded Process for Hadoop
- SAS In-Database Code Accelerator for Hadoop
- SAS Data Quality Accelerator for Hadoop
- SAS Quality Knowledge Base (QKB)
- Other components

For SAS Data Loader for Hadoop to operate, some of these products and components require additional configuration. A system administrator installs and configures these SAS products and components.

SAS recommends configuring the following products and components in the following order:

1. **SAS/ACCESS Interface to Hadoop** – For information about SAS/ACCESS configuration, see the *SAS 9.4 Hadoop Configuration Guide for Base SAS and SAS/ACCESS*.
2. **SAS Embedded Process for Hadoop** – The SAS Embedded Process is included in an in-database deployment package. This package must be deployed on the Hadoop cluster. For information about deploying the SAS Embedded Process, see the *SAS In-Database Products: Administrator's Guide*, accessible from the SAS In-Database Technologies Learn & Support Page.
3. **SAS Data Quality Accelerator for Hadoop** – If you use the Software Deployment Manager, the SAS Data Quality Accelerator is installed silently when you start the in-database package deployment. You can also deploy this component manually. For more information, see the *SAS In-Database Products: Administrator's Guide*, accessible from the SAS In-Database Technologies Learn & Support Page.
4. **SAS Quality Knowledge Base**: The SAS QKB component can be installed using the SAS Deployment Wizard, or you can install it manually. For information about this component, see the *SAS In-Database Products: Administrator's Guide*, accessible from the SAS In-Database Technologies Learn & Support Page.

   ***Note**: The SAS In-Database Products: Administrator's Guide also describes tasks for configuring your Hadoop cluster and security.*

The Data Loader for Hadoop vApp

The second part of the deployment process for SAS Data Loader for Hadoop sets up the SAS Data Loader for Hadoop vApp. This part of the process requires you to use a separate software order (that is, a separate Software Order Email) to install and configure the SAS Data Loader vApp on a client machine. As part of this client configuration, you must request Hadoop connection and security information from the system administrator who installed SAS In-Database Technologies for Hadoop.

The *SAS Data Loader for Hadoop: vApp Deployment Guide* contains information about how to install and configure the vApp on a client machine to enable the SAS Data Loader for Hadoop.

For more information, see the documentation for the SAS Data Loader for Hadoop, accessible from the following web page: `https://support.sas.com/en/software/data-loader-for-hadoop-support.html`.

Chapter 10 – Post-Installation Configuration for SAS/ASSIST

This chapter describes how to add a master profile to SAS/ASSIST software. You can use a master profile to override the default SAS settings. 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 information on the profile options, refer to the *SAS/ASSIST Software System Administrator's Guide.*

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 directory to which all users of SAS/ASSIST will have read access.

   All users with write access to this directory will automatically have write access to the master profile in SAS/ASSIST. Select a name that conforms to the naming conventions of your installation. The name of this new directory must be stored in an entry in the `SASHELP` library. This requires that you have write access to the `SASHELP` library.

   On line 1 of the `Program Editor` window of the SAS Display Manager System, type the physical pathname of the master profile directory. Execute the `Save` command to store this pathname in the `SASHELP.QASSIST` catalog. Save it as `SASHELP.QASSIST.PARMS.SOURCE`. The location of the master profile will now be known by SAS/ASSIST.

2. Create the master profile.

   The first time SAS/ASSIST is started, a master profile is created if `SASHELP.QASSIST.PARMS.SOURCE` contains the name of an existing physical pathname, and the person who starts SAS/ASSIST has write access to this physical pathname.

3. Customize the master profile by starting SAS/ASSIST and selecting:

   ```
   Setup, then
   Profiles, and then
   Master/group ...
   ```

   If you have write access to the SAS library containing the master profile, you can specify default values. New users will use these default values when they start SAS/ASSIST.

