```
   CCCCCCCCCCCC  LLL                  IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                  IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                  IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
CCC              LLL                     III        UUU         UUU        TTT        LLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL      IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL      IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL      IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
```

FILEID**SETPRO

```
 SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      OOOOOO
 SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      OOOOOO
SS         EE              TT      PP      PP  RR      RR  OO      OO
SS         EE              TT      PP      PP  RR      RR  OO      OO
SS         EE              TT      PP      PP  RR      RR  OO      OO
SS         EE              TT      PP      PP  RR      RR  OO      OO
 SSSSSS    EEEEEEEE        TT      PPPPPPPP    RRRRRRRR    OO      OO
 SSSSSS    EEEEEEEE        TT      PPPPPPPP    RRRRRRRR    OO      OO
       SS  EE              TT      PP          RR  RR      OO      OO
       SS  EE              TT      PP          RR  RR      OO      OO
       SS  EE              TT      PP          RR    RR    OO      OO        ....
       SS  EE              TT      PP          RR    RR    OO      OO        ....
SSSSSSSS   EEEEEEEEEE      TT      PP          RR      RR    OOOOOO          ....
SSSSSSSS   EEEEEEEEEE      TT      PP          RR      RR    OOOOOO          ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE setpro ( IDENT = 'V04-000',
0                 ADDRESSING_MODE (EXTERNAL = GENERAL)) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:  SETPRO Command
1 !
1 ! ABSTRACT:
1 !
1 !       This utility sets protection for files and directories.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX/VMS operating system. Privileged user mode,
1 !
1 ! AUTHOR:  Greg Robert, Nov 1979
1 !
1 ! Modified by:
1 !
1 !       V03-006 GAS0112         Gerry Smith             29-Mar-1983
1 !               Remove references to old command dispatcher.
1 !
1 !       V03-005 GAS0100         Gerry Smith             12-Mar-1983
1 !               Remove all references to old CLI interface.
1 !
1 !       V03-004 GAS0094         Gerry Smith             19-Nov-1982
1 !               Fix /LOG qualifier.
1 !
1 !       V03-003 GAS0093         Gerry Smith             27-Oct-1982
1 !               Fix protection parse algorithm.  Also, change the
1 !               protection descriptor to a dynamic one, and let the
1 !               system de/allocate memory as appropriate.
1 !
```

```
1 !       V03-002 GAS0092         Gerry Smith            22-Oct-1982
1 !               Hybridize the CLI interface, so that if /DEVICE is
1 !               not specified, use the new interface.  This fixes a
1 !               problem that keeps SET PROT=prot from working correctly.
1 !
1 !       V03-001 GAS0091         Gerry Smith            19-Oct-1982
1 !               Change input request for new CLD syntax.
1 !
1 !       V02-006 MLJ0066         Martin L. Jack, 29-Dec-1981  13:36
1 !               Integrate new LIB$SET_FILE_PROT (formerly LIB$SET_PROT).
1 !
1 !       V205    GRR2005         G. R. Robert           17-Nov-1981
1 !               Fixed handling of filespecs in confirmation and logging
1 !               messages to use expanded string if resultant not available,
1 !               and to use RMS parse to insure that passwords are blanked.
1 !
1 !       V204    GRR2004         G. R. Robert           16-Nov-1981
1 !               Made all external references addressing mode general.
1 !
1 !       V203    GRR2003         G. R. Robert           14-Sep-1981
1 !               Fixed improper descriptor initialization in confirm_action
1 !               by making them OWN instead of LOCAL.
1 !
1 !       002     GRR2002         Greg Robert     12-Jun-1981
1 !               Added non-fatal error message to continue
1 !               processing files after a fatal error.
1 !
1 !       001     TMH0001         Tim Halvorsen   21-Jan-1980
1 !               Add dispatching for SET PROT/DEVICE to SETDEV and include
1 !               code from require file into here and obsolete require file.
1 !--
1
1 !
1 ! Include files
1 !
1
1 LIBRARY 'SYS$LIBRARY:STARLET.L32';          ! VAX/VMS common definitions
```

```
!
! Table of contents
!

FORWARD ROUTINE
    set$protection : NOVALUE,           ! Main setpro routine
    setpro_action,                      ! Called for each file
    get_output_file,                    ! Get next output file
    confirm_action,                     ! Interrogate user
    log_results,                        ! Inform user of results
    expand_prot,                        ! Converts binary prot to ascii
    parse_class,                        ! Parse protection of one user class
    search_error,                       ! LIB$FILE_SCAN error handler
    cli_error;                          ! Handler for CLI errors

!
! External routines
!

EXTERNAL ROUTINE
    cli$get_value,                      ! Get value from CLI interface
    cli$present,                        ! Test for presence of qualifier
    sys$fao,                            ! Formats ascii output
    sys$setprv,                         ! Set privileges
    lib$get_command,                    ! Reads from SYS$COMMAND
    lib$file_scan,                      ! Handles stickiness etc
    lib$set_file_prot,                  ! Set file protection
    set$device;                         ! SET PROT/DEVICE action routine

!
! External data references
!

EXTERNAL LITERAL
    set$_pronotchg;             ! Error message - "Protection not changed"
```

```
 1
 1 !
 1 !       DEFINE VMS BLOCK STRUCTURES
 1 !
 1
 1 STRUCTURE
 1         BBLOCK [O, P, S, E; N] =
 1                 [N]
 1                 (BBLOCK+O)<P,S,E>;
```

```
MACRO
!
! A) Macro to describe a string
! B) Macro to generate a quadword string descriptor
! C) Macro to generate the address of a string descriptor

   PRIMDESC (str) = %CHARCOUNT (str), UPLIT (%ASCII str)%,
   INITDESC (str) = BBLOCK [DSC$C_S_BLN] INITIAL (PRIMDESC (str))%,
   ADDRDESC (str) = UPLIT (PRIMDESC (str))%,

!
!       Macro to signal a condition to the handler
!

   write_message(msg) =
       SIGNAL(msg                           ! Pass the message code
       %IF %LENGTH GTR 1                    ! and if more than 1 arg
           %THEN ,%REMAINING %FI) %;        ! then the rest too
```

```
  1 !
  1 !       Define message codes.
  1 !
  1
P 1 $shr_msgdef(set,119,local,
P 1         (protected,info),       ! File protection changed
P 1         (syntax,severe),        ! Failure in parse routines
  1         (searchfail,error));    ! Could not find file
  1
```

```
!
!       EQUATED SYMBOLS
!

LITERAL
    TRUE        = 1,                        ! BOOLEAN TRUE
    FALSE       = 0,                        ! BOOLEAN FALSE
    OK          = 1,                        ! SUCCESS RETURN CODE
    ERROR       = 2;                        ! ERROR RETURN CODE



!
!       Define CLI qualifier bit flag numbers (see setpro$flags)
!

LITERAL
    $EQULST (QUAL_,,0,1,
        (protection,),              ! /PROTECTION
        (log,),                     ! /LOG
        (confirm,),                 ! /CONFIRM
        (device,));                 ! /DEVICE
```

```
!
! Storage definitions
!

OWN
    setpro$flags         : BITVECTOR[32] ! General DCL flagword
        INITIAL(0),                      ! Initially none present

    qualifier$flags      : BITVECTOR[32] ! Qualifier presence bitmap
        INITIAL(0),                      ! Initially clear

    setpro_prot          : WORD          ! Contains /PROTECTION value
                         INITIAL(0),     ! In case user gives no value

    setpro_mask          : WORD          ! Contains /PROTECTION mask
                         INITIAL(0),     ! In case user gives no value

    global_mask          : WORD,         ! Command wide mask
    global_prot          : WORD,         ! Command wide protection

    result_mask          : WORD,         ! Combined mask
    result_prot          : WORD,         ! Combined protection

    oldpriv              : BBLOCK[8],    ! Permanent priv's stored here

    newpriv              : BBLOCK[8]     ! Mask describing system priv
                PRESET ([prv$v_sysprv]=true), ! Initialize this bit

    worst_error: BBLOCK[4]               ! Worst error encountered
                INITIAL(ss$_normal);     ! Initially normal status
```

```
!
! Define RMS blocks
!

OWN
    output_filename:                    ! Place to store the file name
            VECTOR[nam$c_maxrss,BYTE],

    output_nam_result:                  ! Resultant output name
            VECTOR [nam$c_maxrss,BYTE],

    output_nam_expanded:                ! Expanded output name
            VECTOR [nam$c_maxrss,BYTE],

    output_rlf: $NAM(                   ! Related name block
            ),

    output_nam: $NAM(                   ! File name block
        RLF = output_rlf,               ! Point to related name block
        ESA = output_nam_expanded,      ! File name before open
        ESS = nam$c_maxrss,
        RSA = output_nam_result,        ! File name after open
        RSS = nam$c_maxrss),

    output_fab: $FAB(                   ! FAB for output
        NAM = output_nam);              ! Address of name block
```

```
1 GLOBAL ROUTINE set$protection : NOVALUE =
1
1 !++
1 ! Functional description
1 !
1 !       This is the main control routine for the SET PROTECTION cmd.
1 !       It is called from the command language interpreter to
1 !       set protection for files and directories.
1 !
1 ! Calling sequence
1 !
1 !       set$protection (option_desc_block) from the CLI as an
1 !       action routine for the /PROTECTION OPTION of the SET command
1 !
1 ! Input parameters
1 !
1 !       None
1 !
1 ! Output parameters
1 !
1 !       None
1 !
1 ! Routine value
1 !
1 !       Worst error encountered during processing or SS$_NORMAL.
1 !
1 !----
1
2 BEGIN
2
2 LOCAL
2     prot_desc : $BBLOCK[dsc$c_s_bln],     ! Protection descriptor
2     status;                               ! Status return
2
2
2 !
2 ! The SET command is installed with system privileges (SYSPRV), however
2 ! we don't want the user to have this much power unless s/he
2 ! explicitly asks for it.  Therefore we will inspect the existing
2 ! privileges.  If SYSPRV is true on a process permanent basis we will
2 ! leave it alone, but if only enabled on a temporary basis (for this
2 ! image) then we will disable the privilege
2 !
2
2 !
2 ! First we get the permanent privileges
2 !
2
3 IF NOT (status =
3 SYS$SETPRV      (1,                 ! Enable
3                 0,                  ! No mask supplied
3                 1,                  ! Permanent privileges
3                 oldpriv)            ! Place to deposit current privileges
3 ) THEN BEGIN
3       write_message (.status);            ! Tell the user whats wrong
3       return;                             ! And give up
2       END;
```

```
2
2 !
2 ! Now decide whether or not to clear the privilege
2 !
2
3 IF NOT (.oldpriv[prv$v_sysprv]) ! If privilege NOT permanent
3 THEN BEGIN
4         IF NOT (status =
4         SYS$SETPRV      (0,       ! Disable
4                         newpriv,! This mask
4                         0,        ! Temporary
4                         0)        ! Don't return old mask
4         ) THEN BEGIN
4                 write_message (.status);! Tell the user what's wrong
4                 return;           ! And give up
4                 END
2     END;
2
```

```
2 !
