CHAPTER 25

Windows: DECnet Access Method

SAS Support for DECnet on Windows

Note: The DECnet communications access method can be used with Windows NT, Windows 95, Windows 98, and Windows 32s.

However, beginning with Version 7, the Windows 32s platform is not supported. Information about Windows 32s is included here for Version 6 users. △

Tasks That Are Common to SAS/CONNECT and SAS/SHARE

System Administrator or User

To use the DECnet access method with a Windows host for SAS/CONNECT and SAS/SHARE, perform these tasks:

1. Verify that you have met all your site and software requirements.
2. Verify that you know how to set options in SAS software.
3. Set the SAS/CONNECT and SAS/SHARE options that you want.

System and Software Requirements for SAS/CONNECT and SAS/SHARE

Ensure that the following conditions have been met:

1. The DECnet software has been installed at your site.
2. SAS has been installed on both the local and remote hosts.
3. For Version 3.51 of Windows NT, Digital Equipment Corporation Pathworks for Windows NT Version 5.1 or a subsequent version has been installed at your site.
4. For Version 4.0 of Windows NT, Digital Equipment Corporation Pathworks 32 has been installed at your site.
5. For Windows 95 and Windows 98, the Microsoft Winsock Version 2.0 and Digital Equipment Corporation Pathworks 32 have been installed at your site.
6. For Windows 32s, Digital Equipment Corporation Pathworks for DOS and Windows Version 5.1 or subsequent version have been installed at your site.

Setting SAS Options and Variables

You may set specific options in SAS to establish the connections that you want with SAS/CONNECT and SAS/SHARE when using the DECnet communications access method.

You may specify an option in any of several forms, as follows:

- in an OPTIONS statement in a SAS session or in an AUTOEXEC file:

  OPTIONS SET=*variable-name value*

  Example:

  ```
  options set=sassecur _secure_;
  ```

- in a SAS configuration file or at a SAS invocation:

  -SET *variable-name value*

  Example:

  ```
  -set sassecur _secure_
  ```

- as a SAS macro variable:

  %LET *variable*=*value*;

  Example:

  ```
  %let sassecur=_secure_;
  ```

□ as a DOS operating system environment variable:

SET *variable-name*=*value*

Example:

```
set sassecur=_secure_
```

Values for these options can contain up to eight characters, consisting of alphanumeric characters, the percent sign (%), the dollar sign ($), the pound sign (#), the at sign (@), and the underscore (_).

If you set multiple forms of the same option, the order of precedence follows:

SAS macro variable

OPTIONS statement

AUTOEXEC file

SAS invocation

SAS configuration file

DOS environment variable.

Setting Security for SAS/CONNECT and SAS/SHARE

For SAS/CONNECT, you must supply identifying information to sign on without a script to a remote host running a spawner program. A SAS/SHARE server, running secured, requires identification from each connecting client. The next two sections outline the version-specific methods for specifying client identification for SAS/CONNECT and SAS/SHARE.

Providing Client Identification in a Version 8 Session

Note: In the Windows environment, SAS/SHARE server security is supported on the Windows NT platform only. △

In Version 8, you provide client identification to a SAS/CONNECT remote host or a SAS/SHARE server using the USER= and PASSWORD= options. These options are valid in the following statements:

SIGNON

RSUBMIT

LIBNAME

PROC SQL

Connect to Remote

PROC OPERATE

(in the PROC statement)

set server

stop server

quiesce server

start server

display server

Specifying client identification in the SASSECUR= option is still accepted but is not recommended in Version 8. The USER= and PASSWORD= options take precedence over the client SASSECUR= option when both are specified. For example, a SAS/SHARE client's execution of a LIBNAME statement with values assigned to the USER= and

PASSWORD= options would override a SASSECUR= option setting in the same client SAS session.

CAUTION:

In order to make a SAS/SHARE server secured, the SASSECUR= option must be set at a SAS/SHARE server that can run on any host. △

Here is the syntax and definitions for these options:

USER | **USERNAME** | **USERID** | **UID**=*username* | _PROMPT_

PASSWORD | **PASSWD** | **PASS** | **PWD** | **PW**=*password* | _PROMPT_

Specifying these options allows a user on the local host whose username and password have been verified to access the remote host.

username
is a valid userid for the remote host and is thus host-dependent in form. If the value contains blanks or special characters, it must be enclosed in quotes. On Windows NT only, the username can also include the domain name, which locates the specified username in a domain.

password
is the password, if any, required for authentication of the supplied username. This value will not be echoed in the SAS log. If the value contains blanks or special characters, it must be enclosed in quotes.