   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
```

By setting the default values in the master profile, you can control if users should use the New or Old style of SAS/ASSIST software. In addition, there are many other profile options. For more information on these options, refer to the *SAS/ASSIST Software System Administrator's Guide*.

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 status.

   Select `Setup...Master/Group`

   then `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`. This 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. If the value name starts with `&`, it is a macro variable; otherwise, it is an environment variable, which is set before starting SAS Foundation 9.4.

Chapter 11 – Post-Installation Configuration for SAS/CONNECT

TCP/IP is the access method supported for UNIX environments and their derivatives. For information about the access methods supported by other systems, refer to *the SAS/CONNECT 9.4 User's Guide*.

Storing and Locating SAS/CONNECT Script Files

SAS/CONNECT ships several sample script files that are used to establish a connection to a remote SAS session. The `SASSCRIPT` configuration option points to the location of the SAS/CONNECT script files. The `SASSCRIPT` option is used by SAS/ASSIST software and can be used by user-written SCL applications.

The script files are installed into the `!SASROOT/misc/connect` directory by default. The following line has been included in the `sasv9.cfg` file to define the default script file location:

```
-SASSCRIPT !SASROOT/misc/connect
```

If you want to move the script files to another directory, you must edit the `sasv9.cfg` file and update the `SASSCRIPT` option with the new directory location.

Configuring the SAS UNIX Spawner Program

The SAS UNIX Spawner is stored in the `!SASROOT/utilities/bin` directory and can be executed manually from the `!SASROOT/utilities/bin` directory at any time. For complete documentation on the UNIX spawner and the supported options, see the *SAS/CONNECT 9.4 User's Guide*.

Chapter 12 – Post-Installation Configuration for SAS/GRAPH

Loading SAS Fonts to Your X Display Server

Many SAS/GRAPH procedures and devices now support ODS styles in all destinations, including the LISTING destination. By default, all colors, fonts, symbols, and graph sizes are derived from the current style. The default fonts in these styles are the TrueType fonts provided by SAS. Devices that use FreeType rendering are able to find these fonts by default and render them in an environment without a valid X display available. For devices like XCOLOR that use host-rendering, the fonts must be registered with the display in order for them to work. You may override the default font setting by using the `FTEXT` option on the `GOPTIONS` statement or by creating a modified style sheet. However, SAS recommends making the TrueType fonts available to the display device to take advantage of their benefits.

Refer to your vendor user documentation for your X display server for the instructions for making the SAS fonts available to it. SAS fonts are located at `$SASROOT/misc/fonts`.

Making System Fonts Available to SAS

One of the main advantages of using FreeType rendering is that TrueType and other hardware fonts that produce high quality text are available in an environment that lacks an X display. The graphics devices that use FreeType rendering will only recognize fonts that have been registered in SAS.

To register additional fonts to SAS, including system or display fonts, use the `FONTREG` procedure to update the SAS registry to include these fonts. For more information about the use and syntax of the `FONTREG` procedure, see the appropriate chapter in the *Base SAS 9.4 Procedures Guide*, available at https://go.documentation.sas.com/?docsetId=proc&docsetTarget=p1vagzo4p5p417n16wect9d85cu7.htm&docsetVersion=9.4&locale=en.

Chapter 13 – 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`. The "What's New" page at this website lists any recent changes to the product or documentation.

Overview

All SAS/IntrNet installations are made up of two components:

1. The SAS/IntrNet server (also referred to as the Application Server). This is where SAS Foundation is installed.
