```
   CCCCCCCCCCC  LLL              IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCC  LLL              IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCC  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
   CCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
```

FILEID**SHOWAUDIT

```
  SSSSSSSS  HH      HH    OOOOOO    WW      WW    AAAAAA    UU      UU  DDDDDDDD     IIIIII    TTTTTTTTTT
  SSSSSSSS  HH      HH    OOOOOO    WW      WW    AAAAAA    UU      UU  DDDDDDDD     IIIIII    TTTTTTTTTT
SS          HH      HH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
SS          HH      HH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
SS          HH      HH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
SS          HH      HH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
  SSSSSS    HHHHHHHHHH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
  SSSSSS    HHHHHHHHHH   OO    OO   WW      WW   AA    AA   UU      UU  DD      DD     II        TT
        SS  HH      HH   OO    OO   WW  WW  WW   AAAAAAAAAA   UU      UU  DD      DD     II        TT
        SS  HH      HH   OO    OO   WW  WW  WW   AAAAAAAAAA   UU      UU  DD      DD     II        TT
        SS  HH      HH   OO    OO   WWWW WWWW   AA    AA   UU      UU  DD      DD     II        TT
        SS  HH      HH   OO    OO   WWWW WWWW   AA    AA   UU      UU  DD      DD     II        TT
SSSSSSSS    HH      HH    OOOOOO    WW      WW   AA    AA   UUUUUUUUUU  DDDDDDDD     IIIIII    TT
SSSSSSSS    HH      HH    OOOOOO    WW      WW   AA    AA   UUUUUUUUUU  DDDDDDDD     IIIIII    TT

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
```

```
;     1          0001  0 MODULE showaudit ( IDENT = 'V04-000',
;     2          0002  0                    ADDRESSING_MODE (EXTERNAL = GENERAL)) =
;     3          0003  1 BEGIN
;     4          0004  1
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !*                                                                           *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                      *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE     *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER      *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    17          0017  1 !*  TRANSFERRED.                                                              *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    21          0021  1 !*  CORPORATION.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 ! FACILITY:  SHOW Command
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This module implements the DCL command SHOW AUDIT.
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:
;    37          0037  1 !
;    38          0038  1 !       VAX/VMS operating system, user and kernel mode
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:  Gerry Smith                              29-Jun-1983
;    41          0041  1 !
;    42          0042  1 ! Modified by:
;    43          0043  1 !
;    44          0044  1 !       V03-005 RSH0103         R. Scott Hanna          17-Feb-1984
;    45          0045  1 !               Fix field test 1 problems, comment out journaling code
;    46          0046  1 !               and make changes due to new layout of $NSAEVTDEF.
;    47          0047  1 !
;    48          0048  1 !       V03-004 RSH0102         R. Scott Hanna          05-Feb-1984
;    49          0049  1 !               Temporarily disable SHOW AUDIT.
;    50          0050  1 !
;    51          0051  1 !       V03-003 GAS0190         Gerry Smith             22-Sep-1983
;    52          0052  1 !               Fix an index which caused an accvio in file_access display.
;    53          0053  1 !
;    54          0054  1 !       V03-002 GAS0176         Gerry Smith              9-Sep-1983
;    55          0055  1 !               Add more comments, remove some kludges, and straighten
;    56          0056  1 !               up the displays.
;    57          0057  1 !
```

```
;   58          0058  1 !       V03-001 GAS0171          Gerry Smith                24-Aug-1983
;   59          0059  1 !               Remove mailbox and terminal I/O, add remote and
;   60          0060  1 !               interactive login/out.
;   61          0061  1 !
;   62          0062  1 !--
```

```
;    64          0063  1 !
;    65          0064  1 ! Include files
;    66          0065  1 !
;    67          0066  1 LIBRARY 'SYS$LIBRARY:LIB';                      ! VAX/VMS common definitions
;    68          0067  1 !REQUIRE 'SHRLIB$:JNLDEFINT';                  ! Journal definitions
;    69          0068  1
;    70          0069  1 !
;    71          0070  1 ! Define the linkage for the routine that obtains the device name.
;    72          0071  1 !
;    73          0072  1 !LINKAGE
;    74          0073  1 !    CVTDEV = JSB (REGISTER = 0,                        ! Length of output buffer,
;    75          0074  1 !                  REGISTER = 1,                      ! Address of output buffer
;    76          0075  1 !                  REGISTER = 4,                      ! Format of device name
;    77          0076  1 !                  REGISTER = 5;                      ! Address of UCB
;    78          0077  1 !                  REGISTER = 1);                     ! Length of final name
;    79          0078  1
```

```
;    81          0079  1 !