PROMPT
specifies that the SAS System prompts the client for *username* and *password*.

Note: The values provided when prompted must NOT be quoted. △

Specifying USER=_PROMPT_ and omitting the PASSWORD= specification will cause SAS to prompt you for both userid and password.

This is especially useful for allowing the SAS statements containing the USER= and PASSWORD= options to be copied and otherwise effectively reused by others.

For SAS/SHARE, the values supplied for the USER= and PASSWORD= options are valid for the duration of the remote host connection. Additional accesses of the remote host while the connection to that host is still in effect do not require re-supplying of the USER= and PASSWORD= options. For example, while the first connecting library assign to a SAS/SHARE server may require specification of the options, subsequent assigns to the same server will not need specification of these options as long as the original connection is in effect. A subsequent re-connect to the same server or connect to a different server would require re-supplying of the USER= and PASSWORD= options.

Here is a Version 8 example for SAS/SHARE:

```
libname test 'prog2 a' user=joeblue password="2muchfun" server=share1;
```

For SAS/CONNECT, these values are valid until SIGNOFF.

Here is a Version 8 example for SAS/CONNECT:

```
signon rmthost user=joeblack password=born2run;
```

As a security precaution, PASSWORD= field entries echoed in the log are replaced with Xs. If _PROMPT_ was specified for entering the password, the entry would not be displayed on the screen as it is typed.

Providing Client Identification in a pre-Version 8 Session

CAUTION:

Windows NT only SAS/SHARE server security is supported on the Windows NT platform only. △

You must set the SASSECUR option in order to pass a remote host user name and password to a SAS/SHARE server for verification. After the user name and password have been verified, the connection to the SAS/SHARE server can proceed. Values for SASSECUR are

SASSECUR _NONE_ | _PROMPT_ | *username.password* | _SECURE_

NONE

must be set at the SAS/SHARE client. This is the default.

Setting _NONE_ does not establish secure sessions for connecting SAS/SHARE clients.

PROMPT

must be set at the SAS/SHARE client.

PROMPT specifies that SAS prompt the user for user name and password information. When prompted for a password, the input field is not displayed. Choosing to prompt for user name and password provides more security than assigning the user name and the password to the system option.

userid.password

must be set at the SAS/SHARE client.

This value specifies both the user name and the password. Assigning the user name and password directly to the SASSECUR option at the SAS/SHARE client may inadvertently publicize this information and compromise the security of the SAS/SHARE server. Assigning the value to the option in a file allows anyone to read it.

SECURE

must be set at the SAS/SHARE server on a Windows NT host only.

The _SECURE_ value for the SASSECUR option requires a SAS/SHARE client to supply a valid user name and password to the remote host or the remote host on which the server is running in order to allow client access to the server.

Specify the SASSECUR option before you create a server.

Examples:

```
%let SASSECUR=_NONE_;
%let SASSECUR=_PROMPT_;
%let SASSECUR=bass.time2go;
%let SASSECUR="apex\bass.time2go";
%let SASSECUR=_SECURE_;
```

See “Setting SAS Options and Variables” on page 366 for examples of the forms that you can use to specify the SASSECUR option.

SAS/CONNECT

Local Host Tasks

User or Applications Programmer

To connect a Windows local host to a remote host, perform these tasks at the local host:

1. Be aware of security considerations for Windows hosts.
2. Specify the communications access method.
3. Specify a remote host name.
4. Sign on to the remote host.

Security Considerations for Windows

Windows NT Security Considerations

Windows NT is a secure operating system, requiring that you supply a valid user name and password in order to log on. When you connect with the DECnet access method, you supply this information by including Access Control Information (ACI).

When connecting to a remote system, you can supply ACI information explicitly with the REMOTE= option. However, doing so requires that you either type your user name and password each time you connect or that you enter permanently your user name and password into the SAS program.

If proxy access is enabled on both the local and the remote nodes, and the user name is valid on both systems, DECnet uses your user name for the default ACI. Any connection you make logs you onto the remote system with the same user name that you used to log on to the local system.

If proxy access is not enabled, or you want to connect to the remote system as a different user, you must supply ACI with the value of the REMOTE= option.

You also can configure a remote Windows NT system to supply a user name for connection requests that do not contain ACI. It is possible to force a system to not send ACI. However, it is more likely that you will receive a connection request without ACI from a Windows 95, Windows 98, or Windows 32s system.