2 !       If SET PROT/DEVICE, call SETDEV module to do the job
2 !
2 IF cli$present(%ASCID 'DEVICE')
2 THEN
3     BEGIN
3     set$device(1);
3     RETURN;
2     END;
2
2 !
2 !       Parse the SET PROTECTION= value if any
2 !
2 $init_dyndesc(prot_desc);                 ! Make a dynamic descriptor
2
2 IF cli$present(%ASCID 'OPTION.PROTECTION.SYSTEM')
2 THEN
3     BEGIN
3     setpro_mask = .setpro_mask OR %X'000F';
3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.SYSTEM',prot_desc)
3     THEN setpro_prot = parse_class(prot_desc);
2     END;
2 IF cli$present(%ASCID 'OPTION.PROTECTION.OWNER')
2 THEN
3     BEGIN
3     setpro_mask = .setpro_mask OR %X'00F0';
3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.OWNER',prot_desc)
3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^4;
2     END;
2 IF cli$present(%ASCID 'OPTION.PROTECTION.GROUP')
2 THEN
3     BEGIN
3     setpro_mask = .setpro_mask OR %X'0F00';
3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.GROUP',prot_desc)
3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^8;
2     END;
2 IF cli$present(%ASCID 'OPTION.PROTECTION.WORLD')
2 THEN
3     BEGIN
3     setpro_mask = .setpro_mask OR %X'F000';
3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.WORLD',prot_desc)
3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^12;
2     END;
2
2 !
2 ! Complement the protection value since at this point, a bit set true
2 ! indicates that we want to ALLOW access, while the system convention
2 ! is that a bit set true indicates that we want to DENY access.
2 !
2 IF .setpro_mask NEQ 0                     ! If any protections specified
2 THEN setpro_prot = NOT .setpro_prot;      ! then get the complement
2
2 !
2 ! Now save the command level protection in the global protection
2 ! area so it can be combined with individual file specifications
2 ! later.  If the user did not supply a command level protection then
2 ! the global_mask will have a value of zero.  If no local protection
```

```
2 ! options are specified then the final result_mask will also have a
2 ! value of zero.  This will cause the protection of the target file
2 ! to be set to the current process default.
2 !
2
2 global_mask = .setpro_mask;
2 global_prot = .setpro_prot;
2
2 !
2 !       Parse command qualifiers
2 !
2 qualifier$flags[qual_log] = cli$present(%ASCID 'LOG');
2 qualifier$flags[qual_confirm] = cli$present(%ASCID 'CONFIRM');
2
2 !
2 ! Loop thru all output files specified
2 !
2 WHILE get_output_file () DO                    ! For each output file
3     BEGIN
3         lib$file_scan (                        ! Search for file names
3                 output_fab,                    !  -using this file name
3                 setpro_action,                 !  -call here on match
3                 search_error)                  !  -call here on error
3
2     END;
2
2 RETURN;
2
1 END;


                                                                  .TITLE  SETPRO
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                          00 00 45 43 49 56 45 44  00000 P.AAB:   .ASCII  \DEVICE\<0><0>
                                         010E0006  00008 P.AAA:   .LONG   17694726
                                        00000000'  0000C          .ADDRESS P.AAB
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00010 P.AAD:   .ASCII  \OPTION.PROTECTION.SYSTEM\
                  4D 45 54 53 59 53 2E 4E 4F  0001F
                                         010E0018  00028 P.AAC:   .LONG   17694744
                                        00000000'  0002C          .ADDRESS P.AAD
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00030 P.AAF:   .ASCII  \OPTION.PROTECTION.SYSTEM\
                  4D 45 54 53 59 53 2E 4E 4F  0003F
                                         010E0018  00048 P.AAE:   .LONG   17694744
                                        00000000'  0004C          .ADDRESS P.AAF
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00050 P.AAH:   .ASCII  \OPTION.PROTECTION.OWNER\<0>
                  00 52 45 4E 57 4F 2E 4E 4F  0005F
                                         010E0017  00068 P.AAG:   .LONG   17694743
                                        00000000'  0006C          .ADDRESS P.AAH
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00070 P.AAJ:   .ASCII  \OPTION.PROTECTION.OWNER\<0>
                  00 52 45 4E 57 4F 2E 4E 4F  0007F
                                         010E0017  00088 P.AAI:   .LONG   17694743
                                        00000000'  0008C          .ADDRESS P.AAJ
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00090 P.AAL:   .ASCII  \OPTION.PROTECTION.GROUP\<0>
                  00 50 55 4F 52 47 2E 4E 4F  0009F
```

```
                                           010E0017  000A8 P.AAK:   .LONG    17694743                          ;
                                          00000000'  000AC          .ADDRESS P.AAL                             ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  000B0 P.AAN:   .ASCII   \OPTION.PROTECTION.GROUP\<0>   ;
                     00 50 55 4F 52 47 2E 4E 4F  000BF                                                     ;
                                           010E0017  000C8 P.AAM:   .LONG    17694743                          ;
                                          00000000'  000CC          .ADDRESS P.AAN                             ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  000D0 P.AAP:   .ASCII   \OPTION.PROTECTION.WORLD\<0>   ;
                     00 44 4C 52 4F 57 2E 4E 4F  000DF                                                     ;
                                           010E0017  000E8 P.AAO:   .LONG    17694743                          ;
                                          00000000'  000EC          .ADDRESS P.AAP                             ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  000F0 P.AAR:   .ASCII   \OPTION.PROTECTION.WORLD\<0>   ;
                     00 44 4C 52 4F 57 2E 4E 4F  000FF                                                     ;
                                           010E0017  00108 P.AAQ:   .LONG    17694743                          ;
                                          00000000'  0010C          .ADDRESS P.AAR                             ;
                                        00 47 4F 4C  00110 P.AAT:   .ASCII   \LOG\<0>                          ;
                                           010E0003  00114 P.AAS:   .LONG    17694723                          ;
                                          00000000'  00118          .ADDRESS P.AAT                             ;
                         00 4D 52 49 46 4E 4F 43  0011C P.AAV:   .ASCII   \CONFIRM\<0>                      ;
                                           010E0007  00124 P.AAU:   .LONG    17694727                          ;
                                          00000000'  00128          .ADDRESS P.AAV                             ;

                                                                    .PSECT   $OWN$,NOEXE,2

                                           00000000  00000 SETPRO$FLAGS:
                                                                    .LONG    0                                 ;
                                           00000000  00004 QUALIFIER$FLAGS:
                                                                    .LONG    0                                 ;
                                               0000  00008 SETPRO_PROT:
                                                                    .WORD    0                                 ;
                                               0000  0000A SETPRO_MASK:
                                                                    .WORD    0                                 ;
                                                     0000C GLOBAL_MASK:
                                                                    .BLKB    2
                                                     0000E GLOBAL_PROT:
                                                                    .BLKB    2
                                                     00010 RESULT_MASK:
                                                                    .BLKB    2
                                                     00012 RESULT_PROT:
                                                                    .BLKB    2
                                                     00014 OLDPRIV:.BLKB     8
                                                00#  0001C NEWPRIV:.BYTE     0[3]                              ;
                                                 10  0001F          .BYTE    16                                ;
                                                     00020          .BLKB    4
                                           00000001  00024 WORST_ERROR:
                                                                    .LONG    1                                 ;
                                                     00028 OUTPUT_FILENAME:
                                                                    .BLKB    255
                                                     00127          .BLKB    1
                                                     00128 OUTPUT_NAM_RESULT:
                                                                    .BLKB    255
                                                     00227          .BLKB    1
                                                     00228 OUTPUT_NAM_EXPANDED:
                                                                    .BLKB    255
                                                     00327          .BLKB    1
                                                 02  00328 OUTPUT_RLF:
                                                                    .BYTE    2                                 ;
                                                 60  00329          .BYTE    96                                ;
```

```
         00  0032A           .BYTE   0                                        ;
         00  0032B           .BYTE   0                                        ;
   0000000C  0032C           .LONG   0                                        ;
         00  00330           .BYTE   0                                        ;
         00  00331           .BYTE   0                                        ;
         00  00332           .BYTE   0                                        ;
         00  00333           .BYTE   0                                        ;
   00000000  00334           .LONG   0                                        ;
   00000000  00338           .LONG   0                                        ;
      0000#  0033C           .WORD   0[8]                                     ;
      0000#  0034C           .WORD   0[3]                                     ;
      0000#  00352           .WORD   0[3]                                     ;
   00000000  00358           .LONG   0                                        ;
   00000000  0035C           .LONG   0                                        ;
         00  00360           .BYTE   0                                        ;
         00  00361           .BYTE   0                                        ;
         00  00362           .BYTE   0                                        ;
         00  00363           .BYTE   0                                        ;
         00  00364           .BYTE   0                                        ;
         00  00365           .BYTE   0                                        ;
        00#  00366           .BYTE   0[2]                                     ;
   00000000  00368           .LONG   0                                        ;
   00000000  0036C           .LONG   0                                        ;
   00000000  00370           .LONG   0                                        ;
   00000000  00374           .LONG   0                                        ;
   00000000  00378           .LONG   0                                        ;
   00000000  0037C           .LONG   0                                        ;
  00000000#  00380           .LONG   0[2]                                     ;
         02  00388 OUTPUT_NAM:
                             .BYTE   2                                        ;
         60  00389           .BYTE   96                                       ;
         FF  0038A           .BYTE   -1                                       ;
         00  0038B           .BYTE   0                                        ;
  00000000'  0038C           .ADDRESS OUTPUT_NAM_RESULT                       ;
         00  00390           .BYTE   0                                        ;
         00  00391           .BYTE   0                                        ;
         FF  00392           .BYTE   -1                                       ;
         00  00393           .BYTE   0                                        ;
  00000000'  00394           .ADDRESS OUTPUT_NAM_EXPANDED                     ;
  00000000'  00398           .ADDRESS OUTPUT_RLF                              ;
      0000#  0039C           .WORD   0[8]                                     ;
      0000#  003AC           .WORD   0[3]                                     ;
      0000#  003B2           .WORD   0[3]                                     ;
   00000000  003B8           .LONG   0                                        ;
   00000000  003BC           .LONG   0                                        ;
         00  003C0           .BYTE   0                                        ;
         00  003C1           .BYTE   0                                        ;
         00  003C2           .BYTE   0                                        ;
         00  003C3           .BYTE   0                                        ;
         00  003C4           .BYTE   0                                        ;
         00  003C5           .BYTE   0                                        ;
        00#  003C6           .BYTE   0[2]                                     ;
   00000000  003C8           .LONG   0                                        ;
   00000000  003CC           .LONG   0                                        ;
   00000000  003D0           .LONG   0                                        ;
   00000000  003D4           .LONG   0                                        ;
   00000000  003D8           .LONG   0                                        ;
```

```
                        00000000  003DC           .LONG   0                                    ;
