CHAPTER

41

Sign-on Script Files

General Script Statement Rules **505**

Syntax **505**

Sample Scripts **506**

TSO.SCR Script **507**

TCPUNIX.SCR Script **508**

LOGCMS.SCR Script **513**

TCPWIN.SCR Script **517**

TCPMVS.SCR Script **520**

General Script Statement Rules

This section contains general rules for writing script statements. A script statement that is not valid for your access method is ignored. For details about the statements that are used in the scripts, see *SAS/SHARE User's Guide*.

Syntax

- Like other SAS statements, all script statements must end with a semicolon(;).
- Script statements have a free format, which means that there are no spacing or indention requirements. You can split a statement across several lines, or you can put one or more statements on a single line. You can put statement keywords in uppercase, lowercase, or mixed case.
- Enclose case-sensitive text strings in quotation marks. For example, if your script defines a text string in a WAITFOR statement, be sure that the uppercase and lowercase characters in the text string exactly match the text string from the remote host.
- You can use either single or double quotation marks to enquote a string, such as a remote host command, in a script statement. The rules that you use to embed quotation marks in a SAS statement and to embed quotation marks in a script statement are the same.
- Any script statement can include a label specification. The label must be a valid SAS name, with a maximum of eight characters. The first character must be an alphabetic character or an underscore. A label must be followed immediately by a colon (:) and it can be defined only once in the script.
- Some script statements specify a time in seconds. The form of the time specification follows:

n SECONDS

where *n* can be any number, including decimal fractions. SECOND is an alias for SECONDS. Examples of valid time specifications follow:

0 SECONDS

0.25 SECONDS

1 SECOND

3.14 SECONDS.

The techniques that are used in these scripts are basic to writing scripts.

When writing a script or modifying an existing script, pay special attention to the WAITFOR and the TYPE statements. To ensure that the script recognizes the expected prompt during each stage of signing on, you must be careful to specify the exact sequence of prompts and responses for the remote host. The simplest method for determining sequence is to go to the remote host and to manually go through the process that you want to capture in the WAITFOR and the TYPE statements. For each display on the remote host, choose a word from that display for the WAITFOR statement. Whatever information that you type to respond to a display should be captured in the TYPE statement. Be sure to note all carriage returns or other special keys.

For example, if TSO is the remote host and you need to use a TYPE statement in a sign-on script whose length is greater than 80 characters, divide the TYPE statement into two or more TYPE statements. To divide the TYPE statement, insert a hyphen (-) at the division point. The remote TSO host interprets the hyphen as the continuation of the TYPE statement from the previous line. For example, consider the following TYPE statement:

```
type "sas options ('dmr comamid=tcp')" enter;
```

To divide the statements, change it to:

```
type "sas options ('dmr comamid=-" enter;
type "tcp')" enter;
```

Note: Do not insert spaces around the hyphen. △

Sample Scripts

The content and explanation follow for typical SAS/CONNECT scripts that sign on and sign off.

TSO.SCR
connects to an OS/390 remote host with the EHLLAPI 3270 access method.

TCPUNIX.SCR
connects to a UNIX remote host with the TCP/IP access method.

LOGCMS.SCR
connects to a CMS remote host with the EHLLAPI access method.

TCPWIN.SCR
connects to a Windows NT or a Windows 95 remote host with the TCP/IP access method.

TCPMVS.SCR
connects to an OS/390 remote host without TSO with the TCP/IP access method.

TSO.SCR Script

The following script signs on and signs off an OS/390 remote host with the EHLLAPI 3270 access method. This script assumes that the user has already logged on to an OS/390 remote host.

```
/* trace on; */
/* echo on;  */
/*-----------------------------------------------------------------------
   Copyright (C) 1996 by SAS Institute Inc., Cary NC
   name:      tso.scr
   purpose:   SAS/CONNECT SIGNON/SIGNOFF script for connecting to an
              OS/390 (with TSO) host using the EHLLAPI access method
              from a local WINDOWS or an OS/2 operating  system.
   assumes:   1. This script assumes the remote session is already
                 logged on.
              2. The command to execute SAS in your remote OS/390 (with TSO)
                 environment is "sas".  If this is incorrect for your
                 site, change the contents of the line that contains ...
                 type "sas ...

   support:   SAS Institute staff
 ---------------------------------------------------------------------*/

❶ log "NOTE: Script file 'tso.scr' entered.";

❷   if signoff then goto signoff;

/*---- EHLLAPI SIGNON -----------------------------------------------*/

❸
    waitfor 'READY', 0 seconds: noinit;
    log 'NOTE: Starting remote SAS now.';
    /* NOTERMINAL suppresses prompts from remote SAS session.     */
    /* NO$SYNTAXCHECK prevents remote side from going into syntax */
    /* checking mode when a syntax error is encountered.          */
    type "sas options('dmr,comamid=pclink,noterminal,no$syntaxcheck')" enter;
    goto continue;
❹ continue:
    waitfor 'IN PROGRESS', 20 seconds: waitsas;

❺ onok:
    log 'NOTE: SAS/CONNECT conversation established.';
    stop;

/*---- EHLLAPI SIGNOFF ----------------------------------------------*/

❻ signoff:
    log 'NOTE: SAS/CONNECT conversation terminated.';
    log 'NOTE: Remote session left logged on.';
    stop;

/*----- SUBROUTINES -------------------------------------------------*/
```

```
❼ waitsas:
    log 'NOTE: Waiting for startup screen...';
    type EREOF enter;
    goto continue;

/*----- ERROR HANDLING ----------------------------------------------*/

❽ noinit:
    snapshot;
    log 'ERROR: Did not get remote prompt.  Remote session not active.';
    log 'NOTE: You must log on to the remote session before signing on';
    log '      using this script file.';
    abort;
```