See “Specifying the Remote Host Name” on page 371 for instructions about defining a default user name on Windows NT. The user name must be valid on the Windows NT system on which it is defined.

Note: Defining a default user name for incoming requests can compromise security. Because anyone can connect using the default id, it should not be a privileged user name, and it should be allowed to have access to sensitive data. △

Windows 95, Windows 98, and Windows 32s Security Considerations

Windows 95, Windows 98, and Windows 32s are not secure operating systems. Because it has no concept of user name, DECnet cannot use a user name as the default ACI. You can supply ACI in the REMOTE= option just as you can with Windows NT.

However, the Pathworks software for Windows 95, Windows 98, and Windows 32s allows you to enter a default user name and password for each remote node that you want to connect to. The Network Control Program (NCP) node database contains the node definitions for remote nodes. The NCP DEFINE NODE command lets you associate a user name and a password for each node. This user name and password must be valid on the node for which it is defined.

If you do not supply ACI when you connect to a remote system, the DECnet access method uses the default ACI that is stored in the NCP node database. If no default ACI is defined for the node, then no ACI is sent. The remote system must then be configured to supply a default incoming user name; otherwise, the connection will fail.

You declare the host name, user name, and password of the remote host at the local host by using the following syntax:

```
DEFINE NODE node-address NAME node-name
       USER access-information
       PASSWORD access-information
```

Example:

```
DEFINE NODE 1.300 NAME RMTHOST USER bass
       PASSWORD time2go
```

See your Pathworks documentation for more information about defining a Windows 95, a Windows 98, or a Windows 32s default user name.

To connect to a Windows 95, a Windows 98, or a Windows 32s remote host, supply the host name as the value for the REMOTE= option.

Note: Defining a default ACI for a remote node can compromise security. Because anyone using the Windows 95, the Windows 98, or the Windows 32s system can get access to that user name on the remote node, the default user name should not be a privileged id, and it should not be allowed to have access to sensitive data. To permit access to such data, SAS should prompt the user for a user name and a password each time the user tries to connect to the remote node. △

Specifying the DECnet Communications Access Method

You must specify the DECnet communications access method to make a remote host connection. Use the following syntax:

```
OPTIONS COMAMID=access-method-id;
```

where COMAMID is an acronym for Communications Access Method Identification. *access-method-id* identifies the method used by the local host to communicate with the remote host. DECnet (an acronym for the Digital Equipment Corporation Networking architecture) is an example of *access-method-id*.

Example:

```
options comamid=decnet;
```

Alternatively, you may specify this option at a SAS invocation or in a SAS configuration file.

Specifying the Remote Host Name

To make a connection from a Windows local host to a remote host, use the following syntax:

```
OPTIONS REMOTE=ACI-information;
```

where *ACI-information* is represented as:

```
nodename"username password"::
         |"? ?"::
         |"username ?"::
         |"? password"::
```

If proxy access is enabled on the DECnet network, specify only the remote node name. Ask your network administrator if proxy access is enabled on the DECnet network. Proxy access precludes a need for you to assign your user name and password to the ACI. Otherwise, include the user name and password information in the ACI. Use one or two question marks, (?) or (??), to request that the local host be prompted for either user name or password or both.

Note: If a password is not required for an account, you may omit the password from the ACI. △

The remote host name and the two question marks signify a request for the local host to prompt for user name and password. Because the specification of ACI (host name, user name, and password) is not a valid SAS name, you must assign the ACI to a macro variable.

Instead of hard-coding user name and password values, you may use prompting as a security aid.

Here are some examples of specifying ACI and using secure user names and passwords.

Example 1:

```
%let rmthost=monarch"? ?";
options remote=rmthost;
```

The remote host name is MONARCH, and the two question marks signify a request for the local host be to prompted for both a user name and a password. Because the specification of ACI (host name, user name, and password) is not a valid SAS name, you must assign the ACI to a macro variable, such as RMTHOST, as shown in the first line. Then, use the SAS macro variable to define the remote host, as shown in the second line.

Example 2:

```
%let rmthost=monarch"bass time2go"::;
options remote=rmthost;
```

This example is similar to the preceding example, except that the ACI contains a literal user name and password instead of two question marks (??), which specify a prompt for a user name and a password.

