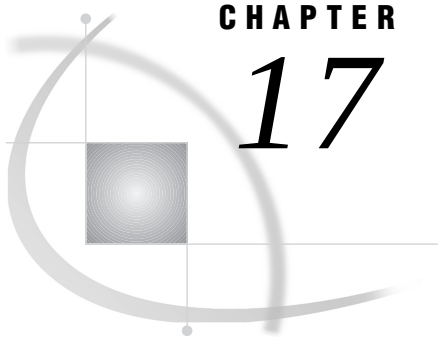

CHAPTER

17

OS/2: TCP/IP Access Method

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

System Administrator or User

To use the TCP/IP access method with an OS/2 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 the resources for the TCP/IP access method have been defined.
3. Verify that you know how to set options in SAS software.
4. 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:

- SAS software has been installed on both the local and remote hosts.
- For OS/2 hosts, a supported TCP/IP package must be installed. Supported TCP/IP products follow:
  - the IBM TCP/IP Version 3.0 or a subsequent version.

Specify the host name during the IBM TCP/IP configuration process. If you omit the host name when you sign on with SAS/CONNECT or with SAS/SHARE, you will receive the following error message:

```
ERROR:  Access method initialization failed.
```

Setting SAS Options and Variables

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

Consult with your network administrator to determine what options must be set and what values to assign to them.

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

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

  OPTIONS SET=*variable value*;

  Example:

  ```
  options set=tcpsec _secure_;
  ```

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

  -SET *option-name value*

  Example:

  ```
  -set tcpsec _secure_
  ```

- SAS macro variable:

  %LET *variable=value*;

  Example:

  ```
  %let tcpsec=_secure_;
  ```

- DOS operating system environment variable:

  SET *variable-name=value*

  Example:

  ```
  set tcpsec=_secure_
  ```

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

If you set multiple forms of the same option, here is the order of precedence that is followed:

SAS macro variable

OPTIONS statement

AUTOEXEC file

SAS invocation

SAS configuration file

DOS environment variable.

Note: If you set the same option using different forms, typically the last option setting will take precedence and override an earlier option setting. △

Displaying SAS System Option Settings

To display the settings of the SAS system options in the SAS log, use the OPTIONS procedure. The following statement produces a list of options with a brief explanation of what each option does:

```
proc options;
run;
```

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. The third section describes how to configure your SAS/SHARE server to either require or not require connecting clients to supply user identification.

Providing Client Identification in a Version 8 Session

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 TCPSEC variable is still accepted but is not recommended in Version 8. The USER= and PASSWORD= options take precedence over the client TCPSEC 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 TCPSEC option setting in the same client SAS session.

CAUTION:
In order to make a SAS/SHARE server secured, the TCPSEC 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.

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

In Version 6 and Version 7, you provide client identification to a SAS/CONNECT remote host or a SAS/SHARE server using the TCPSEC option. TCPSEC must be defined on the local host before you connect to the remote host (using the SIGNON statement) or access a SAS/SHARE server (using the LIBNAME statement).

Here is the syntax and description of this option.

TCPSEC=*userid.password* | _PROMPT_

userid.password
specifies the remote host userid and password and is thus host-dependent in form. If either the userid or password contains blanks or special characters, it must be enclosed in quotes. A period (.) is used as a delimiter between the userid and password and, therefore, is not a valid character.

PROMPT
specifies that the SAS system prompt the client for the userid and password.

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

This technique is especially useful when the configuration file specifying this option is shared among many users.

Examples:

```
options set=tcpsec bass.time2go;
options set=tcpsec _prompt_;
```

Providing Userid-Based Security for a SAS/SHARE Server

The TCPSEC option also specifies whether the TCP/IP access method performs user authentication before connecting to a SAS/SHARE server. The TCPSEC option must be set on the server before you start the SAS/SHARE server.

Here is the syntax and description of this option.

TCPSEC=_SECURE_ | _NONE_

SECURE
The _SECURE_ value for the TCPSEC option causes the TCP/IP access method to attempt to authenticate connecting SAS/SHARE clients. Each client connecting using TCP/IP is required to supply a userid and password valid for the host on which the server is running.