1 LOG statements write messages to the local host SAS log that provide information about the progress of the sign on.

2 If the SIGNOFF statement invokes this script, processing jumps to the SIGNOFF label. See step 6 in this list.

3 The WAITFOR statement looks for the READY prompt from the remote host and processing goes to step 8 if no prompt is received. When the READY prompt is received, the note is written to the local host's log, and the TYPE statement invokes SAS on the remote host. The DMR option is necessary to invoke a special processing mode for SAS/CONNECT. The COMAMID= option specifies the access method that is used to establish the connection.

4 The WAITFOR statement looks for the message IN PROGRESS, which is displayed when a SAS session starts on the remote host. If the message is not found, processing goes to the WAITSAS label in step 7.

5 This LOG statement prints a message to the log to indicate that you have successfully established a link.

6 These statements are executed when step 2 directs processing here. The two notes are written to the local log, and the link is terminated.

7 Processing continues from step 4. The note is written to the local log, and processing returns to step 4.

8 Processing continues from step 3 if the READY prompt is not received. The SNAPSHOT statement captures messages that are displayed on the remote host and writes them to the local host's log. The log messages can help you determine the reasons why the sign on to the remote host was interrupted. You can save or print the local log to have a record of these messages. The error and note messages are written to the local log, and the script aborts.