Example 3:

```
c:> set rmthost=monarch"bass time2go"
```

This example is entered in a DOS window. The first line shows how to assign the remote host MONARCH the user name BASS, and the password TIME2GO to the variable RMTHOST.

Example 4:

```
options remote=monarch::;
```

Because proxy access is assumed, only the host name MONARCH is needed.

Alternatively, you may specify this option at a SAS invocation or in a SAS configuration file.

Signing On to the Remote Host

To complete your sign on to the remote host, enter the SIGNON statement, as follows:

```
signon;
```

Note: Sign-on script files are not needed on a Windows local host with the DECnet access method because DECnet connects to a spawner that runs on the remote host. △

Although no errors are produced if you specify a script file, you do waste processing time. If you defined the RLINK fileref before establishing a connection, when you sign on, SAS/CONNECT processes and loads the script file that is identified by the fileref, but the DECnet access method will ignore the script.

If you do not want to omit the RLINK fileref but you want to avoid wasting processing time, use the NOSCRIPT option in the SIGNON and SIGNOFF statements, shown as follows:

```
signon noscript;
.
.
.
signoff noscript;
```

Local Host Example

The following example illustrates the statements that you specify in a Windows local host SAS session to connect to a remote host by using the DECnet access method.

```
%let rmthost=rhost"bass time2go"::;
options comamid=decnet remote=rmthost;
signon;
```

A macro variable is used to assign the remote host name RHOST, the user name BASS, and the password TIME2GO to the alias RMTHOST. The OPTIONS statement specifies the DECnet access method and the macro variable RMTHOST as the remote host. The SIGNON statement performs the sign-on process.

Remote Host Tasks

System Administrator

To allow a connection from a local host, perform these tasks at the remote host:

1. Know about DECnet network drive restrictions.
2. Start the PC spawner program.
3. Optionally, set several remote host options.

Network Drive Restrictions in Windows NT Environments

As a Windows NT security feature, DECnet prohibits users from accessing remote network drives from a remote session on a Windows NT remote host. This feature was inherited by SAS/CONNECT when it uses the DECNET access method on the Windows NT platform. This restriction may be addressed in a future release of Windows NT or Pathworks. If you need to access a network file, then copy the file to a local drive and access it there. For more information, see the Microsoft Development Library and the Windows NT Knowledge Base articles # Q124184, Q132679, and Q122702.

Starting the PC Spawner Program

You must invoke the PC spawner program on the Windows NT, the Windows 95, and Windows 98 remote host to enable local hosts to connect to it. The spawner program resides on a remote host and listens for SAS/CONNECT client requests for connection to the remote host. After the spawner program receives a request, it invokes the remote SAS session.

For Windows NT only, setting the -SECURITY option in the PC spawner invocation command secures the spawner.

The spawner then verifies the user name and the password that are assigned to the ACI.

See Chapter 35, “PC Spawner Program,” on page 471 for information about starting the spawner on the remote host.

Setting Options at the Remote Host

Although sign-on script files are not used for the DECnet access method, you may set remote host options at the remote host.

It is recommended that you set these options:

NO$SYNTAXCHECK

allows the continuation of statement processing at the remote host regardless of syntax error conditions.

This option is valid as part of a configuration file, at a SAS invocation, or in an OPTIONS statement.

NOTERMINAL

specifies whether a terminal is attached at SAS invocation. If NOTERMINAL is specified, requestor windows are not displayed.

Set NOTERMINAL at the remote host so that no terminal is associated with the remote session. NOTERMINAL prevents SAS from displaying error messages and dialog boxes on the remote host, which requires user intervention.

This option is valid as part of a configuration file or at a SAS invocation.

See *SAS Language Reference: Dictionary* for details about this option.

NOXWAIT

Note: Applies to a Windows remote host only. △

specifies whether you have to type EXIT at the DOS prompt before the DOS shell closes. Set NOXWAIT at the remote host to prevent SAS from displaying a dialog box on the remote host. Such a display requires that you explicitly type EXIT at the remote host and gives the appearance that the REMOTE SUBMIT command is hung.

This option is valid as part of a configuration file, at a SAS invocation, or in an OPTIONS statement.

See *SAS Language Reference: Dictionary* for details about this option.