2. 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 from 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 configurations 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. Check "CGI Tools for the Web Server" in the "Select Products to Install" dialog box.
Test the Broker
Configure a Socket Service
Start the Socket Service

Type B Installation Steps
Confirm that the web server software (IIS, Apache, etc.) is on the system machine where you will install CGI Tools.
On your application server system machine, start your SAS installation. Clear the "CGI Tools for the Web Server" option in the "Select Products to Install" dialog box.
On your web server system machine, start your SAS installation. Clear all products except "CGI Tools for the Web Server" in the "Select Products to Install" dialog box. (Optional) Check the IntrNet Monitor or Connect Drivers.
Test the Broker
Configure a Socket Service
Start the Socket Service

Test 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 your *QuickStart Guide* for a description of how to start 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. Check "CGI Tools for the Web Server" in the Select Products to Install dialog box.

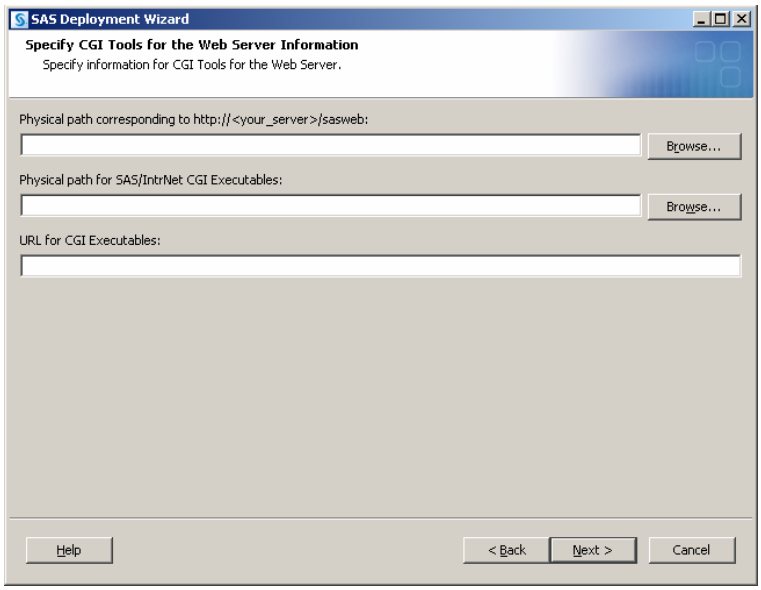
If you are performing a Type B installation (as described in the "Overview" above), do **both** of the following:

- Install the SAS software on the SAS System side, and clear the **CGI Tools for the Web Server** and **SAS/GRAPH Java Applets** options in the **Select Products to Install** dialog box.
- Start the SAS software installation on the web server and check "**CGI Tools for the Web Server**" and "**SAS/GRAPH Java Applets**" in the **Select Products to Install** dialog box. SAS/IntrNet Monitor and SAS/CONNECT Driver for Java are optional selections. Clear everything else.

CGI Tools Installation Dialog Boxes

The following dialog boxes 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 box shown below. Customize the entries according to 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 instruct SAS where your web server is located.

Physical path corresponding to http://*your_server*/sasweb:

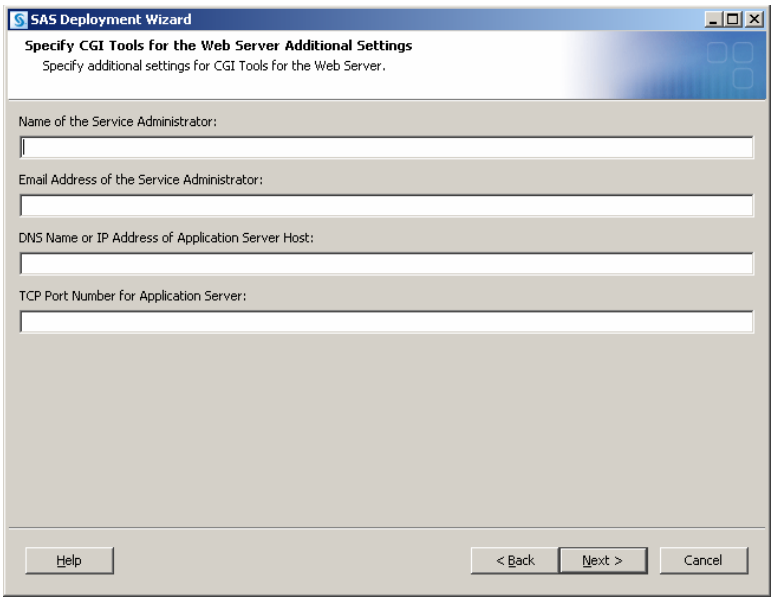
IIS:	`C:\Inetpub\wwwroot\sasweb`
Apache (Windows):	`C:\program files\Apache Software Foundation\Apache2.2\ htdocs\sasweb`
Apache (UNIX):	`/usr/local/apache2/htdocs/sasweb`

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. The `broker.cfg` file is a text file that can be edited after the installation is complete.

Name of the Service Administrator:

(Optional) Enter the name of the administrator (for example, John Doe).

Email Address of the Service Administrator:

(Optional) Enter the email address of the administrator (for example, `NetAdmin@comp.com`).

DNS Name or IP Address of Application Server Host:

Enter the DNS name or IP address of the application server host where SAS Foundation is located.

TCP Port Number for Application Server:

The customary default port number is 5001, but you can use any valid available port on your system between 256 and 65535.

Installing CGI Tools and SAS Foundation on Machines with Different Operating Systems

Your SAS Foundation operating system might be different from that of your CGI Tools. For example, your SAS Foundation might be installed on a Windows system and your CGI Tools might be installed on a UNIX system. The CGI Tools installation from the SAS Deployment Wizard detects the destination operating system and installs the appropriate operating system-specific software.

You must make the SAS Software Depot available to the installer on the destination CGI Tools system. Select one of the following methods, based on the facilities at your site:

- Launch the setup from a SAS Software Depot residing on a remote system. A cross-platform file access method, such as NFS or SAMBA, might be required to connect the two systems.
- Create media from an existing depot using the SAS Deployment Wizard and use it on the host machine.

This process is described 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 a browser on the web server machine and enter `http://localhost`. A web page is displayed if the web server is running.

If you do not see a web page, debug or reinstall your web server before continuing.