TCPUNIX.SCR Script

The following script connects to a UNIX remote host with the TCP/IP access method.

```
/* trace on; */
/* echo  on; */
/*------------------------------------------------------------------*/
/*--           Copyright (C) 1996 by SAS Institute Inc., Cary NC  --*/
/*--                                                              --*/
/*-- name:      tcpunix.scr                                       --*/
/*--                                                              --*/
```

```
/*-- purpose:   SAS/CONNECT SIGNON/SIGNOFF script for connecting   --*/
/*--            to any UNIX host by means of the TCP/IP access     --*/
/*--            method                                             --*/
/*--                                                               --*/
/*-- notes:   1. This script may need modifications that account   --*/
/*--             for the local flavor of your UNIX environment.    --*/
/*--             The logon procedure should mimic the events that  --*/
/*--             you go through when "telnet"-ing to the same      --*/
/*--             UNIX host. If you are connecting to a spawner     --*/
/*--             that is running in your UNIX environment, this    --*/
/*--             script should need little or no modifications.    --*/
/*--                                                               --*/
/*--          2. You must have specified OPTIONS COMAMID=TCP       --*/
/*--             in the local SAS session before using the SIGNON  --*/
/*--             statement.                                        --*/
/*--                                                               --*/
/*-- assumes: 1. The command to execute SAS in your remote (UNIX)  --*/
/*--             environment is "sas". If this is incorrect        --*/
/*--             for your site, change the contents of the line    --*/
/*--             that contains:                                    --*/
/*--             type ’sas ...                                     --*/
/*--                                                               --*/
/*-- support:   SAS Institute staff                                --*/
/*--                                                               --*/
/*-------------------------------------------------------------------*/


   /*----------------------------------------------------------------*/
   /*-- if you are connecting to DEC ULTRIX, and the remote        --*/
   /*-- machine does not run the DECnet connection/gateway         --*/
   /*-- software, logins by means of SAS/CONNECT will appear to    --*/
   /*-- hang. This is due to the ULTRIX "/etc/telnetd" server      --*/
   /*-- treating a DONT ECHO request for both input and output     --*/
   /*-- streams.                                                   --*/
   /*--                                                            --*/
   /*-- This DEBUG statement causes the SAS TCP/IP access method   --*/
   /*-- not to reply to the ECHO request, keeping the DEC telnetd  --*/
   /*-- server happy.                                              --*/
   /*--                                                            --*/
   /*-- Uncomment the DEBUG statement, if the logon appears to hang--*/
   /*----------------------------------------------------------------*/
   /* debug ’00001000’; */

   /*----------------------------------------------------------------*/
   /*-- If you are connecting to INTEL ABI, you need to uncomment  --*/
   /*-- the following DEBUG statement.  This DEBUG statement       --*/
   /*-- allows SAS/CONNECT to set the terminal type to TTY during  --*/
   /*-- the TELNET negotiations that take place during SIGNON.     --*/
   /*----------------------------------------------------------------*/
   /* debug ’00004000’; */


❶ log "NOTE: Script file ’tcpunix.scr’ entered.";
```

```
    if not tcp then goto notcp;
❷  if signoff then goto signoff;

     /* --------------- TCP/IP SIGNON ---------------------------------*/

❸  waitfor ’login:’
           , ’Username:’
           , ’Scripted signon not allowed’ : noscript
           , 120 seconds: noinit;

/*----------------UNIX LOGON----------------------------------------*/
/*-- for some reason, it needs an LF to turn the line around      --*/
/*--  after the login name has been typed. (CR will not do)       --*/
/*------------------------------------------------------------------*/

❹  input ’Userid?’;
     type LF;
❺  waitfor ’Password’, 30 seconds : nolog;
     input nodisplay ’Password?’;
     type LF;

unx_log:
❻  waitfor ’Hello>’               : unxspawn /*- UNIX spawner prompt-*/
           , ’$’               /*-- a common prompt character       --*/
           , ’>’               /*-- another common prompt character --*/
           , ’%’               /*-- another common prompt character --*/
           , ’}’               /*-- another common prompt character --*/
           , ’Login incorrect’      : nouser
           , ’Enter terminal type’  : unx_term
           , ’TERM’                 : unx_term
           , 30 seconds             : timeout
           ;


   log ’NOTE: Logged onto UNIX... Starting remote SAS now.’;
   /* NOTERMINAL suppresses prompts from remote SAS session.     */
   /* NO\$SYNTAXCHECK prevents remote side from going into       */
   /* syntax checking mode when a syntax error is encountered.   */
❼  type ’sas -dmr -comamid tcp -device grlink -noterminal -no\$syntaxcheck’ LF;
❽  waitfor ’SESSION ESTABLISHED’, 90 seconds : nosas;

❾  log ’NOTE: SAS/CONNECT conversation established.’;
   stop;

❿  unxspawn:
      /* The UNIX spawner executes only a single UNIX command     */
      /* after the client logs on.  In the TYPE statement below,  */
      /* you may specify a SAS command line. You may also specify */
      /* a UNIX shell script that issues the SAS command line in  */
      /* addition to any other commands to be executed prior to   */
      /* SAS invocation.  The following is a sample startup       */
      /* file:                                                    */
      /*#---------------------------------------------------------*/
      /*# sas_startup                                             */
      /*#---------------------------------------------------------*/
      /*#!/bin/ksh                                                */
      /*. ~/.profile                                              */
      /*sas -dmr -noterminal -no\$syntaxcheck -device grlink      */
      /*#---------------------------------------------------------*/
      /*                                                          */
      /* If you choose to use a "startup" file, change the TYPE   */
      /* statement below to something like the following:         */
      /* type '/usr/local/whatever/sas_startup' LF;               */

⓫ type 'sas -dmr -comamid tcp -device grlink -noterminal ';
     type '-no\$syntaxcheck' LF;

     waitfor 'SESSION ESTABLISHED', 90 seconds : nosas;
     stop;

/*---------------- TCP/IP SIGNOFF --------------------------------------*/
signoff:
    /* If you have established your connection to UNIX by means */
    /* of a UNIX spawner, you should delete or comment the      */
    /* following WAITFOR and TYPE statements.  They are not     */
    /* necessary for signing off a UNIX spawner and will        */
    /* result in slower performance of SIGNOFF.                 */
⓬ waitfor '$'
           , '>'                /*-- another common prompt character --*/
           , '%'                /*-- another common prompt character --*/
           , '}'                /*-- another common prompt character --*/
           , 30 seconds
           ;

   type    'logout' LF;
   log 'NOTE: SAS/CONNECT conversation terminated.';
   stop;

/*--------------- SUBROUTINES -----------------------------------*/

unx_term:
/*---------------------------------------------------------------*/
/*-- Some UNIX platforms want the terminal type,               --*/
/*-- so tell them we are the most basic of terminals.          --*/
/*---------------------------------------------------------------*/
   type 'tty' LF;
   goto unx_log;


/*--------------- ERROR ROUTINES --------------------------------*/


⓭ timeout:
     log 'ERROR: Timeout waiting for remote session response.';
     abort;

     nouser:
     log 'ERROR: Unrecognized userid or password.';
   abort;

notcp:
   log 'ERROR: Incorrect communications access method.';
   log 'NOTE: You must set "OPTIONS COMAMID=TCP;" before using this';
   log '      script file.';
   abort;

noinit:
   log 'ERROR: Did not understand remote session banner.';

nolog:
   log 'ERROR: Did not receive userid or password prompt.';
   abort;

nosas:
  log 'ERROR: Did not get SAS software startup messages.';
   abort;

noscript:
   /* This is the result of trying to sign on with a script file */
   /* to a UNIX spawner that has been invoked with the -NOSCRIPT */
   /* option.  You need to clear any script file reference and   */
   /* then re-execute SIGNON.                                     */
   log 'ERROR: Scripted signons are not allowed.';
   log 'NOTE:  Clear any script file reference and retry SIGNON.';
   abort;
```

1 The LOG statement sends the enquoted message to the log file or to the LOG window of the local SAS session. Although it is not necessary to include LOG statements in your script file, the LOG statements keep the user informed about the progress of the connection.