Remote Host Example

In order to allow a local host to connect to a Windows remote host, a PC spawner program must be invoked from the remote host. The spawner program is invoked with the DECnet access method by using

```
\sas\connect\sasexe\spawner  -comamid decnet -file spawnsas.bat;
```

The SPAWNSAS.BAT file is used to set the configuration on the PC. The SPAWNSAS.BAT file content is

```
@echo off
sas -config config.sas %1 %2 %3 %4 %5 %6 %7 %8
```

The following example illustrates the configuration file entries for a Windows remote host:

```
-no$syntaxcheck
-noterminal
-noxwait
```

SAS/SHARE

Client Tasks

System Administrator, Applications Programmer, and User

To access a SAS/SHARE server, perform the following tasks:

1. For Windows NT only, assign the appropriate rights to each connecting client.
2. For Windows NT only, set security for connecting clients.
3. For Windows NT, Windows 95, or Windows 98, specify the DECnet access method.
4. Specify the server name.

Assigning the Appropriate Rights for Connecting Clients

CAUTION:

Windows NT only This process applies to a Windows NT client only. △

The account in which a connecting client runs must have the appropriate rights. To assign these rights

1. Click on the Administrative Tools icon.
2. Click on the User Manager icon.
3. From the Policies pull-down menu, select “User Rights.”
4. Click on the “Show Advanced User Rights” box.
5. Assign “Log on as a batch job” rights to the appropriate users.

Setting Security for Connecting Clients

CAUTION:

Windows NT only This process applies to a Windows NT client only. △

Requiring connecting clients to supply both a valid user name and password enforces server security. At the client, set the preferred security method for relaying a userid and password that are valid on the server host. For details, see “Setting Security for SAS/CONNECT and SAS/SHARE” on page 367.

Specifying the DECnet Access Method

You must specify the DECnet communications access method at the client before you access a server.

Use the following syntax to specify the DECnet access method at each connecting client:

```
OPTIONS COMAMID=access-method-id;
```

where COMAMID is an acronym for Communications Access Method Identification. *access-method-id* identifies the method used by the client to communicate with the server. DECnet (an acronym for the Digital Equipment Corporation Networking architecture) is an example of an *access-method-id*.

Example:

```
options comamid=decnet;
```

The server is accessed using the DECnet access method.

You may specify the COMAMID option in an OPTIONS statement, at a SAS invocation, or in a SAS configuration file.

Additionally, you may use the COMAUX1 and COMAUX2 options to designate auxiliary communications access methods in a search list. See Table 1.3 on page 10 for the supported access methods by host. If the first method fails to access a server, the second method is attempted, and so on. You can specify up to two auxiliary access methods, depending on the number of methods that are supported between client and server hosts.

COMAUX options can be specified only at a SAS invocation or in a SAS configuration file. The syntax for the COMAUX options follows:

```
-COMAUX1 alternate-method
-COMAUX2 alternate-method
```

An example of configuration file entries for a Windows NT client connecting to a Windows NT server follows:

```
-comamid decnet
-comaux1 tcp
-comaux2 appc
```

If the server cannot be reached with the DECnet access method, a second attempt is made with the TCP/IP access method, and then with the APPC access method.

Specifying the Server Name

You must specify the server name in the PROC SERVER statement. Use the following syntax:

```
SERVER=server-id
```

See *SAS Language Reference: Concepts* for details about SAS naming rules. See *SAS/SHARE User's Guide* for details about the LIBNAME statement and the PROC OPERATE statement.

If the client and server sessions are running on different network nodes, you must include the DECnet host name in the server identifier in the LIBNAME statement and the PROC OPERATE statement as follows:

```
SERVER=host-name.serverid
```

This representation is known as a two-level server name.

The host name must be a valid DECnet host name. If the server and the client sessions are running on the same host, you may omit the host name.

If the DECnet host name is not a valid SAS name, you may assign the name of the server host to a SAS macro variable, then use the name of that macro variable as the *host-name* in the two-level server name.

The following example shows how to assign a server host name to a SAS macro variable:

```
%let srvhost=2beorno;
libname sales server=srvhost.server1;
```

Note: Do not use an ampersand (&) in a two-level server name. An ampersand causes the macro variable to be resolved by the SAS parser prior to syntactic evaluation of the SERVER option. A macro variable is transparently resolved in a SERVER option that is assigned a two-level server name. △

If you have represented a host name in several forms on your system, the access method will resolve the host name using this order of precedence:

□ acceptable host name

□ SAS macro variable

□ environment variable.

See *SAS Language Reference: Concepts* for details about SAS naming rules. See *SAS/SHARE User's Guide* for details about the PROC OPERATE statement and the LIBNAME statement.