                        00000000# 003E0           .LONG   0[2]                                 ;
                              03  003E8 OUTPUT_FAB:
                                                  .BYTE   3                                    ;
                              50  003E9           .BYTE   80                                   ;
                            0000  003EA           .WORD   0                                    ;
                        00000000  003EC           .LONG   0                                    ;
                        00000000  003F0           .LONG   0                                    ;
                        00000000  003F4           .LONG   0                                    ;
                        00000000  003F8           .LONG   0                                    ;
                            0000  003FC           .WORD   0                                    ;
                              02  003FE           .BYTE   2                                    ;
                              00  003FF           .BYTE   0                                    ;
                        00000000  00400           .LONG   0                                    ;
                              00  00404           .BYTE   0                                    ;
                              00  00405           .BYTE   0                                    ;
                              00  00406           .BYTE   0                                    ;
                              02  00407           .BYTE   2                                    ;
                        00000000  00408           .LONG   0                                    ;
                        00000000  0040C           .LONG   0                                    ;
                        00000000' 00410           .ADDRESS OUTPUT_NAM                          ;
                        00000000  00414           .LONG   0                                    ;
                        00000000  00418           .LONG   0                                    ;
                              00  0041C           .BYTE   0                                    ;
                              00  0041D           .BYTE   0                                    ;
                            0000  0041E           .WORD   0                                    ;
                        00000000  00420           .LONG   0                                    ;
                            0000  00424           .WORD   0                                    ;
                              00  00426           .BYTE   0                                    ;
                              00  00427           .BYTE   0                                    ;
                        00000000  00428           .LONG   0                                    ;
                        00000000  0042C           .LONG   0                                    ;
                            0000  00430           .WORD   0                                    ;
                              00  00432           .BYTE   0                                    ;
                              00  00433           .BYTE   0                                    ;
                        00000000  00434           .LONG   0                                    ;

                                                  .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                                                  .EXTRN  SYS$FAO, SYS$SETPRV
                                                  .EXTRN  LIB$GET_COMMAND
                                                  .EXTRN  LIB$FILE_SCAN, LIB$SET_FILE_PROT
                                                  .EXTRN  SET$DEVICE, SET$_PRONOTCHG

                                                  .PSECT  $CODE$,NOWRT,2

                            00FC  00000           .ENTRY  SET$PROTECTION, Save R2,R3,R4,R5,R6,R7   ; 0248
57        0000V   CF    9E  00002                 MOVAB   PARSE_CLASS, R7                      ;
56    00000000G   00    9E  00007                 MOVAB   SYS$SETPRV, R6                       ;
55    00000000G   00    9E  0000E                 MOVAB   CLI$GET_VALUE, R5                    ;
54    00000000G   00    9E  00015                 MOVAB   CLI$PRESENT, R4                      ;
53        0000'   CF    9E  0001C                 MOVAB   SETPRO_PROT, R3                      ;
5E                08    C2  00021                 SUBL2   #8, SP                               ;
             0C   A3    9F  00024                 PUSHAB  OLDPRIV                              ; 0297
                  01    DD  00027                 PUSHL   #1                                   ;
7E                01    7D  00029                 MOVQ    #1, -(SP)                            ;
66                04    FB  0002C                 CALLS   #4, SYS$SETPRV                       ;
52                50    D0  0002F                 MOVL    R0, STATUS                           ;
```

```
                        15           52  E9 00032         BLBC    STATUS, 1$                    ;
         1A      0F     A3           04  E0 00035         BBS     #4, OLDPRIV+3, 2$             ; 0310
                                     7E  7C 0003A         CLRQ    -(SP)                         ; 0313
                              14     A3  9F 0003C         PUSHAB  NEWPRIV                       ;
                                     7E  D4 0003F         CLRL    -(SP)                         ;
                        66           04  FB 00041         CALLS   #4, SYS$SETPRV                ;
                        52           50  D0 00044         MOVL    R0, STATUS                    ;
                        0A           52  E8 00047         BLBS    STATUS, 2$                    ;
                                     52  DD 0004A 1$:     PUSHL   STATUS                        ; 0318
              00000000G 00           01  FB 0004C         CALLS   #1, LIB$SIGNAL                ;
                                         04 00053         RET                                   ; 0317
                           0000'     CF  9F 00054 2$:     PUSHAB  P.AAA                         ; 0326
                        64           01  FB 00058         CALLS   #1, CLI$PRESENT               ;
                        0A           50  E9 0005B         BLBC    R0, 3$                        ;
                                     01  DD 0005E         PUSHL   #1                            ; 0329
              00000000G 00           01  FB 00060         CALLS   #1, SET$DEVICE                ;
                                         04 00067         RET                                   ; 0328
                     6E 020E0000     8F  D0 00068 3$:     MOVL    #34471936, PROT_DESC          ; 0336
                              04     AE  D4 0006F         CLRL    PROT_DESC+4                   ;
                           0000'     CF  9F 00072         PUSHAB  P.AAC                         ; 0338
                        64           01  FB 00076         CALLS   #1, CLI$PRESENT               ;
                        18           50  E9 00079         BLBC    R0, 4$                        ;
                 02     A3           0F  88 0007C         BISB2   #15, SETPRO_MASK              ; 0341
                                     5E  DD 00080         PUSHL   SP                            ; 0342
                           0000'     CF  9F 00082         PUSHAB  P.AAE                         ;
                        65           02  FB 00086         CALLS   #2, CLI$GET_VALUE             ;
                        08           50  E9 00089         BLBC    R0, 4$                        ;
                                     5E  DD 0008C         PUSHL   SP                            ; 0343
                        67           01  FB 0008E         CALLS   #1, PARSE_CLASS               ;
                        63           50  B0 00091         MOVW    R0, SETPRO_PROT               ;
                           0000'     CF  9F 00094 4$:     PUSHAB  P.AAG                         ; 0345
                        64           01  FB 00098         CALLS   #1, CLI$PRESENT               ;
                        1C           50  E9 0009B         BLBC    R0, 5$                        ;
                 02     A3    F0     8F  88 0009E         BISB2   #240, SETPRO_MASK             ; 0348
                                     5E  DD 000A3         PUSHL   SP                            ; 0349
                           0000'     CF  9F 000A5         PUSHAB  P.AAI                         ;
                        65           02  FB 000A9         CALLS   #2, CLI$GET_VALUE             ;
                        0B           50  E9 000AC         BLBC    R0, 5$                        ;
                                     5E  DD 000AF         PUSHL   SP                            ; 0350
                        67           01  FB 000B1         CALLS   #1, PARSE_CLASS               ;
                        50           10  C4 000B4         MULL2   #16, R0                       ;
                        63           50  A8 000B7         BISW2   R0, SETPRO_PROT               ;
                           0000'     CF  9F 000BA 5$:     PUSHAB  P.AAK                         ; 0352
                        64           01  FB 000BE         CALLS   #1, CLI$PRESENT               ;
                        1C           50  E9 000C1         BLBC    R0, 6$                        ;
                 03     A3           0F  88 000C4         BISB2   #15, SETPRO_MASK+1            ; 0355
                                     5E  DD 000C8         PUSHL   SP                            ; 0356
                           0000'     CF  9F 000CA         PUSHAB  P.AAM                         ;
                        65           02  FB 000CE         CALLS   #2, CLI$GET_VALUE             ;
                        0C           50  E9 000D1         BLBC    R0, 6$                        ;
                                     5E  DD 000D4         PUSHL   SP                            ; 0357
                        67           01  FB 000D6         CALLS   #1, PARSE_CLASS               ;
         50             50           08  78 000D9         ASHL    #8, R0, R0                    ;
                        63           50  A8 000DD         BISW2   R0, SETPRO_PROT               ;
                           0000'     CF  9F 000E0 6$:     PUSHAB  P.AAO                         ; 0359
                        64           01  FB 000E4         CALLS   #1, CLI$PRESENT               ;
                        1D           50  E9 000E7         BLBC    R0, 7$                        ;
```

```
                 03  A3      F0  8F  88 000EA       BISB2   #240, SETPRO_MASK+1                    ; 0362
                                 5E  DD 000EF       PUSHL   SP                                     ; 0363
                       0000'     CF  9F 000F1       PUSHAB  P.AAQ                                  ;
                     65          02  FB 000F5       CALLS   #2, CLI$GET_VALUE                      ;
                     0C          50  E9 000FB       BLBC    R0, 7$                                 ;
                                 5E  DD 000FB       PUSHL   SP                                     ; 0364
                     67          01  FB 000FD       CALLS   #1, PARSE_CLASS                        ;
            50       50          0C  78 00100       ASHL    #12, R0, R0                            ;
                     63          50  A8 00104       BISW2   R0, SETPRO_PROT                        ;
                     50      02  A3  3C 00107 7$:   MOVZWL  SETPRO_MASK, R0                        ; 0372
                                 03  13 0010B       BEQL    8$                                     ;
                     63          63  B2 0010D       MCOMW   SETPRO_PROT, SETPRO_PROT               ; 0373
                 04  A3          50  B0 00110 8$:   MOVW    R0, GLOBAL_MASK                        ; 0385
                 06  A3          63  B0 00114       MOVW    SETPRO_PROT, GLOBAL_PROT               ; 0386
                       0000'     CF  9F 00118       PUSHAB  P.AAS                                  ; 0391
                     64          01  FB 0011C       CALLS   #1, CLI$PRESENT                        ;
   FC    A3        01  01        50  F0 0011F       INSV    R0, #1, #1, QUALIFIER$FLAGS            ;
                       0000'     CF  9F 00125       PUSHAB  P.AAU                                  ; 0392
                     64          01  FB 00129       CALLS   #1, CLI$PRESENT                        ;
   FC    A3        01  02        50  F0 0012C       INSV    R0, #2, #1, QUALIFIER$FLAGS            ;
                0000V  CF        00  FB 00132 9$:   CALLS   #0, GET_OUTPUT_FILE                    ; 0397
                     15          50  E9 00137       BLBC    R0, 10$                                ;
                       0000V     CF  9F 0013A       PUSHAB  SEARCH_ERROR                           ; 0399
                       0000V     CF  9F 0013E       PUSHAB  SETPRO_ACTION                          ;
                       03E0      C3  9F 00142       PUSHAB  OUTPUT_FAB                             ;
            00000000G  00        03  FB 00146       CALLS   #3, LIB$FILE_SCAN                      ;
                                 E3  11 0014D       BRB     9$                                     ; 0398
                                     04 0014F 10$:  RET                                            ; 0408

; Routine Size:  336 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 ROUTINE setpro_action (fab): =
1
1 !----
1 !
1 ! Functional description
1 !
1 !       This routine is called from lib$file_scan whenever
1 !       a file name match occurs
1 !
1 ! Input parameters
1 !
1 !       fab = Address of block describing the file
1 !       fab$l_nam = pointer to name block
1 !
1 ! Output parameters
1 !
1 !       First error encountered, or TRUE is RETURNed
1 !
1 !----
1
2 BEGIN
2
2 MAP fab: REF BBLOCK;                      ! Define fab block format
2 BIND nam = .fab[fab$l_nam]: BBLOCK;       ! Define name block
2
2 LOCAL
2     p_res_mask,                           ! Enable-mask parameter
2     p_res_prot,                           ! Value-mask parameter
2     final_prot: WORD,                     ! Recieves final protection
2     desc:       VECTOR[2],                ! Temporary sting descriptor
2     status;                               ! Recieves status
2
2 !