2 The IF/THEN statement can detect whether the script was called by the SIGNON statement or the SIGNOFF statement. When you are signing off, the IF/THEN statement directs script processing to the statement labeled SIGNOFF. See step 12.

3 The WAITFOR statement awaits the login prompt from the remote host. If the statement does not receive the prompt within 120 seconds, it directs script processing to branch to the statement labeled NOINIT.

4 The INPUT statement displays a window with the text **`Userid?`** to allow the user to enter a remote host logon userid. The TYPE statement sends a line feed to the remote host to enter the userid to the remote host.

5 The WAITFOR statement waits for the password prompt from the remote host and branches to the NOLOG label if it is not received within 30 seconds. The INPUT statement that follows the WAITFOR statement displays a window in which the user enters a password.

6 The WAITFOR statement waits for one of several common UNIX prompts and branches to various error handles if a prompt is not displayed. For a connection to the UNIX spawner, the string "Hello >" is received and the control branches to the **`unxspawn`** label in step 10. Verify that the WAITFOR statement in the script looks for the correct prompt for your site.

7 The TYPE statement invokes SAS on the remote host. The DMR option is necessary to invoke a special processing mode for SAS/CONNECT. The

COMAMID= option specifies the access method that is used to make the connection.

8 The message **SESSION ESTABLISHED** is displayed when a SAS session is started on the remote host with the DMR and the COMAMID=TCP options. The WAITFOR statement awaits the display of the message **SESSION ESTABLISHED** to be issued by the remote host. If the **SESSION ESTABLISHED** response is received within 90 seconds, processing continues with the next LOG statement. If the **SESSION ESTABLISHED** response does not occur within 90 seconds, the script assumes that the remote SAS session has not started, and processing branches to the statement labeled NOSAS.

9 After the connection has been successfully established, the user must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

10 This section of code is executed when you connect to a remote UNIX spawner.

11 The TYPE statement invokes SAS on the remote host. The DMR option is necessary to invoke a special processing mode for SAS/CONNECT. The COMAMID= option specifies the access method that is used to make the connection.

12 This section of code is executed when the script is invoked to terminate the link. The IF statement (see step 2) sends processing to this section of the script when the script is invoked by a SIGNOFF statement. This section logs the user off the remote host after the user executes **LOGOFF**. Before it stops the link, the script issues a LOG statement to notify the user that the link is terminated.

13 These statements are processed only if the prompts expected in the previous steps are not received. This section of the script issues messages to the local SAS log and then abnormally ends the script processing as well as the SIGNON.

