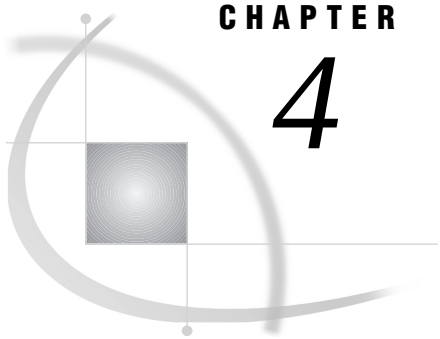

CHAPTER

4

CMS: 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 a CMS 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 environment variables in SAS software.
4. Set the desired SAS/CONNECT and SAS/SHARE environment variables.

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

Ensure that the following conditions have been met:

1. TCP/IP has been installed at both the local and remote hosts at your site.
2. Your system is running IBM CMS TCP/IP Version 2, Release 3, or a subsequent release.
3. SAS is installed on both the local and remote hosts.

Defining Resources for the TCP/IP Access Method

System Administrator, SAS Site Representative, Applications Programmer, User

Before you can use SAS/CONNECT or SAS/SHARE with the TCP/IP access method, you must first define TCP/IP resources for the CMS system. See "System Configuration for the TCP/IP Access Method" on page 65 for information about defining resources for SAS/CONNECT and SAS/SHARE.

Setting SAS Options and Variables

You may need to set specific variables in SAS to establish the connections that you want with SAS/CONNECT and SAS/SHARE when using the TCP/IP communications access method. Ask your network administrator for advice about these settings.

You may specify a SAS variable in one of these forms:

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

  OPTIONS *variable=value*;

  Example:

```
OPTIONS COMAMID=TCP;
```

- a SAS macro variable:

  %let *variable-name=value*;

  Example:

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

- a CMS global variable:

  GLOBALV SETL *variable-name value*

  Example:

```
GLOBALV SETL TCPTN3270 1
```

For the global variable method, *variable-name* must be uppercase, and *value* may be mixed case.

Values for these variables 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 variable, this is the order of precedence that is followed:

OPTIONS statement

SAS macro variable

CMS global 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. △

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 variable 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 variable setting in the same client SAS session.

CAUTION:
In order to make a SAS/SHARE server secured, the TCPSEC option must still 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 side whose username and password have been verified to access the remote side.

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 variable. 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 variable.

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.

Note: The value of TCPSEC will be set and displayed like any other macro variable. Thus if the %LET statement that is used to set the value of TCPSEC appears in the SAS log, the password will also appear in plain text. Similarly, a %PUT statement will also print the password in plain text. △

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 variable is shared among many users.

Examples:

```
%let tcpsec=bass.time2go;
%let tcpsec=_prompt_;
```

Providing Userid-Based Security for a SAS/SHARE Server

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

Here is the syntax and description of this variable.

TCPSEC=_SECURE_ | _NONE_

SECURE
The _SECURE_ value for the TCPSEC variable 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 variable 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:

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

SAS/CONNECT Only Options and Variables

TCPPORTFIRST

TCPPORTLAST

The TCPPORTFIRST and TCPPORTLAST 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 in an OPTIONS statement, at a SAS invocation, in the configuration file, or in the AUTOEXEC file. △

TCPTN3270
: TCPTN3270 is an environment variable that is set on the local host to support connections to CMS and OS/390 remote hosts that use the full-screen 3270 TELNET protocol. The following sample script files are provided with SAS/CONNECT:

  CMS TCPCMS32.SCR

  MVS TCPTSO32.SCR

  See “Identifying a Script File for Signing On and Signing Off” on page 59 for information about these script files.

  At the CMS local host, set the TCPTN3270 variable as follows:

  ```
  GLOBALV SETL TCPTN3270 1
  ```

  If you do not set this variable, the TCP/IP access method uses the TELNET line mode protocol by default.

SAS/SHARE Only Variable

By default, a secure server accepts userids and passwords from clients in either encrypted or in plain text form. Being able to accept either form ensures compatibility with client sessions that are running earlier releases of SAS (releases prior to 6.09E).

To require only encrypted userids and passwords, you must set the CMS global variable AUTHENCR or a SAS macro variable. Requiring encryption ensures that all clients have been upgraded to Release 6.09E or to Release 6.11 of SAS software.