NONE
The _NONE_ value for the TCPSEC option causes the TCP/IP access method to NOT attempt to authenticate connecting SAS/SHARE clients. This is the default action when TCPSEC has not been set.

Examples:

```
options set=tcpsec _secure_;
options set=tcpsec _none_;
```

SAS/CONNECT Only Options

TCPPORTFIRST

TCPPORTLAST

The TCPPORTFIRST and TCPPORTLAST SAS options restrict the range of TCP/IP ports through which local hosts can remotely connect to remote hosts.

These options must be set at the SAS/CONNECT remote host.

Define the range of TCP/IP ports by assigning a beginning range value to TCPPORTFIRST and an ending range value to TCPPORTLAST, within the range of 0 through 32767.

Consult with your network administrator for advice about these settings.

Use the following syntax for the configuration file:

```
-TCPPORTFIRST n
-TCPPORTLAST n
```

Use the following syntax for the AUTOEXEC file:

```
OPTIONS TCPPORTFIRST=n;
OPTIONS TCPPORTLAST=n;
```

In the following example, the local host is restricted to TCP/IP ports 4020 through 4050 when making a remote host connection:

```
options tcpportfirst=4020;
options tcpportlast=4050;
```

To restrict the range of ports to only one port, you may set the TCPPORTFIRST and TCPPORTLAST options to the same number.

Note: At the remote host, you may set TCPPORTFIRST and TCPPORTLAST at a SAS invocation or in the configuration file. △

TCPTN3270

TCPTN3270 is an environment variable that is set on the local host to support a connection to an OS/390 or CMS host that uses full-screen 3270 TELNET protocol. The following script files are provided:

CMS	TCPCMS32.SCR
OS/390	TCPTSO32.SCR

See “Identifying a Script File for Signing On and Signing Off” on page 251 for more information.

Set TCPTN3270 to the value of 1 at the OS/2 local host in the SAS configuration file or in an OPTIONS statement.

To set the TCPTN3270 variable, enter the following command on the OS/2 local host:

Example:

```
-set tcptn3270 1
```

SAS/CONNECT

Local Host Tasks

User or Applications Programmer

To connect an OS/2 local host to a remote host, perform these tasks at the local host:

1. Consider the requirements of the remote host that you are connecting to.
2. If you connect to a UNIX, an OS/390, or an OpenVMS Alpha or VAX remote host by means of the spawner program, configure the spawner in the SERVICES file, as necessary.
3. If you connect to a remote host by means of a spawner, set security, as necessary.
4. Configure the local and remote hosts' names and Internet addresses in the local HOSTS file or through the domain server.
5. Specify the communications access method.
6. Specify the remote node name.
7. Identify the script file to be used for signing on and signing off, as necessary.
8. Sign on to the remote host.

Remote Host Connection Considerations

If you are connecting to a Windows 95, Windows 98, or Windows NT remote host, you *must* connect by means of a spawner program that is already running on the remote host. If you are connecting to an OS/2, a UNIX, an OS/390, or an OpenVMS Alpha remote host, you optionally *may* connect by means of a spawner program, that also must already be running on the remote host. A spawner program allows the encryption of userids and passwords when passed through the network. Without a spawner, readable userids and passwords are passed through the network, which may present a security risk. See Chapter 32, "Spawner Programs," on page 457 for information about starting the spawner on the remote host.

You may also sign on to the remote host with a script file. If you do not sign on with a script file, as a security measure, set the USER= and PASSWORD= options in the SIGNON statement, which is passed to the remote host, allowing a local host connection.

Note: Setting the Version 7 TCPSEC variable at the local host will also work. △

If the -NOSCRIPT option is set at the spawner invocation, sign on with a script is prohibited. Ask your network administrator whether the -NOSCRIPT option is set at the spawner invocation.

For all other hosts, you will sign on with a script.

Configuring the Spawner Service in the SERVICES File

To prepare for local hosts that connect to a UNIX, an OS/390, or an OpenVMS Alpha remote host with the spawner program, configure the spawner service in the SERVICES file at the local host. See "Configuring the SERVICES File" on page 485 for more information.

Setting Security for Local Hosts

If you are not using a script file to sign on to the remote host, set security at the local host using either of the methods explained in "Setting Security for SAS/CONNECT and SAS/SHARE" on page 245. For Version 8 security behavior, specify the USER= and PASSWORD= options in the SIGNON statement. For details, see "Providing Client Identification in a Version 8 Session" on page 245.