2 ! If /CONFIRM was set by the user then interrogate him to see if
2 ! this file is to be unlocked
2 !
2
3 IF (.qualifier$flags[qual_confirm])       ! If confirmation requested
2 THEN
3       IF NOT (confirm_action (             ! Call confirmation rout. with
3               .fab))                       !  -address of fab
2          THEN return(false);               ! If not confirmed then exit
2
2 !
2 ! Now load the local descriptor with the file name size and address
2 ! from the nam block
2 !
2
2 desc[0] = .nam[nam$b_rsl];                ! Load file name size
2 desc[1] = .nam[nam$l_rsa];                ! Load file name address
2
2 !
2 ! But if this is a non-wildcarded network specification, then filescan
2 ! doesn't do the $SEARCH, so we use the expanded file specification instead.
2 !
2
2 IF .nam [nam$b_rsl] EQL 0
```

```
  3 THEN BEGIN
  3         desc[0] = .nam[nam$b_esl];       ! Load file name size
  3         desc[1] = .nam[nam$l_esa];       ! Load file name address
  2      END;
  2
  2
  2 !
  2 ! Compute the parameters for lib$set_file_prot.  If no /PROTECTION value was
  2 ! specified, call with null parameters for enable-mask and value-mask to cause
  2 ! protection to be set to the process default.
  2 !
  2
  2 p_res_mask = result_mask;                ! Assume not default
  2 p_res_prot = result_prot;
  2 IF .result_mask EQL 0                    ! If no protection values specified,
  2     THEN p_res_mask = p_res_prot = 0;    !      pass null parameters
  2
  2 !
  2 ! Call lib$set_file_prot to set file protection
  2 !
  2
  3 IF NOT (status = lib$set_file_prot (     ! Call library routine with
  3                         desc,            !  - file name
  3                         .p_res_mask,     !  - result mask
  3                         .p_res_prot,     !  - result protection
  3                         final_prot))     !  - final protection returned
  3                                          !    by lib$set_file_prot
  2     THEN
  3         BEGIN
P 3         write_message (                  ! Tell the user of error
P 3                         set$_pronotchg,! - "Not changed" error message
P 3                         1,               !  - 1 FAO arguement
P 3                         desc,            !  - descriptor of filename
  3                         .status);        !  - original error
  3         return (.status);                ! Return to the caller
  3         END;
  2
  2
  2 !
  2 ! If /LOG was set then do it
  2 !
  2
  3 IF (.qualifier$flags[qual_log])          ! If logging requested
  2 THEN log_results (.fab,.final_prot);     !  then call the logger
  2
  2 RETURN (true);
  2
  1 END;
```

```
                                   001C 00000 SETPRO_ACTION:
                                                     .WORD   Save R2,R3,R4                   ; 0409
              54      0000'   CF   9E 00002          MOVAB   QUALIFIER$FLAGS, R4             ;
              5E              0C   C2 00007          SUBL2   #12, SP                         ;
```

```
                     53       04 AC D0 0000A        MOVL    FAB, R3                                ; 0432
                     52       28 A3 D0 0000E        MOVL    40(R3), R2                             ;
         0A          64          02 E1 00012        BBC     #2, QUALIFIER$FLAGS, 1$                ; 0446
                                 53 DD 00016        PUSHL   R3                                     ; 0449
              0000V  CF          01 FB 00018        CALLS   #1, CONFIRM_ACTION                     ;
                     68          50 E9 0001D        BLBC    R0, 6$                                 ;
                04   AE       03 A2 9A 00020 1$:    MOVZBL  3(R2), DESC                            ; 0457
                08   AE       04 A2 D0 00025        MOVL    4(R2), DESC+4                          ; 0458
                              03 A2 95 0002A        TSTB    3(R2)                                  ; 0465
                                 0A 12 0002D        BNEQ    2$                                     ;
                04   AE       0B A2 9A 0002F        MOVZBL  11(R2), DESC                           ; 0467
                08   AE       0C A2 D0 00034        MOVL    12(R2), DESC+4                         ; 0468
                     51       0C A4 9E 00039 2$:    MOVAB   RESULT_MASK, P_RES_MASK                ; 0478
                     50       0E A4 9E 0003D        MOVAB   RESULT_PROT, P_RES_PROT                ; 0479
                              0C A4 B5 00041        TSTW    RESULT_MASK                            ; 0480
                                 02 12 00044        BNEQ    3$                                     ;
                                 50 7C 00046        CLRQ    P_RES_PROT                             ; 0481
                            4001 8F BB 00048 3$:    PUSHR   #^M<R0,SP>                             ; 0490
                                 51 DD 0004C        PUSHL   P_RES_MASK                             ; 0489
                              10 AE 9F 0004E        PUSHAB  DESC                                   ; 0487
          00000000G  00          04 FB 00051        CALLS   #4, LIB$SET_FILE_PROT                  ;
                     52          50 D0 00058        MOVL    R0, STATUS                             ;
                     18          52 E8 0005B        BLBS    STATUS, 4$                             ;
                                 52 DD 0005E        PUSHL   STATUS                                 ; 0499
                              08 AE 9F 00060        PUSHAB  DESC                                   ;
                                 01 DD 00063        PUSHL   #1                                     ;
                       00000000G 8F DD 00065        PUSHL   #SET$_PRONOTCHG                        ;
          00000000G  00          04 FB 0006B        CALLS   #4, LIB$SIGNAL                         ;
                     50          52 D0 00072        MOVL    STATUS, R0                             ; 0500
                                    04 00075        RET                                            ;
         0A          64          01 E1 00076 4$:    BBC     #1, QUALIFIER$FLAGS, 5$                ; 0508
                     7E          6E 3C 0007A        MOVZWL  FINAL_PROT, -(SP)                      ; 0509
                                 53 DD 0007D        PUSHL   R3                                     ;
              0000V  CF          02 FB 0007F        CALLS   #2, LOG_RESULTS                        ;
                     50          01 D0 00084 5$:    MOVL    #1, R0                                 ; 0511
                                    04 00087        RET                                            ;
                                 50 D4 00088 6$:    CLRL    R0                                     ; 0513
                                    04 0008A        RET                                            ;

; Routine Size:  139 bytes,    Routine Base:  $CODE$ + 0150
```

```
1 ROUTINE get_output_file =
1
1 !----
1 ! Functional Description
1 !
1 !       Obtain the next file specification from the output file
1 !       list and perform the initial parsing to ensure legality.
1 !
1 ! Output Parameters
1 !
1 !       RETURNs TRUE if a file name was obtained
1 !       RETURNS FALSE if the command line is exhausted
1 !
1 !       output_fab = FAB initialized for current file name
1 !       output_nam = NAM block associated with FAB
1 !
1 !----
1
2 BEGIN
2
2 LOCAL
2     prot_desc : $BBLOCK[dsc$c_s_bln],    ! Descriptor to define protection
2     file_desc : $BBLOCK[dsc$c_s_bln];    ! Descriptor to define filename
2
2 CH$FILL(0, dsc$c_s_bln, file_desc);
2 file_desc[dsc$w_length] = nam$c_maxrss;
2 file_desc[dsc$a_pointer] = output_filename;
2
2 !
2 !       If the output file list is depleted, then exit.
2 !
2 IF NOT cli$get_value(%ASCID 'FILE',file_desc)
2 THEN RETURN false;
2
2 !
2 ! Otherwise, continue processing.  Put new filename into FAB.
2 !
2 output_fab[fab$l_fna] = .file_desc[dsc$a_pointer];
2 output_fab[fab$b_fns] = .file_desc[dsc$w_length];
2
2 !
2 ! Get protection values (if any) for this file
2 !
2 setpro_prot = 0;          ! Clear any left over values
2 setpro_mask = 0;          ! Clear any left over values
2
2 IF cli$present(%ASCID 'PROTECTION')
2 THEN
3     BEGIN
3
3     $init_dyndesc(prot_desc);   ! Make protection descriptor dynamic
3
3     IF cli$present(%ASCID 'PROTECTION.SYSTEM')
3     THEN
4         BEGIN
4         setpro_mask = .setpro_mask OR %X'000F';
4         IF cli$get_value(%ASCID 'PROTECTION.SYSTEM',prot_desc)
```

```
;  4              THEN setpro_prot = parse_class(prot_desc);
;  3              END;
;  3          IF cli$present(%ASCID 'PROTECTION.OWNER')
;  3          THEN
;  4              BEGIN
;  4              setpro_mask = .setpro_mask OR %X'00F0';
;  4              IF cli$get_value(%ASCID 'PROTECTION.OWNER',prot_desc)
;  4              THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^4;
;  3              END;
;  3          IF cli$present(%ASCID 'PROTECTION.GROUP')
;  3          THEN
;  4              BEGIN
;  4              setpro_mask = .setpro_mask OR %X'0F00';
;  4              IF cli$get_value(%ASCID 'PROTECTION.GROUP',prot_desc)
;  4              THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^8;
;  3              END;
;  3          IF cli$present(%ASCID 'PROTECTION.WORLD')
;  3          THEN
;  4              BEGIN
;  4              setpro_mask = .setpro_mask OR %X'F000';
;  4              IF cli$get_value(%ASCID 'PROTECTION.WORLD',prot_desc)
;  4              THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^12;
;  3              END;
;  3
;  3      !
;  3      ! Complement the protection value since at this point, a bit set true
;  3      ! indicates that we want to ALLOW access, while the system convention
;  3      ! is that a bit set true indicates that we want to DENY access.
;  3      !
;  3          setpro_prot = NOT .setpro_prot;
;  3          END;
;  2
;  2      !
;  2      ! Build the combined protection values.  That is, apply the command
;  2      ! level protection first, and then any local protection values
;  2      ! supplied with this parameter
;  2      !