LOGCMS.SCR Script

The following script signs on and signs off a CMS remote host with the EHLLAPI access method.

```
/* trace on;  */
/* echo on;   */
/*------------------------------------------------------------------
   Copyright (C) 1996 by SAS Institute Inc., Cary NC
   name:      logcms.scr
   purpose:   SAS/CONNECT SIGNON/SIGNOFF script for automatically
              connecting to a CMS host using the EHLLAPI access method
              from a local WINDOWS or an OS/2 operating system.
   notes:     1. This script may need modifications that account for
                 the local flavor of your VM/CMS environment.  The
                 SIGNON sequence should mimic the events that you go
                 through when logging on to the same VM/CMS host.
   assumes:   1. The command to execute SAS in your remote (CMS)
                 environment is "sas".  If this is incorrect for your
                 site, change the contents of the line that contains...
                 type "sas ...
   support:   SAS Institute staff
 ------------------------------------------------------------------*/

❶  log "NOTE: Script file 'logcms.scr' entered.";
```

```
❷ if signoff then goto signoff;

/*----- EHLLAPI SIGNON ---------------------------------------------*/

❸ waitfor 'SAS Institute Inc.', 0 seconds: noinit;
     type 'vm ' enter;
❹ waitfor 'USERID' , 20 seconds: nolog;
     input 'Enter userid: ';
     type enter;
❺ waitfor 'ENTER PASSWORD',
             'Enter logon password',
             'SESSION NOT BOUND'      : novtam,
             'NOT IN CP DIR'          : nouser,
             20 seconds               : nolog;
     input nodisplay 'Enter password: ';
     type enter;

❻ waitrdy:
   waitfor 'R;',
             'Ready;',
             'READY',
             'INCORRECT PASSWORD'     : nopass,
             'RECONNECT'              : recon32,
             'UNSUCCESSFUL'           : nostrt,
             'MORE...'                : more,
             30 seconds               : nostrt;
  ready32:
     type clear;
     waitfor 'VM READ', 'RUNNING', 20 seconds: timeout;

     log 'NOTE: Logged on to CMS ... Starting remote SAS now.';
     /* NOTERMINAL suppresses prompts from remote SAS session.    */
     /* NO$SYNTAXCHECK prevents remote side from going into syntax */
     /* checking mode when a syntax error is encountered.          */
❼ type 'sas (dmr comamid=pclink noterminal no$syntaxcheck)' enter;
     goto continue;
  continue:
❽ waitfor 'IN PROGRESS', 60 seconds: nosas;
❾ log 'NOTE: SAS/CONNECT conversation established.';
   stop;

/*----- EHLLAPI SIGNOFF --------------------------------------------*/

signoff:
❿ type 'logoff' enter;
     waitfor 'Press enter key', 10 seconds: noterm;
     type enter;
     waitfor 'SAS Institute Inc.' , 10 seconds: noterm;
     type tab;
     type tab;
     type 'vmexit' enter;
     waitfor 'SAS Institute Inc.', 10 seconds: noterm;
⓫    log 'NOTE: SAS/CONNECT conversation terminated.';
     stop;
```

```
/*----- SUBROUTINES -------------------------------------------------*/

⓬ recon:
    type 'begin' enter;
    waitfor 'RUNNING', 'VM READ', 20 seconds: timeout;
    type 'hx' enter;
    goto ready32;

 more:
    snapshot;
    type clear;
    goto waitrdy;

/*----- ERROR HANDLING ----------------------------------------------*/

⓭ noinit:
   snapshot;
    log 'ERROR: No data center network screen.';
    abort;

 novtam:
    snapshot;
    log 'ERROR: VTAM session not bound - contact VTAM support.';
    abort;

 nolog:
    snapshot;
    log 'ERROR: Did not get userid or password prompt.';
    abort;

 nouser:
    log 'ERROR: Unrecognized userid.';
    goto kill;

 nopass:
    log 'ERROR: Invalid password.';
    goto kill;

 nostrt:
    snapshot;
    log 'ERROR: Did not get CMS startup messages after logon.';
    goto kill;

 nosas:
    snapshot;
    log 'ERROR: Did not get SAS software startup messages.';
    goto kill;

 noterm:
    snapshot;
    log 'WARNING: Did not get messages confirming logoff.';
    goto kill;
```

```
timeout:
   log 'ERROR: Timeout waiting for remote session response.';

kill:
   snapshot;
   type '#cp logoff' enter;
   abort;
```

1 The LOG statement issues the enquoted message in the log file or in the LOG window of the local SAS session. Although it is not necessary to include LOG statements in your script file, the LOG statements keep the user informed about the progress of the connection.

2 The IF/THEN statement detects whether the script was called by the SIGNON command or statement or by the SIGNOFF command or statement. When you sign off, the IF/THEN statement directs script processing to the statement that is labeled SIGNOFF.

3 This section logs the user onto CMS with the EHLLAPI access method.

4 The WAITFOR statement awaits the login prompt from the remote host. If the statement does not receive the prompt within 20 seconds, it directs script processing to branch to the statement that is labeled NOLOG. The INPUT statement displays a window with the text **`Enter userid:`** to allow the user to enter a remote host logon userid. The TYPE statement sends an ENTER to the remote host to enter the userid to the remote host.

5 The WAITFOR statement awaits the password prompt from the remote host. If the statement does not receive the prompt within 20 seconds, it directs script processing to branch to the statement that is labeled NOLOG. The INPUT statement that follows the WAITFOR statement displays a window in which the user enters a password.

6 The WAITFOR statement looks for the READY prompt from the remote host, and processing goes to the NOSTRT label if no prompt is received.

7 When the READY prompt is received, the TYPE statement invokes SAS on the remote host. The DMR option is necessary to invoke a special processing mode for SAS/CONNECT. The COMAMID= option specifies the PCLINK access method that is used to establish the connection.

8 The WAITFOR statement looks for the message IN PROGRESS, which is displayed when a SAS session starts on the remote host. If the message is not found, processing goes to the NOSAS label.

9 This LOG statement prints a message to the log that indicates that you have successfully established a link. After the connection has been successfully established, you must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

10 This is the SIGNOFF section for the EHLLAPI connection. See step 2. The TYPE statement logs you off CMS.

11 This section issues a LOG statement to notify the user that the link is terminated. After the connection has been terminated, you must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

12 These are subroutines that handle various conditions that are found in the WAITFOR statements throughout the script file.

13 These statements are processed only if the prompts that are expected in the previous steps are not received. This section of the script issues messages to the local SAS log and then abnormally ends the script processing as well as the SIGNON.