For Version 7 security behavior, if you set the TCPSEC option at the local host, either specify a userid and a password that are valid on the remote host or specify _PROMPT_ to supply the userid and password when connecting to a remote host. For information about setting the TCPSEC option, see "Providing Client Identification in a pre-Version 8 Session" on page 247.

Configuring Local and Remote Host Names and Internet Addresses

You must specify the names and Internet addresses of the local and remote hosts in the HOSTS file or through the name server. A name server program supplies name-to-address translation, mapping from domain names to IP addresses. The name server often runs on a dedicated processor, and the host itself is referred to as the name server.

The format for a HOSTS file entry follows:

```
Internet-address host-name optional-alias
```

Example:

```
172.20.10.200       monarch       local
172.20.10.201       omega         remote
```

Specifying the TCP/IP Communications Access Method

You must specify the TCP/IP 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.

TCP (short for TCP/IP, which is an acronym for Transmission Control Protocol/ Internet Protocol) is an example of *remote-host-id*.

Example:

```
options comamid=tcp;
```

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

Specifying the Remote Node Name

To make a connection from an OS/2 local host to a remote host, use the following syntax:

```
OPTIONS REMOTE=node-name.<service-name>;
```

The value for *node-name* that you specify is based on the type of remote host that you are connecting to.

- If you are connecting to a Windows NT, a Windows 95, a Windows 98, or an OS/2 remote host that is running the PC spawner program, use the name of the node on which the PC spawner is running. See Chapter 35, "PC Spawner Program," on page 471 for more information.
- If you are connecting to a UNIX, an OS/390, or an OpenVMS Alpha host that is running the spawner program, use a two-level name in the form of

node-name.service-name, where *node-name* specifies the node on which the spawner program is running and *service-name* specifies the port on which the spawner is listening for a connection request.

See Chapter 32, "Spawner Programs," on page 457 for information about the spawner program and "Starting the UNIX Spawner Program" on page 479 for information about configuring the spawner service in the SERVICES file.

- If you are connecting to any other remote host platforms that use a sign-on script, use the node name of the remote host. The remote host must be defined in a local HOSTS file or in a domain name server.

The value of the REMOTE= option must be a valid SAS name. See *SAS Language Reference: Dictionary* for details about SAS naming rules.

Example:

```
options remote=node-name;
```

If you use an Internet address (or some other invalid SAS name), you must assign the address to a macro variable, then specify the macro variable for the value of the REMOTE= option.

Example:

```
%let node=Internet-address;
options remote=node;
```

Do not choose a macro name that is also a valid host name on your network. SAS first attempts to reach a network host of the REMOTE= option value (in this example, MYNODE).

Example:

```
%let mynode=172.20.10.200;
options remote=mynode;
```

Identifying a Script File for Signing On and Signing Off

Note: If you use the spawner NOSCRIPT option, you will not use a script file. In the absence of a script file, be sure to set security. For information about setting security, see "Setting Security for SAS/CONNECT and SAS/SHARE" on page 245. See Chapter 32, "Spawner Programs," on page 457 for full details about starting the spawner. Because there is no script file, you will not define an RLINK fileref statement. △

To use one of the sample script files that is supplied with SAS/CONNECT for signing on and signing off, assign the RLINK fileref to the appropriate script file, based on the remote host that you are connecting to. The sample scripts are installed at *!SASROOT*\CONNECT\SASLINK. You must customize the sample scripts to accurately reflect your site logon process. Failure to do so will produce errors.

The FILEREF syntax follows:

```
FILENAME RLINK '!sasroot\connect\saslinkscript-name';
```

where *script-name* specifies the appropriate script file for the remote host.