Client Example

The following example illustrates the statements that you specify in a Windows NT client SAS session to access a server with the DECnet access method:

```
options comamid=decnet;
libname sasdata 'edc.prog2.sasdata' user=_prompt_ server=rhost.share1;
```

The COMAMID option specifies the DECnet access method. The LIBNAME statement specifies the data library that is accessed through the server RHOST.SHARE1 by means of a prompt for a username and a password that are valid on the server.

Server Tasks

Server Administrator

Note: Server security is supported on the Windows NT platform only. △

To set up a secure server and to make it accessible to a client, perform the following tasks:

1. For Windows NT only, assign the appropriate rights for a secure server.
2. For Windows NT, Windows 95, or Windows 98, require that only validated clients can access a secure server.
3. For Windows NT, Windows 95, or Windows 98, set DECnet access method security.
4. For Windows NT, Windows 95, or Windows 98, specify the DECnet access method.
5. For Windows NT, Windows 95, or Windows 98, specify the server name.

Assigning the Appropriate Rights for a Secure Server

CAUTION:

Windows NT only This process is supported on the Windows NT platform only. △

The account in which a secure server runs must have the appropriate rights. To assign these rights

1. Click on the Administrative Tools icon.
2. Click on the User Manager icon.
3. From the Policies pull-down menu, select "User Rights."
4. Click the "Show Advanced User Rights" box.
5. Assign "Act as part of the operating system" rights to the appropriate users.

Setting DECnet Access Method Security

Note:

CAUTION:

Windows NT only This process is supported on the Windows NT platform only. △

△

Before you can create a secure SAS/SHARE server, make the access method secure by assigning the value _SECURE_ to the SASSECUR option. See "Providing Client Identification in a pre-Version 8 Session" on page 368 for information about setting the SASSECUR option.

Specifying the DECnet Access Method at the Server

You must specify the DECnet communications access method before you can create and access a SAS/SHARE server.

Use the following syntax to specify the DECnet access method at the server:

```
OPTIONS COMAMID=access-method-id;
```

where COMAMID is an acronym for Communications Access Method Identification. *access-method-id* identifies the method used by the server to communicate with the client. DECnet (an acronym for the Digital Equipment Corporation Networking architecture) is an example of an *access-method-id*.

For a server that is running on a host on which only one communications access method is available, use the COMAMID option.

Example:

```
options comamid=decnet;
```

The server will be available only to SAS/SHARE sessions that use the DECNET access method.

You may specify the COMAMID option in an OPTIONS statement, at a SAS invocation, or in a SAS configuration file.

However, if the host on which a server is running supports multiple access methods, you may specify up to two auxiliary access methods that clients may use to access the server using the COMAUX1 and COMAUX2 options. See Table 1.3 on page 10 for the supported access methods by host.

All of the access methods initialize when the server initializes. The activation of multiple access methods makes a server available to several groups of clients, each using a different communications access method simultaneously.

COMAUX options can be specified only at a SAS invocation or in a SAS configuration file. The syntax for the COMAUX options follows:

```
-COMAUX1 alternate-method
-COMAUX2 alternate-method
```

An example of configuration file entries for a server that is running on a Windows NT host follows:

```
comamid decnet
comaux1 tcp
comaux2 appc
```

When the server starts, all of the communications access methods are initialized. The server is simultaneously available to client sessions that use the DECnet access method as well as to clients that use the TCP/IP and APPC access methods.

Specifying a Server Name

You must specify the server name in the PROC SERVER statement. Use the following syntax:

SERVER=*server-id*

See *SAS Language Reference: Concepts* for details about SAS naming rules. See *SAS/SHARE User's Guide* for details about the LIBNAME statement and the PROC OPERATE statement.

Server Example

The following example illustrates the statements that you specify in a SAS session on the Windows NT host at which you start a server:

```
%let sassecur=_secure_;
options comamid=decnet;
proc server id=share1 authenticate=req;
run;
```

The value _SECURE_ (for the SASSECUR macro variable) requires that clients specify a user name and a password that are valid on the server. The DECnet access method is declared, and the server SHARE1 is started on the Windows NT host. The additional options in the PROC SERVER statement allow only validated clients to access the server.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Communications Access Methods for SAS/CONNECT and SAS/SHARE Software, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 643.

Communications Access Methods for SAS/CONNECT and SAS/SHARE Software, Version 8

ISBN 1–58025–479–9

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

1st printing, September 1999

The Institute is a private company devoted to the support and further development of its software and related services.