TCPWIN.SCR Script

The following script signs on and signs off a Windows NT or a Windows 95 remote host with the TCP/IP access method.

```
/* trace on; */
/* echo  on; */
/*-------------------------------------------------------------------*/
/*--            Copyright (C) 1996 by SAS Institute Inc., Cary NC  --*/
/*--                                                               --*/
/*-- name:      tcpwin.scr                                         --*/
/*--                                                               --*/
/*-- purpose:   SAS/CONNECT SIGNON/SIGNOFF script for connecting   --*/
/*--            to either a Windows 95 or a Windows NT host by     --*/
/*--            means of the TCP/IP access method.                 --*/
/*--                                                               --*/
/*-- notes:   1. You must have the spawner program executing on    --*/
/*--             the remote Windows 95 or Windows NT workstation   --*/
/*--             in order for the local session to be able to      --*/
/*--             establish the connection.  If the spawner is      --*/
/*--             running on the remote node, you will receive a    --*/
/*--             message telling you that the connection has       --*/
/*--             been refused.                                     --*/
/*--                                                               --*/
/*--          2. You must have specified OPTIONS COMAMID=TCP       --*/
/*--             in the local SAS session before using the SIGNON  --*/
/*--             command.                                          --*/
/*--                                                               --*/
/*-- assumes: 1. The command to execute SAS in your remote         --*/
/*--             (Windows 95 or Windows NT) environment is "sas".  --*/
/*--             If this is incorrect for your site, change the    --*/
/*--             contents of the line that contains:               --*/
/*--             type ’sas ...                                     --*/
/*--                                                               --*/
/*-- support:   SAS Institute staff                                --*/
/*--                                                               --*/
/*-------------------------------------------------------------------*/

❶ log "NOTE: Script file ’tcpwin.scr’ entered.";

     if not tcp then goto notcp;
❷ if signoff then goto signoff;

     /* --------------- TCP/IP SIGNON ---------------------------------*/

❸ waitfor ’Username:’
           , ’Hello>’                 :  ready
           , ’access denied’          :  nouser
           , 120 seconds              :  noprompt
           ;
```

```
❹ input 'Userid?';
     type LF;
❺ waitfor 'Password:' , 120 seconds: nolog;
     input nodisplay 'Password?';
     type LF;

❻ waitfor 'Hello>'
          , 'access denied'         :  nouser
          , 120 seconds             :  timeout
          ;

  ready:
     log 'NOTE: Logged onto Windows... Starting remote SAS now.';
     /* NOTERMINAL suppresses prompts from remote SAS session.    */
     /* NO$SYNTAXCHECK prevents remote side from going into syntax */
     /* checking mode when a syntax error is encountered.          */
❼ type 'sas -dmr -comamid tcp -device grlink -noterminal -no$syntaxcheck' LF;
❽ waitfor 'SESSION ESTABLISHED', 120 seconds : nosas;

❾ log 'NOTE: SAS/CONNECT conversation established.';
     stop;


/*---------------- TCP/IP SIGNOFF -----------------------------------*/

❿ signoff:
     log 'NOTE: SAS/CONNECT conversation terminated.';
     stop;

/*--------------- SUBROUTINES -----------------------------------*/


/*--------------- ERROR ROUTINES --------------------------------*/
⓫
  notcp:
     log 'ERROR: Incorrect communications access method.';
     log 'NOTE: You must set "OPTIONS COMAMID=TCP;" before using this';
     log '      script file.';
     abort;

  noprompt:
     log 'ERROR: Did not receive userid prompt.';
     log 'NOTE:  Ensure spawner process is running on remote node.';
     abort;

  nolog:
     log 'ERROR: Did not receive password prompt.';
     abort;


  nouser:
     log 'ERROR: Unrecognized userid or password.';
     abort;
```

```
nosas:
   log 'ERROR: Did not get SAS software startup messages.';
   abort;

timeout:
   log 'ERROR: Timeout waiting for remote session response.';
   abort;
```

1 The LOG statement sends the enquoted message to the log file or to the LOG window of the local SAS session. Although it is not necessary to include LOG statements in your script file, the LOG statements keep the user informed about the progress of the connection.

2 The IF/THEN statement detects whether the script was called by the SIGNON statement or by the SIGNOFF statement. When you sign off, the IF/THEN statement directs script processing to the statement that is labeled SIGNOFF. See step 10.

3 The WAITFOR statement awaits the login prompt from the remote host and branches to various error handles if this prompt is not displayed.

4 The INPUT statement displays a window with the text **`Userid?`** to allow the user to enter a remote host logon userid. The TYPE statement sends a line feed to the remote host to enter the userid to the remote host.

5 The WAITFOR statement awaits the password prompt from the remote host and branches to the NOLOG label if it is not received within 120 seconds. The INPUT statement that follows the WAITFOR statement displays a window in which the user enters a password.

6 The WAITFOR statement awaits the "Hello > " prompt that it expects to see from the PC spawner. If the statement does not receive the prompt within 120 seconds, it directs script processing to branch to the statement that is labeled TIMEOUT.

7 The TYPE statement invokes SAS on the remote host. The DMR option is necessary to invoke a special processing mode for SAS/CONNECT. The COMAMID= option specifies the access method that is used to make the connection.