The following table lists the scripts that are supplied by SAS Institute:

Table 17.1 OS/2 TCP/IP SAS/CONNECT Sign-on Scripts

Remote Host	Script Name
CMS	TCPCMS.SCR
CMS (using full-screen 3270 TELNET protocol)	TCPCMS32.SCR
OS/390 (with TSO)	TCPTSO.SCR
OS/390 (without TSO)	TCPMVS.SCR
OS/390 (using full-screen 3270 TELNET protocol)	TCPTSO32.SCR
OpenVMS	TCPVMS.SCR
OS/2	TCPOS2.SCR
UNIX	TCPUNIX.SCR
Windows NT, Windows 95, and Windows 98	TCPWIN.SCR

Example:

```
filename rlink '!sasroot\connect\saslink\tcptso32.scr';
```

Note: If you connect to an OS/2, a Windows NT, a Windows 95, or a Windows 98 remote host by means of a TELNET daemon, then your script must invoke the SASDMR program in the TYPE statement that starts the remote SAS session. If you connect to an OS/2, a Windows NT, a Windows 95, or a Windows 98 remote host by means of a PC spawner program, then your script must invoke the SAS program in the TYPE statement that starts the remote SAS session. You use the same SAS options with the SASDMR or SAS program as you would in the SAS command. The SASDMR command invokes a special version of the SAS program that redirects the output from the remote SAS session back to the local SAS session. Otherwise, if you connect to an OS/2 remote host, your script must invoke the **sas.exe** file. △

To secure the remote host, set security using an approved method. See "Setting Security for SAS/CONNECT and SAS/SHARE" on page 245. For information about the spawner, see Chapter 32, "Spawner Programs," on page 457.

Should you want to prevent sign ons with a script, set the NOSCRIPT option in the PC spawner invocation command. See Chapter 35, "PC Spawner Program," on page 471 for details about the NOSCRIPT option. Because there is no script file, you will not define an RLINK fileref statement.

Signing On to the Remote Host

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

```
signon user=_prompt_;
```

To set security at the remote host, specify valid values for the USER= and PASSWORD= options in the SIGNON statement. For details, see "Providing Client Identification in a Version 8 Session" on page 245.

An example of signing on without a script to a Windows NT remote host follows:

```
signon user=_prompt_  noscript;
```

Local Host Example

The following example illustrates the statements that you specify in an OS/2 local host SAS session to connect to a remote host with the TCP/IP access method.

```
filename rlink '!sasroot\connect\saslink\tcptso.scr';
options comamid=tcp remote=rmtnode;
signon;
```

The first line identifies the script file that you use to sign on to an OS/390 remote host. The script file contains a prompt for a userid and a password that are valid on the remote host. The TCP/IP communications access method is declared with a connection to the remote host RMTNODE.

Remote Host Tasks

System Administrator

Before a user on a local host can sign on through a spawner program to either an OS/2, a UNIX, a Windows NT, a Windows 95, or a Windows 98 remote host, perform the following task:

1. Start the spawner program at the remote host, as necessary.

Starting the PC Spawner Program

Note: Running the PC spawner program on an OS/2 remote host is optional. △

Optionally, you invoke the PC spawner program on the OS/2 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. See Chapter 35, “PC Spawner Program,” on page 471 for information about starting the PC spawner program.

Remote Host Example

Set the following options to restrict port access in the remote host AUTOEXEC file to ports 5020 through 5050:

```
options tcpportfirst=5020;
options tcpportlast=5050;
```

SAS/SHARE

Client Tasks

User and Applications Programmer

To prepare for accessing a SAS/SHARE server, perform these tasks:

1. Configure the server in the client SERVICES file.
2. Set security for connecting clients.
3. Specify the TCP/IP access method.
4. Specify a server name.

Configuring the Server in the SERVICES File

Each server must be defined as a service in the SERVICES file on each host node from which a client session accesses the server. This file usually is located in the directory in which the TCP/IP product is installed.

If you are using IBM TCP/IP, the SERVICES file is located in the \TCPIP\ETC directory, by default. If you are using Novell LAN WorkPlace, the SERVICES file is in \LANWP file, by default. For all other versions of TCP/IP, refer to your documentation to find out where the SERVICES file is installed. See "Configuring the SERVICES File" on page 485 for information about editing the SERVICES file.

Setting Security for Connecting Clients

Requiring connecting clients to supply both a valid userid 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 245.

Specifying the TCP/IP Communications Access Method

You must specify the TCP/IP communications access method at the client before you access a server.

Use the following syntax to specify the TCP/IP 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. TCP (an abbreviation for Transmission Control Protocol/Internet Protocol) is an example of an *access-method-id*.