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:

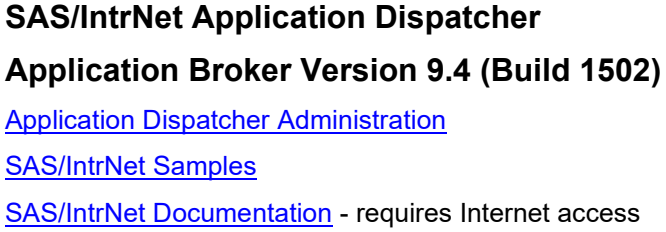
IIS - `http://your_webserver/scripts/broker.exe`

Apache - `http://your_webserver/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 similar to the following:



If you do not receive this page, 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 the value of the **Physical path for SAS/IntrNet CGI Executables** parameter during the CGI Tools installation.

Configure a Socket Service

To create the default service for an Application Server running in a UNIX environment, take the following steps:

1. From a system prompt, submit the following command:

   ```
   SASHOME/SASDeploymentManager/9.4/sasdm.sh
   ```

   where *SASHOME* is the path to the SAS home directory. The **Choose Language** window appears. Select the desired language and click **OK**.

2. The SAS Deployment Manager displays the **Select SAS Deployment Manager Task** window. Under **SAS/IntrNet Service Tasks** select **Create Socket Service** and click **Next** to continue.

3. The **Specify Service Name** window opens. The default value for the Service Name field is *default*. Create this as your first service because this is what is used when you run the samples. Click **Next** to continue.

4. The **Specify Service Directory** window opens. The SDM selects a default service root directory based on the location that you chose for user files when you installed SAS software. This default location is recommended for most users, although you can use the **Browse** button to select a different directory. Remember this directory because the start.sh script to start the Application Server will be created in it. Click **Next** to continue.

5. The **Specify Service Ports** window opens. Type the TCP/IP port number that you reserved for the default Application Dispatcher service. Click **Next** to continue.

6. The **Specify Administrator Password** window opens. A password is not necessary for the default service. You can add an administrator password later if you use this service for production applications. Click **Next** to continue.

7. The **Summary** window states, **Stage 1: Create Socket Service**. If you want to go back and change any of the values you have entered previously, click the **Back** button to step backward through the steps. If you are satisfied that the information you have entered is correct, click **Start**.

8. The **In Progress** window displays while SDM creates the service.

9. The **Deployment Complete** window displays when the task has completed. If the service is created correctly, a green checkmark appears next to the **1. SAS/IntrNet** under **Stage 1: Create Socket Service**. If there was a problem a yellow exclamation or red X will appear and you should check the log for a description of the problem. The logs reside in `SASHOME/SASFoundation/9.4/misc/intrnet`.

10. The configuration utility creates a `start.sh` file to start the default Application Server. Change to the service directory path and start the server by submitting the following command:

    ```
    ./start.sh
    ```

Starting the Socket Service

Change to the service directory path and start the server by submitting the following command: `./start.sh`

Testing the Socket Service

1. To verify that the service was installed and started correctly, point your web browser to this URL:

   Windows:

   IIS - `http://your-webserver/scripts/broker.exe`

   Apache - `http://your-webserver/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 the Application Broker to a different directory. You should see the following web page:

   ### SAS/IntrNet Application Dispatcher

   #### Application Broker Version 9.4 (Build 1502)

   Application Dispatcher Administration

   SAS/IntrNet Samples

   SAS/IntrNet Documentation - requires Internet access

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 the **Application Dispatcher Administration** link and then click the **ping** link under the **SocketService default** heading. If the ping is successful, it returns the following:

   ```
   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://your-webserver/scripts/broker.exe?_service=default&_program=sample.webhello.sas`