Setting the variable AUTHENCR in a server session enables encryption for clients that are connecting to a secure server. The values for this variable are:

AUTHENCR=OPTIONAL | REQUIRED

OPTIONAL
: means that a client can optionally encrypt the username and the password that it sends to the server. This is the default. When using the default, the server allows connections from clients that are capable of encryption and from clients that are incapable of using encryption because they are running earlier releases of SAS that do not support encryption (releases prior to Release 6.09E and Release 6.11).

REQUIRED
means that each client must encrypt the username and the password that it sends to the server.

See “Setting SAS Options and Variables” on page 52 for examples of the forms that you can use to specify the AUTHENCR variable.

SAS/CONNECT

Local Host Tasks

User or Applications Programmer
To connect a CMS 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. Configure the Alpha/VMS, the UNIX, or the OS/390 spawner in the SERVICES file, as necessary.
3. Set security for local hosts.
4. Specify the communications access method.
5. Specify a remote host to connect to.
6. Identify the script file to be used for signing on and signing off, as necessary.
7. 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 Alpha/VMS remote host, you optionally *may* connect by means of a spawner program, which 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 also works. △

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

Before connecting to either a UNIX, an OS/390, or an Alpha/VMS remote host through a spawner program, configure the spawner service in the SERVICES file on 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 53. 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 53.

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 54.

Specifying the TCP/IP 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 abbreviation for Transmission Control Protocol/Internet Protocol) is an example of an *access-method-id*.

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

Example:

```
options comamid=tcp;
```

Specifying the Remote Node Name

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

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

The value that you specify for *node-name* 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 information.
- If you are connecting to a UNIX host or an OS/390 host that is running a 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 more information. See “Configuring the SERVICES File” on page 485 for information about configuring the spawner service.
- If you are connecting to a remote host platform that uses 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 and specify the macro variable as 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 with the value of the REMOTE= option (in this example, MYNODE).

Example:

```
%let mynode=149.999.228.6;
options remote=mynode;
```

Identifying a Script File for Signing On and Signing Off

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, depending on the remote host that you are connecting to. The sample scripts are installed at SASCONNE MACLIB. You must customize the sample scripts to accurately reflect your site's logon procedures; failure to do so will produce errors.

The fileref format follows:

```
filename rlink 'sasconne maclib';
```

These commands save the script file in the local host's environment.

On the command line in the PROGRAM EDITOR window, enter the following commands:

```
inc rlink(script-name)
file 'script-name SCR'
```

Then enter the following statement:

```
filename rlink 'script-name SCR';
```

where *script-name* identifies the script that corresponds to the remote host that you want to connect to.

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

Table 4.1 CMS TCP/IP SAS/CONNECT Sign-on Scripts

Remote Host	Script Name
CMS	TCPCMS
CMS (using fulls-creen 3270 TELNET protocol)	TCPCMS32
TSO under OS/390	TCPTSO
OS/390 (without TSO)	TCPMVS

Remote Host	Script Name
OS/390 (using full-screen 3270 TELNET protocol)	TCPTSO32
OpenVMS	TCPVMS
OS/2	TCPOS2
UNIX	TCPUNIX
Windows NT, Windows 95 , and Windows 98	TCPWIN

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 53.

Local Host Example

The following example illustrates the statements that you specify in a CMS local host SAS session in order to connect to a remote host running the spawner program configured for the TCP/IP access method.

These commands save the script file in the local host's environment. The fileref format follows:

```
filename rlink 'sasconne maclib';
```

Issue the following commands from the command line:

```
inc rlink(tcpunix)
file 'tcpunix scr'
```

The following statements are issued in the CMS local host SAS session:

```
filename rlink 'tcpunix scr a';
options comamid=tcp remote=rmthost.unxspawn;
signon user=_prompt_;
```

The first line identifies the script file that you use to sign on to the UNIX remote host by means of the UNIX spawner program. The script file contains a prompt for a userid and a password that are valid at the remote host. The TCP/IP communications access method is declared with a connection to a remote UNIX spawner, which is identified by the two-level name RMTHOST.UNXSPAWN. The USER= option in the SIGNON statement specifies that the connecting local host be prompted for a userid and a password that are valid on the remote host.

Remote Host

You do not perform any tasks at the CMS remote host for the TCP/IP access method.

SAS/SHARE

Client Tasks

User and Applications Programmer

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

1. Configure the server in the client SERVICES file.
2. Set security.
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 will access the server. This file usually is located in the directory in which the TCP/IP software 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 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 53.

Specifying the TCP/IP Access Method

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 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*.