;  2      result_mask =   .global_mask OR .setpro_mask;               ! OR masks
;  3      result_prot =   (.global_prot AND (NOT .setpro_mask))       ! -global part
;  2                                  OR                              ! -OR'd with
;  2                      (.setpro_prot AND .setpro_mask);            ! -local part
;  2
;  2      RETURN true;
;  2
;  1      END;


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                45 4C 49 46  0012C P.AAX:     .ASCII  \FILE\                              ;
                                  010E0004   00130 P.AAW:     .LONG   17694724                            ;
                                  00000000'  00134            .ADDRESS P.AAX                              ;
             00 00 4E 4F 49 54 43 45 54 4F 52 50  00138 P.AAZ: .ASCII \PROTECTION\<0><0>                ;
                                  010E000A   00144 P.AAY:     .LONG   17694730                            ;
                                  00000000'  00148            .ADDRESS P.AAZ                              ;
54 53 59 53 2E 4E 4F 49 54 43 45 54 4F 52 50  0014C P.ABB:    .ASCII  \PROTECTION.SYSTEM\<0><0><0>        ;
```

```
                                      00  00  00  4D  45   0015B
                                           010E0011   00160 P.ABA:   .LONG    17694737                                 ;
                                          00000000'   00164          .ADDRESS P.ABB                                    ;
54  53  59  53  2E  4E  4F  49  54  43  45  54  4F  52  50   00168 P.ABD:   .ASCII   \PROTECTION.SYSTEM\<0><0><0>             ;
                                      00  00  00  4D  45   00177                                                    ;
                                           010E0011   0017C P.ABC:   .LONG    17694737                                 ;
                                          00000000'   00180          .ADDRESS P.ABD                                    ;
45  4E  57  4F  2E  4E  4F  49  54  43  45  54  4F  52  50   00184 P.ABF:   .ASCII   \PROTECTION.OWNER\                       ;
                                                  52   00193                                                    ;
                                           010E0010   00194 P.ABE:   .LONG    17694736                                 ;
                                          00000000'   00198          .ADDRESS P.ABF                                    ;
45  4E  57  4F  2E  4E  4F  49  54  43  45  54  4F  52  50   0019C P.ABH:   .ASCII   \PROTECTION.OWNER\                       ;
                                                  52   001AB                                                    ;
                                           010E0010   001AC P.ABG:   .LONG    17694736                                 ;
                                          00000000'   001B0          .ADDRESS P.ABH                                    ;
55  4F  52  47  2E  4E  4F  49  54  43  45  54  4F  52  50   001B4 P.ABJ:   .ASCII   \PROTECTION.GROUP\                       ;
                                                  50   001C3                                                    ;
                                           010E0010   001C4 P.ABI:   .LONG    17694736                                 ;
                                          00000000'   001C8          .ADDRESS P.ABJ                                    ;
55  4F  52  47  2E  4E  4F  49  54  43  45  54  4F  52  50   001CC P.ABL:   .ASCII   \PROTECTION.GROUP\                       ;
                                                  50   001DB                                                    ;
                                           010E0010   001DC P.ABK:   .LONG    17694736                                 ;
                                          00000000'   001E0          .ADDRESS P.ABL                                    ;
4C  52  4F  57  2E  4E  4F  49  54  43  45  54  4F  52  50   001E4 P.ABN:   .ASCII   \PROTECTION.WORLD\                       ;
                                                  44   001F3                                                    ;
                                           010E0010   001F4 P.ABM:   .LONG    17694736                                 ;
                                          00000000'   001F8          .ADDRESS P.ABN                                    ;
4C  52  4F  57  2E  4E  4F  49  54  43  45  54  4F  52  50   001FC P.ABP:   .ASCII   \PROTECTION.WORLD\                       ;
                                                  44   0020B                                                    ;
                                           010E0010   0020C P.ABO:   .LONG    17694736                                 ;
                                          00000000'   00210          .ADDRESS P.ABP                                    ;



                                                                     .PSECT   $CODE$,NOWRT,2

                                                07FC 00000 GET_OUTPUT_FILE:
                                                                     .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10      ; 0514
                            5A       0000V   CF   9E 00002           MOVAB    PARSE_CLASS, R10                      ;
                            59   00000000G   00   9E 00007           MOVAB    CLI$PRESENT, R9                       ;
                            58   00000000G   00   9E 0000E           MOVAB    CLI$GET_VALUE, R8                     ;
                            57       0000'   CF   9E 00015           MOVAB    P.AAW, R7                             ;
                            56       0000'   CF   9E 0001A           MOVAB    SETPRO_PROT, R6                       ;
                            5E               10   C2 0001F           SUBL2    #16, SP                               ;
          08          00    6E               00   2C 00022           MOVC5    #0, (SP), #0, #8, FILE_DESC           ; 0538
                                             6E      00027                                                    ;
                            6E          FF   8F   9B 00028           MOVZBW   #255, FILE_DESC                       ; 0539
                       04   AE          20   A6   9E 0002C           MOVAB    OUTPUT_FILENAME, FILE_DESC+4          ; 0540
                                      4080   8F   BB 00031           PUSHR    #^M<R7,SP>                            ; 0545
                            68               02   FB 00035           CALLS    #2, CLI$GET_VALUE                     ;
                            03               50   E8 00038           BLBS     R0, 1$                                ;
                                          00E8   31 0003B           BRW      8$                                    ;
                     040C   C6          04   AE   D0 0003E 1$:       MOVL     FILE_DESC+4, OUTPUT_FAB+44            ; 0551
                     0414   C6               6E   90 00044           MOVB     FILE_DESC, OUTPUT_FAB+52              ; 0552
                                             66   D4 00049           CLRL     SETPRO_PROT                           ; 0557
                                        14   A7   9F 0004B           PUSHAB   P.AAY                                 ; 0560
                            69               01   FB 0004E           CALLS    #1, CLI$PRESENT                       ;
```

```
                03          50  E8 00051       BLBS    R0, 2$                                    ;
                          00A7  31 00054       BRW     7$                                        ;
             08    AE 020E0000  8F D0 00057 2$: MOVL   #34471936, PROT_DESC                      ; 0564
                       0C       AE D4 0005F    CLRL    PROT_DESC+4                               ;
                       30       A7 9F 00062    PUSHAB  P.ABA                                     ; 0566
                69              01 FB 00065    CALLS   #1, CLI$PRESENT                           ;
                19              50 E9 00068    BLBC    R0, 3$                                    ;
             02    A6           0F 88 0006B    BISB2   #15, SETPRO_MASK                          ; 0569
                       08       AE 9F 0006F    PUSHAB  PROT_DESC                                 ; 0570
                       4C       A7 9F 00072    PUSHAB  P.ABC                                     ;
                68              02 FB 00075    CALLS   #2, CLI$GET_VALUE                         ;
                09              50 E9 00078    BLBC    R0, 3$                                    ;
                       08       AE 9F 0007B    PUSHAB  PROT_DESC                                 ; 0571
                6A              01 FB 0007E    CALLS   #1, PARSE_CLASS                           ;
                66              50 B0 00081    MOVW    R0, SETPRO_PROT                           ;
                       64       A7 9F 00084 3$: PUSHAB P.ABE                                     ; 0573
                69              01 FB 00087    CALLS   #1, CLI$PRESENT                           ;
                1D              50 E9 0008A    BLBC    R0, 4$                                    ;
             02    A6     F0    8F 88 0008D    BISB2   #240, SETPRO_MASK                         ; 0576
                       08       AE 9F 00092    PUSHAB  PROT_DESC                                 ; 0577
                       7C       A7 9F 00095    PUSHAB  P.ABG                                     ;
                68              02 FB 00098    CALLS   #2, CLI$GET_VALUE                         ;
                0C              50 E9 0009B    BLBC    R0, 4$                                    ;
                       08       AE 9F 0009E    PUSHAB  PROT_DESC                                 ; 0578
                6A              01 FB 000A1    CALLS   #1, PARSE_CLASS                           ;
                50              10 C4 000A4    MULL2   #16, R0                                   ;
                66              50 A8 000A7    BISW2   R0, SETPRO_PROT                           ;
                     0094       C7 9F 000AA 4$: PUSHAB P.ABI                                     ; 0580
                69              01 FB 000AE    CALLS   #1, CLI$PRESENT                           ;
                1E              50 E9 000B1    BLBC    R0, 5$                                    ;
             03    A6           0F 88 000B4    BISB2   #15, SETPRO_MASK+1                        ; 0583
                       08       AE 9F 000B8    PUSHAB  PROT_DESC                                 ; 0584
                     00AC       C7 9F 000BB    PUSHAB  P.ABK                                     ;
                68              02 FB 000BF    CALLS   #2, CLI$GET_VALUE                         ;
                0D              50 E9 000C2    BLBC    R0, 5$                                    ;
                       08       AE 9F 000C5    PUSHAB  PROT_DESC                                 ; 0585
                6A              01 FB 000C8    CALLS   #1, PARSE_CLASS                           ;
          50    50              08 78 000CB    ASHL    #8, R0, R0                                ;
                66              50 A8 000CF    BISW2   R0, SETPRO_PROT                           ;
                     00C4       C7 9F 000D2 5$: PUSHAB P.ABM                                     ; 0587
                69              01 FB 000D6    CALLS   #1, CLI$PRESENT                           ;
                1F              50 E9 000D9    BLBC    R0, 6$                                    ;
             03    A6     F0    8F 88 000DC    BISB2   #240, SETPRO_MASK+1                       ; 0590
                       08       AE 9F 000E1    PUSHAB  PROT_DESC                                 ; 0591
                     00DC       C7 9F 000E4    PUSHAB  P.ABO                                     ;
                68              02 FB 000E8    CALLS   #2, CLI$GET_VALUE                         ;
                0D              50 E9 000EB    BLBC    R0, 6$                                    ;
                       08       AE 9F 000EE    PUSHAB  PROT_DESC                                 ; 0592
                6A              01 FB 000F1    CALLS   #1, PARSE_CLASS                           ;
          50    50              0C 78 000F4    ASHL    #12, R0, R0                               ;
                66              50 A8 000F8    BISW2   R0, SETPRO_PROT                           ;
                66              66 B2 000FB 6$: MCOMW  SETPRO_PROT, SETPRO_PROT                  ; 0600
    08    A6    04 A6     02    A6 A9 000FE 7$: BISW3  SETPRO_MASK, GLOBAL_MASK, RESULT_MASK     ; 0608
                51        06    A6 3C 00105    MOVZWL  GLOBAL_PROT, R1                           ; 0609
                50        02    A6 3C 00109    MOVZWL  SETPRO_MASK, R0                           ;
                51              50 CA 0010D    BICL2   R0, R1                                    ;
                50              66 3C 00110    MOVZWL  SETPRO_PROT, R0                           ; 0611
```

```
                       52     02  A6  3C 00113      MOVZWL  SETPRO_MASK, R2              ;
                       52         52  D2 00117      MCOML   R2, R2                       ;
                       50         52  CA 0011A      BICL2   R2, R0                       ;
          0A    A6     50         51  A9 0011D      BISW3   R1, R0, RESULT_PROT          ;
                       50         01  D0 00122      MOVL    #1, R0                       ; 0613
                                  04  00125         RET                                  ;
                                  50  D4 00126 8$:  CLRL    R0                           ; 0615
                                  04  00128         RET                                  ;

; Routine Size:  297 bytes,    Routine Base:  $CODE$ + 01DB
```

```
1 ROUTINE confirm_action (fab) =
1
1 !----
1 !
1 ! Functional description
1 !
1 !       This routine is called from the main loop whenever
1 !       confirmation is requested.
1 !
1 ! Input parameters
1 !
1 !       fab = Address of block describing the file
1 !       fab$l_nam = pointer to name block
1 !
1 ! Output parameters
1 !
1 !               TRUE    --> Action should be taken
1 !               FALSE   --> Action should be cancelled
1 !
1 !