8 The message **`SESSION ESTABLISHED`** is displayed when a SAS session is started on the remote host with the DMR and COMAMID=TCP options. The WAITFOR statement awaits the display of the message **`SESSION ESTABLISHED`** to be issued by the remote host. If the **`SESSION ESTABLISHED`** response is received within 120 seconds, processing continues with the next LOG statement. If the **`SESSION ESTABLISHED`** response does not occur within 120 seconds, the script assumes that the remote SAS session has not started and processing branches to the statement labeled NOSAS.

9 After the connection has been successfully established, the user must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

10 This section of code is executed when the script is invoked to terminate the link. The IF statement (see step 2) sends processing to this section of the script when the script is invoked by a SIGNOFF statement. Before it stops the link, the script issues a LOG statement to notify the user that the link is terminated.

11 These statements are processed only if the prompts expected in the previous steps are not received. This section of the script issues messages to the local SAS log and then abnormally ends the script processing as well as the SIGNON.

TCPMVS.SCR Script

The following script signs on and signs off an OS/390 remote host with TSO by using the TCP/IP access method.

```
/*-------------------------------------------------------------------*/
/*--            Copyright (C) 1990 by SAS Institute Inc., Cary NC  --*/
/*--                                                               --*/
/*-- name:      tcpmvs.scr                                         --*/
/*--                                                               --*/
/*-- purpose:   SAS/CONNECT SIGNON/SIGNOFF script for connecting   --*/
/*--            to any OS/390 host with the TCP/IP access method   --*/
/*--                                                               --*/
/*-- notes:   1. This script may need modifications that account   --*/
/*--             for the local flavor of your OS/390 environment.  --*/
/*--             The logon procedure should mimic the events that  --*/
/*--             you go through when "telnet"-ing to the same      --*/
/*--             OS/390 host, either to TSO or to the OS/390       --*/
/*--             spawner.                                          --*/
/*--                                                               --*/
/*--          2. You must have specified OPTIONS COMAMID=TCP       --*/
/*--             in the local SAS session before using the SIGNON  --*/
/*--             command.                                          --*/
/*--                                                               --*/
/*--          3. This script supports two flavors of connection:   --*/
/*--             through a TSO session whose logon procedure       --*/
/*--             invokes SAS directly rather than the TSO TMP, or  --*/
/*--             through the OS/390 spawner.                       --*/
/*--                                                               --*/
/*--          4. If you use TSO to start the SAS session, in the   --*/
/*--             signoff portion of the script, uncomment the      --*/
/*--             LOGOFF command to complete session termination.   --*/
/*--                                                               --*/
/*--          5. If you use the OS/390 spawner to start the SAS    --*/
/*--             session, and the client session is running a      --*/
/*--             release prior to 6.09E or 6.11 TS040, uncomment   --*/
/*--             the ’type CR LF;’ statements after the prompts.   --*/
/*--                                                               --*/
/*-- support:   SAS Institute staff                                --*/
/*--                                                               --*/
/*-------------------------------------------------------------------*/

❶ log "NOTE: Script file ’tcpmvs.scr’ entered.";

❷ if not tcp then goto notcp;
   if signoff then goto signoff;

/* ------------------------- TCP/IP SIGNON --------------------------*/
/* make sure we are running the IBM TCP/IP or the OS/390 spawner      */
❸ waitfor ’Userid: ’                      : spnlogon,
           ’ENTER USERID’                 : tsologon,
           120 seconds                    : noinit;

/*------------------------- SPAWNER LOGON ---------------------------*/
```

```
spnlogon:
❹ input 'Userid?';
/* type CR LF; */

❺ waitfor 'Password',
           120 seconds                    : spnfail;

   input nodisplay 'Password?';
/* type CR LF; */

spndone:
❻ waitfor 'Options',
           'Userid'                       : spnlogon,
           'Password expired'             : spnnewp,
           120 seconds                    : spnfail;

❼ type "DMR NOTERMINAL NO$SYNTAXCHECK COMAMID=TCP";
/* type CR LF; */

❽ waitfor 'SESSION ESTABLISHED',
           120 seconds                    : spnfail;

❾ log 'NOTE: SAS/CONNECT conversation established.';
   stop;


spnnewp:
 ❿ input nodisplay 'New Password?';
/* type CR LF; */

   waitfor 'Verify new password',
           120 seconds                    : spnfail;

   input nodisplay 'Verify New Password';
/* type CR LF; */

   goto spndone;

spnfail:
   log 'ERROR: Invalid SPAWNER prompt message received.';
   abort;

/*-------------------------- TSO LOGON -----------------------------*/

tsologon:
 ⓫ input 'Userid?';
   type LF;

⓬ waitfor 'ENTER PASSWORD',
           120 seconds                    : nolog;

tsopass:
   input nodisplay 'Password?';
   type LF;

tsodone:
 ⓭ waitfor 'SESSION ESTABLISHED',
           'PASSWORD INVALID'             : tsopass,
           'ENTER NEW PASSWORD'           : tsonewp,
           'CURRENTLY LOGGED ON'          : dup_log,
           'NOT VALID'                    : nouser,
           120 seconds                    : notso;
   waitfor 1 second;

⓮ log 'NOTE: SAS/CONNECT conversation established.';

   stop;

tsonewp:
 ⓯ input nodisplay 'New Password?';
   type LF;

   waitfor 'VERIFY NEW PASSWORD',
           120 seconds                    : notso;

   input nodisplay 'Verify New Password';
   type LF;

   goto tsodone;

/*----------------------------- SIGNOFF ------------------------------*/

⓰ signoff:

/* ------------ for TSO, uncomment the following section ------------*/
   type 'logoff' LF;
   waitfor 'LOGGED OFF'                   : logoff,
           20 seconds;

   log 'WARNING: Did not get messages confirming logoff.';
   abort;

logoff:
/* ------------ for TSO, uncomment the previous section ------------ */

   log 'NOTE: SAS/CONNECT conversation terminated.';
   stop;

/* ----------------------- TSO ERROR ROUTINES ------------------------*/

⓱ nouser:
   log 'ERROR: Unrecognized userid.';
   abort;

nopass:
   log 'ERROR: Invalid password.';
   abort;
```

```
notcp:
   log ’ERROR: Incorrect communications access method.’;
   log ’NOTE: You must set "OPTIONS COMAMID=TCP;" before using this’;
   log ’      script file.’;
   abort;

noinit:
   log ’ERROR: Did not understand remote session banner.’;
   abort;

nolog:
   log ’ERROR: Did not get userid or password prompt.’;
   abort;

notso:
   log ’ERROR: Did not get TSO startup messages after logon.’;
   abort;

dup_log:
   log ’ERROR: User is already logged onto TSO.’;
   abort;
```