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 1.1 on page 8 for the supported access methods by host. If the first method that is designated 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 access 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 CMS client connecting to a CMS server follows:

```
comamid=tcp
comaux1=appc
comaux2=iucv
```

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

Specifying a Server Name

If the client and server sessions are running on different network nodes, you must include the node name 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;
```

If the TCP/IP node name is not a valid SAS name, you can assign the name of the server's node to a SAS macro variable. Use the name of the macro variable for *node* in the two-level server name.

The access method evaluates the node name, in this order of precedence:

- node name that is defined to the TCP/IP software and is a valid SAS name
- SAS macro variable
- environment variable.

The following example shows the assignment of a SAS macro variable to a server's node name:

```
%let srvnode=mktserve.acme.com;
libname sales 'sasdata a' server=srvnode.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 macro facility prior to syntactic evaluation of the SERVER= option. The access method evaluates the node name in a two-level server name. △

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

Client Example

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

```
options comamid=tcp;
libname sasdata 'sasdata a' user=_prompt_ server=rmtnode.share1;
```

The COMAMID option declares the TCP/IP access method. The LIBNAME statement specifies the data library that is accessed through the server (which is identified 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 host.

Server Tasks

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

1. Configure SAS/SHARE servers in the SERVICES file.
2. Set the TCPSEC variable for server security.
3. Set the AUTHENCR variable to enforce client userid and password encryption.
4. Set authentication and permissions for the server.
5. Specify the TCP/IP access method.
6. Specify the server name.

Note: Optional tasks apply to setting up server security. △

Configuring the Server in the SERVICES File

Each SAS/SHARE server must be defined as a service in the SERVICES file of each remote host node on which a server runs and on each node from which a user session accesses the server. This file is located in the directory in which TCP/IP software is installed.

Example:

```
server1      5010/tcp    # SAS/SHARE server 1
```

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 secure server allows connections only from those clients that provide valid userids and passwords for the host on which the server is running. A secure server uses a validated userid and password to verify a user's authority to access a SAS library or file.

Requiring connecting clients to supply a valid userid and password enforces server security. From a server session, set the TCPSEC variable to the value _SECURE_. See "Providing Client Identification in a pre-Version 8 Session" on page 54 for more information.

Enforcing Server Userid and Password Encryption

As a security measure, you may set the AUTHENCR variable to enforce the encryption of userids and passwords when they are passed from the client to the server. See “SAS/SHARE Only Variable” on page 56 for details about setting the AUTHENCR variable.

Set User Authentication and Permissions for the Server

Authentication and permissions are set by calling “user exits” and are supplied by the user. SAS provides several examples that can be used. Documentation for these exits is provided in the *CMS Installation Guide*.

Specifying the TCP/IP Access Method

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

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 a CMS host follows:

```
comamid=tcp
comaux1=appc
comaux2=iucv
```

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 and IUCV access methods.

Specifying a Server Name

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

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 CMS host at which you start a server:

```
%let tcpsec=_secure_;
options comamid=tcp;
proc server id=share1 authenticate=req;
run;
```

The value _SECURE_ for the TCPSEC macro variable requires clients to supply a valid server userid and password in order to allow client access to the server. The TCP/IP access method is declared and the server SHARE1 is started on the CMS host.

System Configuration for the TCP/IP Access Method

System Administrator, SAS Site Representative, Applications Programmer, User
The TCP/IP access method for the CMS host requires IBM TCP/IP Version 2, Release 3 or subsequent release.

To configure the TCP/IP access method for SAS/CONNECT and SAS/SHARE, you must perform the following tasks or have knowledge of the following issues

1. Define the TCP/IP server host.
2. Understand the IBM TCP/IP network configuration files.
3. Understand the search order for locating host names and Internet addresses.
4. Know about the HOSTS file.

Defining the TCP/IP Server Host

The access method must locate a TCP/IP virtual machine whose name can vary from site to site. The TCP/IP access method searches for the virtual machine name in the file TCPIP DATA. If this file is unavailable, the access method uses a default name of TCPIP.

Creating the Server VM Directory Entry for the Server Virtual Machine

You must include the following statements in the VM directory entry for the server's virtual machine:

```
USER SASSHARE XXXXXXX 20M 20M G 100
MACHINE XA
IPL CMS PARM AUTOCR
CONSOLE 009 3215
SPOOL   00C 2540 READER *
SPOOL   00D 2540 PUNCH A
SPOOL   00E 1403 A
LINK MAINT 19E 19E RR
LINK MAINT 19D 19D RR
LINK MAINT 190 190 RR
MDISK 191 3380 707 5 VM0800 MR XXXXXXX
MDISK 192 3380 501 3 VM0450 MR XXXXXXX
```

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