1 !----
1
2 BEGIN
2
2 MAP fab: REF BBLOCK;                          ! Define fab block format
2 BIND nam = .fab[fab$l_nam]: BBLOCK;           ! Define name block
2
2 OWN
2     file_desc:  BBLOCK[dsc$c_s_bln],          ! file_descriptor for file name
2     fao_desc:   BBLOCK[dsc$c_s_bln],          ! FAO work area descriptor
2     reply_desc: BBLOCK[dsc$c_s_bln]           ! Buffer desc. for reply
2                 INITIAL (
2                 WORD (0),                     !  -size = 0
2                 BYTE (dsc$k_dtype_t),         !  -arguement type = ascii
2                 BYTE (dsc$k_class_d),         !  -class = dynamic
2                 LONG (0));                    !  -pointer = 0
2 LOCAL
2     status:     BLOCK[1],                     ! Recieves status
2     length :    WORD,                         ! Length of resultant message
2     char,                                     ! Character work area
2     fao_buffer: VECTOR[512,BYTE];             ! FAO work area
2
2
2 !
2 ! Initialize descriptors with:
2 !       1) FAO buffer   -->     fao_desc
2 !       2) file name    -->     file_desc
2 !
2
2 fao_desc[dsc$w_length] = 512;                         ! FAO work area size
2 fao_desc[dsc$a_pointer] = fao_buffer;                 ! FAO work area address
2
2 file_desc[dsc$w_length] = .nam[nam$b_rsl];           ! Resultant file name length
2 file_desc[dsc$a_pointer] = .nam[nam$l_rsa];          ! Resultant file name address
2
2 IF .nam [nam$b_rsl] EQL 0                            ! If resultant name is blank
3 THEN BEGIN                                           !    then use expanded name
```

```
        file_desc[dsc$w_length] = .nam[nam$b_esl];  ! - expanded file name length
        file_desc[dsc$a_pointer] = .nam[nam$l_esa]; ! - expanded file name address
        END;

    !
    ! Now call SYS$FAO to expand message
    !

    IF NOT (status = SYS$FAO (                   ! Call system service with
                                                 !  -messages desc. address
                  addrdesc('!AS, change protection? (Y or N): '),
                  length,                        !  -place to put result length
                  fao_desc,                      !  -descriptor of a work area
                  file_desc))                    !  -one FAO arguement
        THEN
            BEGIN
            write_message (.status);             ! Tell the user
            return (.status);                    ! Return to the caller
            END;

    !
    ! Now question user by calling LIB$GET_COMMAND using the result
    ! of the FAO call as a prompt string
    !

    fao_desc[dsc$w_length] = .length;            ! Move in prompt string size

    IF NOT (status = LIB$GET_COMMAND (           ! Call library routine with
                  reply_desc,                    !  -reply buffer desc. addr.
                  fao_desc))                     !  -prompt string desc. addr.
        THEN
            BEGIN
            IF (.status EQL rms$_eof)            ! If this was end of file then
            THEN status[sts$v_severity] =        !  make it a severe (i.e. non-
                  sts$k_severe;                  !  returnable) error
            write_message (.status);             ! Tell the user
            return (.status);                    ! Return to the caller
            END;


    !
    ! Now retrieve one charater from the buffer and set the return
    ! status depending on its value
    !

    IF (.reply_desc[dsc$w_length] EQL 0)         ! Did user hit <CR>?
    THEN return (false);                         ! Yes, return false

    char = CH$RCHAR (CH$PTR (.reply_desc[dsc$a_pointer]));

    IF (.char EQL 'Y' OR                         ! If user responded 'Y'
        .char EQL 'y' OR                         !     or 'y'
        .char EQL 'T' OR                         !     or 'T' (TRUE)
        .char EQL 't')                           !     or 't'
    THEN return (true)                           ! Return 'TAKE ACTION'
    ELSE return (false);                         ! Return 'CANCEL ACTION'
```

```
;  742          0730  2
;  743          0731  1 END;


                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

6F 72 70 20 65 67 6E 61 68 63 20 2C 53 41 21  00214 P.ABR:  .ASCII  \!AS, change protection? (Y or N): \<0>      ;
20 72 6F 20 59 28 20 3F 6E 6F 69 74 63 65 74  00223                                                            ;
                                 00 20 3A 29 4E  00232                                                         ;
                                             00  00237          .ASCII  <0>                                    ;
                                       00000022  00238 P.ABQ:   .LONG   34                                     ;
                                       00000000' 0023C          .ADDRESS P.ABR                                 ;

                                                          .PSECT  $OWN$,NOEXE,2

                                                 00438 FILE_DESC:
                                                                .BLKB   8
                                                 00440 FAO_DESC:
                                                                .BLKB   8
                                           0000  00448 REPLY_DESC:
                                                                .WORD   0
                                             0E  0044A          .BYTE   14                                     ;
                                             02  0044B          .BYTE   2                                      ;
                                       00000000  0044C          .LONG   0                                      ;


                                                          .PSECT  $CODE$,NOWRT,2

                                           000C  00000 CONFIRM_ACTION:
                                                                .WORD   Save R2,R3                             ; 0616
                 53        0000'  CF  9E  00002                 MOVAB   FAO_DESC, R3                           ;
                 5E        FDFC   CE  9E  00007                 MOVAB   -516(SP), SP                           ;
                 50          04   AC  D0  0000C                 MOVL    FAB, R0                                ; 0641
                 50          28   A0  D0  00010                 MOVL    40(R0), R0                             ;
                 63        0200   8F  B0  00014                 MOVW    #512, FAO_DESC                         ; 0665
         04      A3          04   AE  9E  00019                 MOVAB   FAO_BUFFER, FAO_DESC+4                 ; 0666
         F8      A3          03   A0  9B  0001E                 MOVZBW  3(R0), FILE_DESC                       ; 0668
         FC      A3          04   A0  D0  00023                 MOVL    4(R0), FILE_DESC+4                     ; 0669
                             03   A0  95  00028                 TSTB    3(R0)                                  ; 0671
                                  0A  12  0002B                 BNEQ    1$                                     ;
         F8      A3          0B   A0  9B  0002D                 MOVZBW  11(R0), FILE_DESC                      ; 0673
         FC      A3          0C   A0  D0  00032                 MOVL    12(R0), FILE_DESC+4                    ; 0674
                             F8   A3  9F  00037 1$:             PUSHAB  FILE_DESC                              ; 0682
                                  53  DD  0003A                 PUSHL   R3                                     ;
                             08   AE  9F  0003C                 PUSHAB  LENGTH                                 ;
                           0000'  CF  9F  0003F                 PUSHAB  P.ABQ                                  ; 0684
  00000000G      00               04  FB  00043                 CALLS   #4, SYS$FAO                            ;
                 52               50  D0  0004A                 MOVL    R0, STATUS                             ;
                 23               52  E9  0004D                 BLBC    STATUS, 2$                             ; 0682
                 63               6E  B0  00050                 MOVW    LENGTH, FAO_DESC                       ; 0699
                                  53  DD  00053                 PUSHL   R3                                     ; 0701
                             08   A3  9F  00055                 PUSHAB  REPLY_DESC                             ;
  00000000G      00               02  FB  00058                 CALLS   #2, LIB$GET_COMMAND                    ;
                 52               50  D0  0005F                 MOVL    R0, STATUS                             ;
                 1B               52  E8  00062                 BLBS    STATUS, 3$                             ;
```

```
                          0001827A   8F          52  D1 00065         CMPL     STATUS, #98938                       ; 0706
                                                 05  12 0006C         BNEQ     2$                                   ;
        52          03               00          04  F0 0006E         INSV     #4, #0, #3, STATUS                   ; 0707
                                                 52  DD 00073 2$:     PUSHL    STATUS                               ; 0709
                          00000000G  00          01  FB 00075         CALLS    #1, LIB$SIGNAL                       ;
                                     50          52  D0 0007C         MOVL     STATUS, R0                           ; 0710
                                                     04 0007F         RET                                           ;
                                           08    A3  B5 00080 3$:     TSTW     REPLY_DESC                           ; 0719
                                                 2C  13 00083         BEQL     5$                                   ;
                                     50    0C    B3  9A 00085         MOVZBL   @REPLY_DESC+4, CHAR                  ; 0722
                          00000059   8F          50  D1 00089         CMPL     CHAR, #89                            ; 0724
                                                 1B  13 00090         BEQL     4$                                   ;
                          00000079   8F          50  D1 00092         CMPL     CHAR, #121                           ; 0725
                                                 12  13 00099         BEQL     4$                                   ;
                          00000054   8F          50  D1 0009B         CMPL     CHAR, #84                            ; 0726
                                                 09  13 000A2         BEQL     4$                                   ;
                          00000074   8F          50  D1 000A4         CMPL     CHAR, #116                           ; 0727
                                                 04  12 000AB         BNEQ     5$                                   ;
                                     50          01  D0 000AD 4$:     MOVL     #1, R0                               ; 0729
                                                     04 000B0         RET                                           ;
                                                 50  D4 000B1 5$:     CLRL     R0                                   ; 0731
                                                     04 000B3         RET                                           ;

; Routine Size:  180 bytes,    Routine Base:  $CODE$ + 0304
```

```
1 ROUTINE log_results (fab,final_prot): =
1
1 !----
1 !
1 ! Functional description
1 !
1 !       This routine is called from the main loop whenever
1 !       logging is requested
1 !
1 ! Input parameters
1 !
1 !       fab = Address of block describing the file
1 !       fab$l_nam = pointer to name block
1 !
1 ! Output parameters
1 !
1 !       First error encountered, or TRUE is RETURNed
1 !
1 !----
1
2 BEGIN
2
2 LITERAL
2     pbufsize = 32;                          ! Buffer for generating string
2
2 MAP fab: REF BBLOCK;                        ! Define fab block format
2
2 BIND nam = .fab[fab$l_nam]: BBLOCK;         ! Define name block
2
2 LOCAL
2     status:,                                ! Recieves status
2     pbuf:       VECTOR[pbufsize,BYTE],      ! Place for protection string
2     pdesc:      VECTOR[2],                  ! Temporary string descriptor
2     desc:       VECTOR[2],                  ! Temporary string descriptor
2     prot_table: VECTOR[4];                  ! Protection string table
2
2 IF .nam[nam$b_rsl] NEQ 0                    ! If result string nonblank,
3 THEN BEGIN
3     desc[0] = .nam[nam$b_rsl];              ! then display it
3     desc[1] = .nam[nam$l_rsa];
3     END
2 ELSE IF .nam[nam$b_esl] NEQ 0               ! Or if expanded name nonblank
3 THEN BEGIN
3     desc[0] = .nam[nam$b_esl];              ! then display it
3     desc[1] = .nam[nam$l_esa];
3     END
3 ELSE BEGIN
3     desc[0] = .fab[fab$b_fns];              ! Otherwise, use original
3     desc[1] = .fab[fab$l_fna];              ! name string in FAB
2     END;
2
2 !
2 ! Now build the resultant protection string from the value passed
2 ! in the call.
2 !