1 The LOG statement sends the enquoted message to the log file or to the LOG window of the local SAS session. Although it is not necessary to include LOG statements in your script file, the LOG statements keep the user informed about the progress of the connection.

2 The IF/THEN statement detects whether the script was called by the SIGNON statement. When you are signing off, the IF/THEN statement directs script processing to the statement labeled SIGNOFF. See step 8.

3 The WAITFOR statement awaits the login prompt from the remote host. If the statement does not receive the prompt within 120 seconds, it directs script processing to branch to the statement labeled NOINIT.

4 The INPUT statement displays a window with the text **Userid?** to allow the user to enter a remote host logon userid. The TYPE statement sends a line feed to the remote host to enter the userid to the remote host.

5 The WAITFOR statement waits for the password prompt from the remote host and branches to the NOLOG label if it is not received within 120 seconds. The INPUT statement that follows the WAITFOR statement displays a window for the user to enter a password.

6 The WAITFOR statement awaits the prompt for SAS options and branches to various condition handlers if this prompt is not received.

7 The TYPE statement sends the appropriate options for the SAS/CONNECT session.

8 The message **SESSION ESTABLISHED** is displayed when a SAS session is started on the remote host with the DMR and COMAMID=TCP options. The WAITFOR statement awaits the display of the message **SESSION ESTABLISHED** to be issued by the remote host. If the **SESSION ESTABLISHED** response is received within 120 seconds, processing continues with the next LOG statement. If the **SESSION ESTABLISHED** response does not occur within 120 seconds, the script assumes that the remote SAS session has not started and processing branches to the statement labeled NOSTRT.

9 After the connection has been successfully established, the user must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

10 This section prompts for a new password if the password has expired.

11 The INPUT statement displays a window with the text **Userid?** to allow the user to enter a remote host logon userid. The TYPE statement sends a line feed to the remote host to enter the userid to the remote host.

12 The WAITFOR statement waits for the password prompt from the remote host and branches to the NOLOG label if it is not received within 120 seconds. The INPUT statement that follows the WAITFOR statement displays a window for the user to enter a password.

13 The message **SESSION ESTABLISHED** is displayed when a SAS session is started on the remote host with the DMR and COMAMID=TCP options. The WAITFOR statement awaits the display of the message **SESSION ESTABLISHED** to be issued by the remote host. If the **SESSION ESTABLISHED** response is received within 120 seconds, processing continues with the next LOG statement. If the **SESSION ESTABLISHED** response does not occur within 120 seconds, the script assumes that the remote SAS session has not started and processing branches to the statement labeled NOSTRT.

14 After the connection has been successfully established, the user must stop the rest of the script from processing. Without this STOP statement, processing continues through the remaining statements in the script.

15 This section prompts for a new password if the password has expired.

16 This section of code is executed when the script is invoked to terminate the link. The IF statement (see step 2) sends processing to this section of the script when the script is invoked by a SIGNOFF statement. This section awaits a remote host prompt before displaying **LOGOFF**, which logs the user off the remote host. Before it stops the link, the script issues a LOG statement to notify the user that the link is terminated.

17 These statements are processed only if the prompts expected in the previous steps are not received. This section of the script issues messages to the local SAS log and then abnormally ends the script processing as well as the SIGNON.

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.