Example:

```
options comamid=tcp;
```

The server is accessed using the TCP/IP 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. 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 based 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 an OS/2 client connecting to an OS/390 server follows:

```
-comamid tcp
-comaux1 appc
```

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

Specifying a Server Name

If the client and server sessions are running on different network nodes, you must include the TCP/IP node in the server identifier in the LIBNAME and PROC OPERATE statements as follows:

```
SERVER=node.server
```

This representation is known as a two-level server name.
node must be a valid TCP/IP node name. If the server and the client sessions are running on the same node, you may omit the node name.
server can represent either a *server-id* or a *port* number.

- *server-id* must be identical to the service name specified in the SERVICES file. See "Configuring the SERVICES File" on page 485 for more information on specifying the *server-id* in the SERVICES file.
- *port* is the location for passing data to and receiving data from the server. The port number is specified with two preceding underscore (_) characters. For example, you can specify the server port as 5000 using the SERVER= option in a LIBNAME statement:

```
libname mylib '.' server=srvnode.__5000;
```

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

Client Example

The following example illustrates the statements that you specify in an OS/2 client SAS session to access a server with the TCP/IP access method:

```
options comamid=tcp;
libname sasdata 'c:edc\prog2\sasdata' user=_prompt_ server=rmtnode.share1;
```

The first line declares the TCP/IP access method. The LIBNAME statement specifies the data library that is accessed through the server, which is specified by the two-level server name RMTNODE.SHARE1, by means of a prompt for a username and a password that are valid on the server.

Server Tasks

Server Administrator

To set up a server, perform the following tasks at the server:

1. Configure a SAS/SHARE server service in the SERVICES file.
2. Set the TCPSEC variable for server security.
3. Specify the TCP/IP access method.
4. Specify the server name.

Configuring the Server Service in the SERVICES File

Each server must be defined as a service in the SERVICES file on each remote host node on which a server runs. This file is located in the directory in which the TCP/IP product is installed. Find out the correct location of the TCP/IP package on your host system. See "Configuring the SERVICES File" on page 485 for more information.

Setting Server Security

You may use file permissions to restrict a user's access to libraries and files through a server. A secured server allows connections only from those clients that provide valid userids and passwords for the host on which the server is running. A secured server uses a validated userid and password pair to verify a user's authority to access a SAS library or a SAS file.

Requiring connecting clients to supply a valid userid and password enforces server security. From a server session, set the TCPSEC option to the value _SECURE_. See "Providing Client Identification in a pre-Version 8 Session" on page 247 for more information about setting this option.

Specifying the TCP/IP Access Method at the Server

You must specify the TCP/IP access method at the server before you start a SAS/SHARE server.

Use the following syntax to specify the TCP/IP 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. TCP (short for TCP/IP, which is an abbreviation for Transmission Control Protocol/Internet Protocol) 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 only the COMAMID option.

Example:

```
options comamid=tcp;
```

The server will be available only to SAS/SHARE sessions that use the TCP/IP 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 by which clients may access the server. 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 an OS/2 host follows:

```
-comamid tcp
-comaux1 appc
```

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

See *SAS/SHARE User's Guide* for details about starting and accessing a server.

Specifying a Server Name

If the client and server sessions are running on different network nodes, you must include the TCP/IP node in the server identifier in the PROC SERVER statement as follows:

```
SERVER=server
```

server can represent either a *server-id* or a *port* number.

- *server-id* corresponds to the service that was configured in the SERVICES file. See "Configuring the SERVICES File" on page 485 for more information.
- *port* is the location for passing data to and receiving data from the server. The port number is specified with two preceding underscore (_) characters. For example, you can specify the server port as 5000 using the SERVER= option in a LIBNAME statement:

```
libname mylib '.' server=__5000;
```

See *SAS Language Reference: Dictionary* for details about SAS naming rules. See *SAS/SHARE User's Guide* for details about the PROC SERVER statement.

Server Example

The following example illustrates the statements that you specify in a SAS session on the OS/2 host at which you start a server:

```
options comamid=tcp;
proc server id=share1;
run;
```

The TCP/IP access method is declared, and the server SHARE1 is started on the OS/2 host.

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