   Apache - `http://your_webserver/cgi-bin/broker.exe?_service=default&_program=sample.webhello.sas`

   Other hosts:

   `http://your-webserver/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://your-webserver/scripts/broker.exe?_service=default&_program=sample.webhello.sas&_debug=131`

Apache - `http://your_webserver/cgi-bin/broker.exe?_service=default&_program=sample.webhello.sas&_debug=131`

Other hosts:

`http://yourwebserver/cgi-bin/broker?_service=default&_program=sample.webhello.sas&_debug=131`

Save the log screen on the browser to help SAS Technical Support with troubleshooting.

Configure Additional Services

This chapter only describes how to configure a simple default Application Dispatcher service. For many reasons, you may 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 14 – Post-Installation Configuration for Encryption and SAS/SECURE

***Important**:* *Starting with SAS 9.4M8, SAS Foundation servers use the cryptographic libraries that are provided and installed on the operating system to provide encryption for data in motion. The operating system's OpenSSL libraries might not provide all the encryption algorithms that were previously available in SAS, mainly because they were deemed insecure. For example, with OpenSSL version 3 libraries, some algorithms are contained in a "legacy" encryption provider, which might not be included in newer operating systems, such as Red Hat Enterprise Linux 9.x. Users might see warnings, which can safely be ignored.*

For more information, see Encryption in SAS 9.4.

Changes in SAS 9.4M9

Starting with SAS 9.4M9, you can use the SAS Deployment Wizard to configure and enable TLS for the following components:

- SAS Web Application Server
- SAS JMS Broker
- Cache Locator

For more information, see (SAS 9.4M9) Use TLS For Internal Connections in *SAS Intelligence Platform: Installation and Configuration Guide*.

Starting with SAS 9.4M9, PKCS#12 certificate files (file extension .p12) are now provided. The trusted certificate files are trustedcerts.pem and trustedcerts.p12. Previously, JKS-formatted certificate files (file extension .jks) were provided.

Changes in SAS 9.4M8

SAS 9.4 through SAS 9.4M7 included SAS/SECURE with Base SAS in order to provide strong encryption for all deployments. For all maintenance releases of SAS 9.4 prior to SAS 9.4M8, updated versions of OpenSSL were provided and updated as needed through hot fixes.

Starting with SAS 9.4M8, operating-system cryptographic libraries are used instead to provide encryption for SAS Foundation. All encryption for data in motion and data at rest is provided by the cryptographic libraries that are installed on the operating system where SAS 9.4M8 or later is deployed. SAS no longer delivers the cryptographic libraries that were used by SAS Foundation. For more information, see "Cryptographic Library Support Starting with SAS 9.4M8" in *Encryption in SAS*.

Encryption for SAS Foundation

SAS/SECURE is included with Base SAS and provides strong encryption support. In SAS 9.4M8 and later, SAS Foundation servers also use the cryptographic libraries that are available from the operating systems that are supported by SAS.

Each SAS-supported cipher suite might not be available from all operating systems. When this issue is encountered, use a cipher suite that is supported by both SAS and that operating system, or install a third-party SSL provider for use by SAS.

Encryption for SAS Web Infrastructure Data Server

SAS delivers the OpenSSL libraries that are used by the ODBC drivers for SAS Web Infrastructure Data Server for TLS. On Linux, the two required OpenSSL libraries are libcrypto.so.3 and libssl.so.3. When SAS 9.4M8 hot fixes are applied or when you upgrade to SAS 9.4M9, the location of the OpenSSL libraries is changed from

SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/9.4/Driver

to

SASHOME/SASODBCDriversfortheWebInfrastructurePlatformDataServer/9.4/Driver/**lib64**

This directory change applies only to Linux operating systems. For more information, see ODBC Drivers on Linux on page 35.

On AIX, if you have installed multiple SAS/ACCESS products, make sure that the operating-system library path is set before the default path for the PostgreSQL ODBC driver. More than one SAS/ACCESS product might rely on ODBC drivers. Setting the `LIBPATH` in the correct order enables SAS/ACCESS products to locate these same encryption libraries. For more information, see Enabling Co-location of Multiple SAS/ACCESS Products on AIX on page 35.

SAS/SECURE Client for Windows

The SAS/SECURE components required by Windows clients can be installed by running SAS Deployment Wizard. SAS/SECURE includes client components that non-SAS system client applications can use to communicate with a SAS server in a secure environment. To apply encryption between a non-SAS system client and a SAS server, you must install the SAS/SECURE client components on the client machine.

