CHAPTER

40

Permission Program

Support for Version 6 Only

Note: Beginning with Version 7, SAS/SHARE uses the **sasperm** program that is automatically invoked to allow clients to access SAS libraries or files. However, information about the **permission** program is included here for Version 6 users. △

Allowing SAS/SHARE Client Access to SAS Libraries or Files

You may write and use the **permission** program on a SAS/SHARE server that is running on a UNIX host to allow clients to access SAS libraries or files.

When presented with a validated userid, the server uses a customer-supplied program named **permission** to verify the following attributes:

- user name
- file or directory path for a SAS library or file
- file or directory access permissions (read or write).

The server invokes this program whenever a client tries to access a SAS library or file.

The **permission** program determines whether or not the requesting user has the specified access to the file or directory. If the user has the appropriate access permissions, the program exits with a zero return code. If the user does not have the appropriate access permissions, the program exits with a non-zero return code.

It is recommended that you write attempts, successes, and failures from the executable file to a log. Also, it is recommended that you fail the validation for any step in the process that has a problem.

After you write and test the program, move it to the *!sasroot* **/utilities/bin** directory where SAS/SHARE expects the program to be located.

Note: Methods for implementing file-access security vary across types of UNIX systems. Although many UNIX systems use conventional UNIX file-access permissions for owner, group, and other, some UNIX systems use different methods, such as Access Control Lists (ACLs). △

In addition, SAS requires the user to have execute permission in order to access a directory that contains a SAS data library. For systems that use ACLs, ask your system administrator or software vendor for the correct methods to validate access on your system.

The sample programs in the *!sasroot* **/utilities/src** directory verify a user's access rights by using both conventional UNIX permissions and ACLs.

Permission Program Examples

The *!sasroot* **/utilities/src** directory contains documented examples of the following permission programs:

- **perm.conv.c**
- **perm.afsacl.c**
- **perm.aixacl.c**
- **perm.hpacl.c**

Each of these programs verifies a user's access rights using both conventional UNIX permissions and ACLs. The filename extension indicates the specific type of UNIX system and the type of permissions for which the programs were designed. For example, **perm.aixacl.c** specifies a permission program for an AIX UNIX system that uses ACLs.

Building the Permission Program

In most cases, the working examples can be built with the following commands:

```
%
cd !sasroot/utilities/src

%
cc -o permission perm.conv.c
```

The **cc** command typically is the name of the C language compiler, but the command that you use on your system may be different. You do not need to set high optimization or to use an ANSI standard compiler to build the program because it already uses the standard C library functions for most of the work. See the README files for details about building the program on specific UNIX systems.

Testing the Permission Program

You can perform all testing outside the SAS/SHARE environment because the programs are stand- alone. The simplest way to test the programs is to look at the UNIX status variable in the UNIX shell. For example, using the C shell, you might test the **permission** program as follows:

```
%
permission /usr/bass/abc.ssd01 bass R

%
echo $status

0
%
```

A zero exit status means that user **bass** has read (**R**) access to the file **abc.ssd01**.

In the following test, because the exit status is non-zero, user **joe** does not have write access (**W**) to the file **abc.ssd01**.

```
%
permission /usr/joe/abc.ssd01 joe W

%
echo $status

1
%
```

After you test the program and are satisfied that it works correctly, move the program to the *!sasroot* **/utilities/bin** directory where SAS/SHARE expects the program to be located.

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