2
2 expand_prot(                                ! Call sub with
```

```
;  802        0789  2              prot_table,                       !  -place for the result
;  803        0790  2              .final_prot);                     !  -final protection value
;  804        0791  2
;  805        0792  2   pdesc[0] = pbufsize;                         ! Initialize descriptor size
;  806        0793  2   pdesc[1] = pbuf;                             ! Initialize descriptor address
;  807        0794  2
;  808        0795  3   IF NOT (status =
;  809      P 0796  3   $FAOL   (                                    ! Call system service with
;  810      P 0797  3          CTRSTR = ADDRDESC ('S:!AS,O:!AS,G:!AS,W:!AS'),  ! -FAO string
;  811      P 0798  3          OUTLEN = pdesc[0],                    !  -place for resultant length
;  812      P 0799  3          OUTBUF = pdesc,                       !  -output buffer descriptor
;  813        0800  4          PRMLST = prot_table)                  !  -address of list of args
;  814        0801  3   ) THEN BEGIN
;  815        0802  3          write_message (.status);              ! Oops, tell the user
;  816        0803  3          return (.status);                     ! And exit immediately
;  817        0804  2          END;
;  818        0805  2
;  819      P 0806  2   write_message (set$_protected,               ! Inform user with
;  820      P 0807  2                  2,                            !  -two FAO arguements
;  821      P 0808  2                  desc,                         !  -file name
;  822        0809  2                  pdesc);                       !  -new protection
;  823        0810  2
;  824        0811  2   RETURN (true);
;  825        0812  2
;  826        0813  1   END;


                                                                          .PSECT  $PLIT$,NOWRT,NOEXE,2

21  3A  47  2C  53  41  21  3A  4F  2C  53  41  21  3A  53  00240 P.ABT:  .ASCII  \S:!AS,O:!AS,G:!AS,W:!AS\<0>          ;
                            00  53  41  21  3A  57  2C  53  41  0024F                                                  ;
                                              00000017  00258 P.ABS:  .LONG   23                                      ;
                                              00000000' 0025C         .ADDRESS P.ABT                                  ;

                                                                          .EXTRN  SYS$FAOL

                                                                          .PSECT  $CODE$,NOWRT,2

                                              000C 00000 LOG_RESULTS:
                                                                          .WORD   Save R2,R3                          ; 0732
                          53 00000000G  00  9E 00002                     MOVAB   LIB$SIGNAL, R3
                          5E        C0  AE  9E 00009                     MOVAB   -64(SP), SP                          ;
                          51        04  AC  D0 0000D                     MOVL    FAB, R1                              ; 0759
                          50        28  A1  D0 00011                     MOVL    40(R1), R0                           ;
                                    03  A0  95 00015                     TSTB    3(R0)                                ; 0768
                                    0C  13  00018                        BEQL    1$                                   ;
                 10  AE             03  A0  9A 0001A                     MOVZBL  3(R0), DESC                          ; 0770
                 14  AE             04  A0  D0 0001F                     MOVL    4(R0), DESC+4                        ; 0771
                                    1B  11  00024                        BRB     3$                                   ; 0768
                                    0B  A0  95 00026 1$:                 TSTB    11(R0)                               ; 0773
                                    0C  13  00029                        BEQL    2$                                   ;
                 10  AE             0B  A0  9A 0002B                     MOVZBL  11(R0), DESC                         ; 0775
                 14  AE             0C  A0  D0 00030                     MOVL    12(R0), DESC+4                       ; 0776
                                    0A  11  00035                        BRB     3$                                   ; 0773
                 10  AE             34  A1  9A 00037 2$:                 MOVZBL  52(R1), DESC                         ; 0779
                 14  AE             2C  A1  D0 0003C                     MOVL    44(R1), DESC+4                       ; 0780
```

```
                           08  AC  DD 00041 3$:     PUSHL   FINAL_PROT                    ; 0790
                           04  AE  9F 00044         PUSHAB  PROT_TABLE                    ; 0788
         0000V  CF             02  FB 00047         CALLS   #2, EXPAND_PROT               ;
            18  AE             20  D0 0004C         MOVL    #32, PDESC                    ; 0792
            1C  AE     20      AE  9E 00050         MOVAB   PBUF, PDESC+4                 ; 0793
                               5E  DD 00055         PUSHL   SP                            ; 0800
                           1C  AE  9F 00057         PUSHAB  PDESC                         ;
                           20  AE  9F 0005A         PUSHAB  PDESC                         ;
                        0000'  CF  9F 0005D         PUSHAB  P.ABS                         ;
     00000000G  00             04  FB 00061         CALLS   #4, SYS$FAOL                  ;
            52                 50  D0 00068         MOVL    R0, STATUS                    ;
            09                 52  E8 0006B         BLBS    STATUS, 4$                    ;
                               52  DD 0006E         PUSHL   STATUS                        ; 0802
            63                 01  FB 00070         CALLS   #1, LIB$SIGNAL                ;
            50                 52  D0 00073         MOVL    STATUS, R0                    ; 0803
                               04     00076         RET                                   ;
                           18  AE  9F 00077 4$:     PUSHAB  PDESC                         ; 0809
                           14  AE  9F 0007A         PUSHAB  DESC                          ;
                               02  DD 0007D         PUSHL   #2                            ;
                    007712AB   8F  DD 0007F         PUSHL   #7803563                      ;
            63                 04  FB 00085         CALLS   #4, LIB$SIGNAL                ;
            50                 01  D0 00088         MOVL    #1, R0                        ; 0811
                               04     0008B         RET                                   ; 0813

; Routine Size:  140 bytes,    Routine Base:  $CODE$ + 03B8
```

```
1 ROUTINE expand_prot (table, protection): =
1
1 !----
1 !
1 ! Functional description
1 !
1 !       This routine fills a given VECTOR with the addresses of
1 !       strings corresponding to a given protection word.
1 !
1 ! Input parameters
1 !
1 !       table = Address of the table to be filled in.
1 !       protection = Protection word.
1 !
1 ! Output parameters
1 !
1 !       table has been filled in with the addresses of descriptors
1 !       of strings describing each type of user (SYS,OWN,GRP,WORLD).
1 !
1 !----
1
2 BEGIN
2
2 BIND
2     prot_table = .table: VECTOR[4];     ! Table of addresses
2
2 OWN
2     prot_values: VECTOR[16] INITIAL(    ! Protection descriptions
2         ADDRDESC('RWED'),
2         ADDRDESC('WED'),
2         ADDRDESC('RED'),
2         ADDRDESC('ED'),
2         ADDRDESC('RWD'),
2         ADDRDESC('WD'),
2         ADDRDESC('RD'),
2         ADDRDESC('D'),
2         ADDRDESC('RWE'),
2         ADDRDESC('WE'),
2         ADDRDESC('RE'),
2         ADDRDESC('E'),
2         ADDRDESC('RW'),
2         ADDRDESC('W'),
2         ADDRDESC('R'),
2         ADDRDESC(''));
2
2 INCR index FROM 0 TO 3 DO
2     prot_table[.index] = .prot_values [.protection<.index*4,4>];
2
2 RETURN (true);           ! Always return true
2
1 END;


                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

     44  45  57  52  00260 P.ABV:   .ASCII  \RWED\                      ;
```

```
                        00000004  00264 P.ABU:  .LONG   4
                        00000000' 00268         .ADDRESS P.ABV
             00  44  45  57  0026C P.ABX:  .ASCII  \WED\<0>
                        00000003  00270 P.ABW:  .LONG   3
                        00000000' 00274         .ADDRESS P.ABX
             00  44  45  52  00278 P.ABZ:  .ASCII  \RED\<0>
                        00000003  0027C P.ABY:  .LONG   3
                        00000000' 00280         .ADDRESS P.ABZ
             00  00  44  45  00284 P.ACB:  .ASCII  \ED\<0><0>
                        00000002  00288 P.ACA:  .LONG   2
                        00000000' 0028C         .ADDRESS P.ACB
             00  44  57  52  00290 P.ACD:  .ASCII  \RWD\<0>
                        00000003  00294 P.ACC:  .LONG   3
                        00000000' 00298         .ADDRESS P.ACD
             00  00  44  57  0029C P.ACF:  .ASCII  \WD\<0><0>
                        00000002  002A0 P.ACE:  .LONG   2
                        00000000' 002A4         .ADDRESS P.ACF
             00  00  44  52  002A8 P.ACH:  .ASCII  \RD\<0><0>
                        00000002  002AC P.ACG:  .LONG   2
                        00000000' 002B0         .ADDRESS P.ACH
             00  00  00  44  002B4 P.ACJ:  .ASCII  \D\<0><0><0>
                        00000001  002B8 P.ACI:  .LONG   1
                        00000000' 002BC         .ADDRESS P.ACJ
             00  45  57  52  002C0 P.ACL:  .ASCII  \RWE\<0>
                        00000003  002C4 P.ACK:  .LONG   3
                        00000000' 002C8         .ADDRESS P.ACL
             00  00  45  57  002CC P.ACN:  .ASCII  \WE\<0><0>
                        00000002  002D0 P.ACM:  .LONG   2
                        00000000' 002D4         .ADDRESS P.ACN
             00  00  45  52  002D8 P.ACP:  .ASCII  \RE\<0><0>
                        00000002  002DC P.ACO:  .LONG   2
                        00000000' 002E0         .ADDRESS P.ACP
             00  00  00  45  002E4 P.ACR:  .ASCII  \E\<0><0><0>
                        00000001  002E8 P.ACQ:  .LONG   1
                        00000000' 002EC         .ADDRESS P.ACR
             00  00  57  52  002F0 P.ACT:  .ASCII  \RW\<0><0>
                        00000002  002F4 P.ACS:  .LONG   2
                        00000000' 002F8         .ADDRESS P.ACT
             00  00  00  57  002FC P.ACV:  .ASCII  \W\<0><0><0>
                        00000001  00300 P.ACU:  .LONG   1
                        00000000' 00304         .ADDRESS P.ACV
             00  00  00  52  00308 P.ACX:  .ASCII  \R\<0><0><0>
                        00000001  0030C P.ACW:  .LONG   1
                        00000000' 00310         .ADDRESS P.ACX
                                  00314 P.ACZ:  .BLKB   0
                        00000000  00314 P.ACY:  .LONG   0
                        00000000' 00318         .ADDRESS P.ACZ

                                                .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00450 PROT_VALUES:
                                                .ADDRESS P.ABU, P.ABW, P.ABY, P.ACA, P.ACC, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00468         P.ACE, P.ACG, P.ACI, P.ACK, P.ACM, P.ACO, -
          00000000' 00000000' 00000000' 00000000' 00480         P.ACQ, P.ACS, P.ACU, P.ACW, P.ACY
```

```
                                                        .PSECT  $CODE$,NOWRT,2

                                     0004 00000 EXPAND_PROT:
                                                        .WORD   Save R2                                  ; 0814
                                  50  D4 00002          CLRL    INDEX                                    ; 0859
            52          50        02  78 00004 1$:      ASHL    #2, INDEX, R2                            ; 0860
 51     08  AC          04        52  EF 00008          EXTZV   R2, #4, PROTECTION, R1                   ;
                  04 BC40   0000'CF41  D0 0000E         MOVL    PROT_VALUES[R1], @TABLE[INDEX]           ;
            EA          50        03  F3 00016          AOBLEQ  #3, INDEX, 1$                            ;
                        50        01  D0 0001A          MOVL    #1, R0                                   ; 0862
                                  04 0001D              RET                                              ; 0864

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0444
```

```
1 ROUTINE parse_class (desc) =
1
1 !---
1 !