Note: *This installation is not required if the SAS system is your client. The SAS system installs the components that it requires as part of the SAS system installation process.*

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

The SAS/SECURE components that are needed by Java clients can be installed by running the SAS Deployment Wizard. The SECUREJAVA folder contains two JAR files that enable Java clients to use the CryptoAPI algorithms:

- `sas.rutil.jar` - should be copied to the location where the client you are running gets started.
- `sas.core.jar` - included in case you do not already have one; however, this file will most likely not be needed.

FIPS 140-2 Support

The Federal Information Processing Systems (FIPS) 140-2 standard defines the security requirements for cryptographic modules. The FIPS 140-2 standard is detailed in the following document: http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

The ENCRYPTFIPS option is required to enable FIPS 140-2 in a SAS deployment. This option is added to SAS/CONNECT spawners, IOM spawners, and Base SAS so that the communication encryption supports FIPS 140-2. With ENCRYPTFIPS, the NETENCRALG option must be set to AES or 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.*

Prior to SAS 9.4M8, the OpenSSL libraries provided by SAS were not FIPS-capable. However, you can download FIPS 140-2 certified OpenSSL libraries and replace the ones that were previously delivered. For more information, see Process to Build FIPS 140-2 Capable OpenSSL on UNIX and Linux.

Starting with SAS 9.4M8, SAS no longer delivers OpenSSL libraries for SAS Foundation servers. SAS instead uses supported cryptographic libraries that are installed on the operating system to provide encryption for data in motion. FIPS should be enabled at the operating-system level first. You must then use the ENCRYPTFIPS SAS system option to place SAS code into FIPS mode. For more information, see How to Use FIPS 140-2 Capable OpenSSL Libraries with SAS 9.4M8 or Later on UNIX and Linux.

For more information about FIPS and encryption in general, refer to *Encryption in SAS 9.4*.

Chapter 15 – Post-Installation Configuration for SAS/SHARE

For more information about SAS/SHARE, refer to the *SAS/SHARE 9.4: User's Guide*.

User Authentication

The steps that are described in "Chapter 3 – Configuration for User Authentication and Identification on page 5 are required. These procedures enable SAS/SHARE software to authenticate a client's identity and check a client's authority to access resources.

System Configuration for the TCP/IP Communications Method (Optional)

SAS recommends that each SAS/SHARE server that runs on a network node be defined as a service in the file `/etc/services` or `/etc/inet/services` on that node. Each entry in this file associates a service name with the port number and protocol used by that service. An entry for a SAS/SHARE server has the following form:

```
server name       port number/tcp   # comments
```

The server name must be one to eight characters in length. The first character must be a letter or underscore; the remaining seven characters can include letters, digits, underscores, the dollar $ sign, or the at @ sign. The port number must be above 1024, as any port equal to or less than 1024 is reserved.

An entry for a server whose name is `MKTSERV` might resemble the following:

```
mktserv     5000/tcp    # SAS/SHARE server for Marketing and Sales
```

The server name is specified with the `SERVER=` option in the `LIBNAME` statement, in the `OPERATE`, and in the `SERVER` procedure. If a server name is not defined in the services file, you must specify "`__port#`" which is two consecutive underscores followed by the port number (for example, `server=__5012)`.

Client 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 Downloads site at `http://support.sas.com/downloads/browse.htm?fil=&cat=87`. These components 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 may 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/SHARE 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.

NLS Information

Sites that develop or support international applications that use SAS/SHARE software should refer to Chapter 7, "Post-Installation Configuration for National Language Support (NLS)" on page 30.

Chapter 16 – Using Host Sort Routines

This chapter provides instructions for making host sort routines available to SAS 9.4. The only supported host sort routine is SyncSort. To use host sort routines with SAS 9.4, complete the following steps:

1. Install the host sort library on your system by following the instructions provided by the vendor. Ensure that the host sort routine works outside of SAS 9.4.
2. Make the host sort library available to SAS 9.4 by following the instructions in the following section, "Making Host Sort Routines Available."
3. Submit an options statement in a SAS session to specify the host sort routine by following the instructions in the section "Using Host Sort Routines in a SAS Session."

Note: *For information on using host sort routines in a SAS session once they are available, refer to the SAS 9.4 Companion for UNIX Environments.*

Making Host Sort Routines Available

This section describes the system-specific instructions for making host sort routines available to SAS 9.4.