1 ! This routine parses one class of user (e.g. SYSTEM, OWNER, GROUP, WORLD)
1 ! to see what protection is allowed.  The value returned in the low 4 bits
1 ! is the protection code, with the bits set to reflect that access is
1 ! requested.  Note that this is exactly the opposite of what the system wants.
1 !
1 ! Inputs:
1 !
1 !       DESC -- a descriptor pointing to the ASCII representation of the
1 !               protection desired
1 !
1 !---
1
2 BEGIN
2
2 MAP desc : REF $BBLOCK;
2
2 LOCAL
2     pointer,                              ! Pointer to string
2     result;                               ! Resultant protection
2
2 !
2 ! Initially set the value to all zeros, no access
2 !
2 result = 0;
2
2 !
2 ! Scan for the occurrence of each keyletter, and, if it is there, set the
2 ! appropriate bit.
2 !
2 pointer = .desc[dsc$a_pointer];
2 INCR index FROM 1 to .desc[dsc$w_length] DO
3     BEGIN
3     LOCAL char : BYTE;
3     char = CH$RCHAR_A(pointer);
3     IF .char EQL'R'
3     THEN result = .result OR %X'1'
3     ELSE IF .char EQL  'W'
3     THEN result = .result OR %X'2'
3     ELSE IF .char EQL 'E'
3         OR .char EQL 'P'
3     THEN result = .result OR %X'4'
3     ELSE IF .char EQL 'D'
3         OR .char EQL 'L'
3     THEN result = .result OR %X'8'
3     ELSE SIGNAL_STOP (set$_syntax, 1, .desc);
2     END;
2
2 RETURN .result;
1 END;
```

```
                                   007C 00000 PARSE_CLASS:
                                                      .WORD   Save R2,R3,R4,R5,R6                 ; 0865
                 52        04    AC   D0 00002        MOVL    DESC, R2                            ; 0898
                 56        04    A2   D0 00006        MOVL    4(R2), POINTER
                 55              62   3C 0000A        MOVZWL  (R2), R5                            ; 0899
                                 53   7C 0000D        CLRQ    INDEX
                                 4C   11 0000F        BRB     8$
                 50              86   90 00011 1$:    MOVB    (POINTER)+, CHAR                    ; 0902
            52   8F              50   91 00014        CMPB    CHAR, #82                           ; 0903
                                 05   12 00018        BNEQ    2$
                 54              01   88 0001A        BISB2   #1, RESULT                          ; 0904
                                 3E   11 0001D        BRB     8$
            57   8F              50   91 0001F 2$:    CMPB    CHAR, #87                           ; 0905
                                 05   12 00023        BNEQ    3$
                 54              02   88 00025        BISB2   #2, RESULT                          ; 0906
                                 33   11 00028        BRB     8$
            45   8F              50   91 0002A 3$:    CMPB    CHAR, #69                           ; 0907
                                 06   13 0002E        BEQL    4$
            50   8F              50   91 00030        CMPB    CHAR, #80                           ; 0908
                                 05   12 00034        BNEQ    5$
                 54              04   88 00036 4$:    BISB2   #4, RESULT                          ; 0909
                                 22   11 00039        BRB     8$
            44   8F              50   91 0003B 5$:    CMPB    CHAR, #68                           ; 0910
                                 06   13 0003F        BEQL    6$
            4C   8F              50   91 00041        CMPB    CHAR, #76                           ; 0911
                                 05   12 00045        BNEQ    7$
                 54              08   88 00047 6$:    BISB2   #8, RESULT                          ; 0912
                                 11   11 0004A        BRB     8$
                                 52   DD 0004C 7$:    PUSHL   R2                                  ; 0913
                                 01   DD 0004E        PUSHL   #1
                        007710FC 8F   DD 00050        PUSHL   #7803132
        00000000G  00            03   FB 00056        CALLS   #3, LIB$STOP
   B0              53            55   F3 0005D 8$:    AOBLEQ  R5, INDEX, 1$                       ; 0899
                   50            54   D0 00061        MOVL    RESULT, R0                          ; 0916
                                      04 00064        RET                                         ; 0917

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 0462
```

```
  1 ROUTINE search_error (fab): =
  1
  1 !----
  1 !
  1 ! Functional description
  1 !
  1 !         This routine is called from RMS whenever an error
  1 !         occurs during an RMS file function call.
  1 !
  1 ! Input parameters
  1 !
  1 !         fab = Address of block used in the RMS call.
  1 !         fab$l_nam = pointer to name block
  1 !
  1 ! Output parameters
  1 !
  1 !         FAB status is returned
  1 !
  1 !----
  1
  2 BEGIN
  2
  2 MAP fab: REF BBLOCK;                          ! Define fab block format
  2 BIND nam = .fab[fab$l_nam]: BBLOCK;           ! Define name block
  2
  2 LOCAL desc: VECTOR[2];                        ! Temporary string descriptor
  2
  2 IF .nam[nam$b_rsl] NEQ 0                      ! If result string nonblank,
  3 THEN BEGIN
  3     desc[0] = .nam[nam$b_rsl];                ! then display it
  3     desc[1] = .nam[nam$l_rsa];
  3     END
  2 ELSE IF .nam[nam$b_esl] NEQ 0                 ! Or if expanded name nonblank
  3 THEN BEGIN
  3     desc[0] = .nam[nam$b_esl];                ! then display it
  3     desc[1] = .nam[nam$l_esa];
  3     END
  3 ELSE BEGIN
  3     desc[0] = .fab[fab$b_fns];                ! Otherwise, use original
  3     desc[1] = .fab[fab$l_fna];                ! name string in FAB
  2     END;
  2
P 2 write_message(set$_searchfail,1,DESC,       ! Output an error message
P 2                 .fab[fab$l_sts],            ! with fab error code
  2                 .fab[fab$l_stv]);           ! and secondary code
  2
  2 RETURN (.fab[fab$l_sts]);
  2
  1 END;


                            0004 00000 SEARCH_ERROR:
                                              .WORD   Save R2                   ; 0918
              5E            08  C2 00002      SUBL2   #8, SP                    ;
```

```
                    52        04   AC  D0 00005              MOVL    FAB, R2                                   ; 0941
                    50        28   A2  D0 00009              MOVL    40(R2), R0                                ;
                              03   A0  95 0000D              TSTB    3(R0)                                     ; 0945
                                   0B  13 00010              BEQL    1$                                        ;
                    6E        03   A0  9A 00012              MOVZBL  3(R0), DESC                               ; 0947
               04   AE        04   A0  D0 00016              MOVL    4(R0), DESC+4                             ; 0948
                                   19  11 0001B              BRB     3$                                        ; 0945
                              0B   A0  95 0001D  1$:         TSTB    11(R0)                                    ; 0950
                                   0B  13 00020              BEQL    2$                                        ;
                    6E        0B   A0  9A 00022              MOVZBL  11(R0), DESC                              ; 0952
               04   AE        0C   A0  D0 00026              MOVL    12(R0), DESC+4                            ; 0953
                                   09  11 0002B              BRB     3$                                        ; 0950
                    6E        34   A2  9A 0002D  2$:         MOVZBL  52(R2), DESC                              ; 0956
               04   AE        2C   A2  D0 00031              MOVL    44(R2), DESC+4                            ; 0957
                    7E        08   A2  7D 00036  3$:         MOVQ    8(R2), -(SP)                              ; 0962
                              08   AE  9F 0003A              PUSHAB  DESC                                      ;
                                   01  DD 0003D              PUSHL   #1                                        ;
                         0077123A  8F  DD 0003F              PUSHL   #7803450                                  ;
          00000000G  00            05  FB 00045              CALLS   #5, LIB$SIGNAL                            ;
                     50       08   A2  D0 0004C              MOVL    8(R2), R0                                 ; 0964
                                       04 00050              RET                                               ; 0966

; Routine Size:  81 bytes,    Routine Base:  $CODE$ + 04C7
```

```
;  984      0967  1 ROUTINE cli_error (cli_block,error): =
;  985      0968  1
;  986      0969  1 !----
;  987      0970  1 !
;  988      0971  1 ! Functional description
;  989      0972  1 !
;  990      0973  1 !       This routine is called as an error handler for
;  991      0974  1 !       CLI errors.
;  992      0975  1 !
;  993      0976  1 ! Input parameters
;  994      0977  1 !
;  995      0978  1 !       cli_block = Address of CLI request block
;  996      0979  1 !       error = CLI error message
;  997      0980  1 !
;  998      0981  1 ! Output parameters
;  999      0982  1 !
; 1000      0983  1 !       Input 'error' is RETURNed
; 1001      0984  1 !
; 1002      0985  1 !----
; 1003      0986  1
; 1004      0987  2 BEGIN
; 1005      0988  2
; 1006      0989  2 MAP cli_block: REF BBLOCK;             ! Define CLI request block
; 1007      0990  2
; 1008      0991  2 signal_stop(.error);
; 1009      0992  2
; 1010      0993  2 RETURN (.error);
; 1011      0994  2
; 1012      0995  1 END;


                                   0000 00000 CLI_ERROR:
                                                 .WORD   Save nothing                   ; 0967
                           08   AC  DD 00002     PUSHL   ERROR                          ; 0991
           00000000G  00        01  FB 00005     CALLS   #1, LIB$STOP                   ;
                      50   08   AC  D0 0000C     MOVL    ERROR, R0                      ; 0993
                                    04 00010     RET                                    ; 0995

; Routine Size:  17 bytes,    Routine Base:  $CODE$ + 0518
```

```
; 1014          0996  1 END
; 1015          0997  0 ELUDOM



                                                                 .EXTRN  LIB$SIGNAL, LIB$STOP

;                                 PSECT SUMMARY
;
;       Name                    Bytes                          Attributes
;
;  $OWN$                          1168  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                          796  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         1321  NOVEC,NOWRT,  RD , EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                          Library Statistics
;
;                                         -------- Symbols --------         Pages        Processing
;       File                              Total    Loaded   Percent         Mapped       Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1       9776       58         0            581         00:01.0





;                                 COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:SETPRO/OBJ=OBJ$:SETPRO MSRC$:SETPRO/UPDATE=(ENH$:SETPRO)

; Size:           1321 code + 1964 data bytes
; Run Time:          00:27.2
; Elapsed Time:      01:59.6
; Lines/CPU Min:      2202
; Lexemes/CPU-Min: 21945
; Memory Used:  154 pages
; Compilation Complete
```

0053 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SETFILE
LIS

SETP0MESS
LIS

SETP0DISP
LIS

SETMISC
LIS

SETPRO
LIS

SETMAIN
LIS

SETPROCES
LIS

SETSHOBRO
LIS

SETVOLUME
LIS

SETPWD
LIS

SETTERM
LIS

SETQUEUE
LIS

SETTIME
LIS