For AIX

Set the environment variable `$LIBPATH` to the directory containing the host sort library. For example, if the directory is `/usr/local/syncsort/lib`, then add these lines to both `!SASROOT/bin/sasenv_local` and `!SASROOT/bin/sasenv_local.ksh`:

```
LIBPATH=/usr/local/syncsort/lib:$LIBPATH
export LIBPATH
```

Add this line to `!SASROOT/bin/sasenv_local.csh`:

```
setenv LIBPATH /usr/local/syncsort/lib:$LIBPATH
```

For HP-UX, Linux, and Solaris

Set the environment variable `$LD_LIBRARY_PATH` to the directory containing the host sort library. For example, if the directory is `/usr/local/syncsort/lib`, add the following lines to both `!SASROOT/bin/sasenv_local` and `!SASROOT/bin/sasenv_local.ksh`:

```
LD_LIBRARY_PATH=/usr/local/syncsort/lib:$LIBPATH
export LD_LIBRARY_PATH
```

Add this line to `!SASROOT/bin/sasenv_local.csh`:

```
setenv LD_LIBRARY_PATH /usr/local/syncsort/lib:$LIBPATH
```

Using Host Sort Routines in a SAS Session

Note: *The options statements throughout this section specify the syntax to submit to the SAS System. You can also specify these options as command line options and options in the sasv9.cfg file. Refer to the SAS Companion for UNIX Environments for more information on setting options.*

Use the `SORTNAME` option to instruct the SAS System which host sort routine should be used. Submit one of the following options statements in a SAS session:

- To use SyncSort (the default):

  ```
  OPTIONS SORTNAME=SYNCSORT;
  ```

- To use CoSORT:

  ```
  OPTIONS SORTNAME=COSORT;
  ```

Once the host sort routine is available, use the `SORTPGM=HOST` or `SORTPGM=BEST` options statements to tell the SAS System when to use the host sort routine.

Submit one of the following options statements in a SAS session:

- `OPTIONS SORTPGM=HOST;`

  Tells the SAS System to always use the host sort routine made available.

- `OPTIONS SORTPGM=BEST;`

  Tells the SAS System to select the best sorting method in a given situation, the SAS System sort or the host sort.

  Two options define how the SAS System selects the "best" sort algorithm. The following examples use the syntax of an options statement that must be submitted to the SAS System:

  - `-sortcut` *`n`*, where *`n`* specifies a number of observations.

    ```
    OPTIONS SORTPGM=BEST SORTCUT=500;
    ```

    `-sortcut` tells the SAS System to select the host sort routine if the number of observations is greater than the number you specify, and to use the SAS System sort if the number of observations is equal to or less than the number specified.

  - `-sortcutp` *`size`*`[kKmM]`, where *`size`* specifies a file size in either kilobytes or megabytes.

    ```
    OPTIONS SORTPGM=BEST SORTCUTP=40M;
    ```

    `-sortcutp` tells the SAS System to select the host sort routine if the size of the data being sorted exceeds the size you specify, and to use the SAS System sort if the size of the data is equal to or smaller than the size you specify.

    If these options are not defined or these options are set to 0, the SAS System selects the SAS System sort routine. If you specify both options and either condition is met, the SAS System selects the host sort routine.

You can change the work directory used for temporary sort files by using the option `sortdev` *`dir`*, where *`dir`* is the directory where you want the temporary files to be created. For example, submit the following statement if you want the temporary files to be created in `/tmp`:

```
OPTIONS SORTPGM=BEST SORTCUT=500 sortdev="/tmp";
```

You can specify the host sort option `sortanom t` to print timing and resource information to the SAS log after each phase of a sort. The following is an example of this option:

```
OPTIONS SORTPGM=HOST SORTANOM=t;
```

You can specify the host sort option `sortanom v` to print to the SAS log the arguments passed to the sort, which might be useful for tuning or debugging:

```
OPTIONS SORTPGM=HOST SORTANOM=v;
```

You can attempt to increase your sort performance by increasing the values of the `sortsize` and `memsize` SAS options. However, make sure that `sortsize` is at least 4M less than `memsize`.

You can see other SAS performance statistics in the SAS log using the `FULLSTIMER` option:

```
OPTIONS FULLSTIMER;
```

support.sas.com

SAS is the world leader in providing software and services that enable customers to transform data from all areas of their business into intelligence. SAS solutions help organizations make better, more informed decisions and maximize customer, supplier, and organizational relationships. For more than 30 years, SAS has been giving customers around the world The Power to Know®. Visit us at **www.sas.com**.