;    82          0080  1 ! Declare some storage for tables
;    83          0081  1 !
;    84          0082  1 $ASSUME($BITPOSITION(nsa$v_evt_spare), EQL, 3)
;    85          0083  1 OWN
;    86          0084  1     sys_events : VECTOR[3]                          ! System events
;    87          0085  1                  INITIAL(%ASCID 'ACL',
;    88          0086  1                          %ASCID 'MOUNT',
;    89          0087  1                          %ASCID 'AUTHORIZATION'),
;    90          0088  1     file_events : VECTOR[8]                         ! File access events
;    91          0089  1                  INITIAL(%ASCID 'FAILURE',
;    92          0090  1                          %ASCID 'SUCCESS',
;    93          0091  1                          %ASCID 'SYSPRV',
;    94          0092  1                          %ASCID 'BYPASS',
;    95          0093  1                          %ASCID 'UPGRADE',
;    96          0094  1                          %ASCID 'DOWNGRADE',
;    97          0095  1                          %ASCID 'GRPPRV',
;    98          0096  1                          %ASCID 'READALL'),
;    99          0097  1     loginout_events : VECTOR[4]                     ! Loginout events
;   100          0098  1                       INITIAL(%ASCID 'BREAKIN',
;   101          0099  1                               %ASCID 'LOGIN',
;   102          0100  1                               %ASCID 'LOGFAILURE',
;   103          0101  1                               %ASCID 'LOGOUT'),
;   104          0102  1     loginout_types : VECTOR[7]                      ! Types of login/logout
;   105          0103  1                      INITIAL(%ASCID 'BATCH,',
;   106          0104  1                              %ASCID 'DIALUP,',
;   107          0105  1                              %ASCID 'LOCAL,',
;   108          0106  1                              %ASCID 'REMOTE,',
;   109          0107  1                              %ASCID 'NETWORK,',
;   110          0108  1                              %ASCID 'SUBPROCESS,',
;   111          0109  1                              %ASCID 'DETACHED,'),
;   112          0110  1     access_types : VECTOR[5]                        ! Types of file access
;   113          0111  1                    INITIAL(%ASCID 'READ,',
;   114          0112  1                            %ASCID 'WRITE,',
;   115          0113  1                            %ASCID 'EXECUTE,',
;   116          0114  1                            %ASCID 'DELETE,',
;   117          0115  1                            %ASCID 'CONTROL,');
```

```
;   119          0116  1 !
;   120          0117  1 ! Table of contents
;   121          0118  1 !
;   122          0119  1 FORWARD ROUTINE
;   123          0120  1     show$audit : NOVALUE,                   ! Main module of SHOW AUDIT
;   124          0121  1     decode_bits : NOVALUE;                  ! Make sense of all the bits
;   125          0122  1 !   get_device;                             ! Get device journal is going to
;   126          0123  1
;   127          0124  1 !
;   128          0125  1 ! Library routines
;   129          0126  1 !
;   130          0127  1 EXTERNAL ROUTINE
;   131          0128  1     show$write_line : NOVALUE,
;   132          0129  1 !   ioc$cvt_devnam : CVTDEV,
;   133          0130  1     str$append;
;   134          0131  1
;   135          0132  1
;   136          0133  1 !
;   137          0134  1 ! Declare literals defined elsewhere
;   138          0135  1 !
;   139          0136  1 !EXTERNAL LITERAL
;   140          0137  1 !    show$_audreaderr,                      ! Error getting alarm/journal bits
;   141          0138  1 !    show$_auddeverr;                       ! Error getting journal device
;   142          0139  1
;   143          0140  1
;   144          0141  1 !
;   145          0142  1 ! Declare some cells in the exec
;   146          0143  1 !
;   147          0144  1 EXTERNAL
;   148          0145  1     ctl$gq_procpriv : $BBLOCK,
;   149          0146  1 !   ctl$gl_ccbbase,
;   150          0147  1 !   nsa$gr_journvec,
;   151          0148  1     nsa$gr_alarmvec;
;   152          0149  1
```

```
;  154     0150  1 GLOBAL ROUTINE show$audit : NOVALUE =
;  155     0151  2 BEGIN
;  156     0152  2
;  157     0153  2 !++
;  158     0154  2 ! Functional description
;  159     0155  2 !
;  160     0156  2 !       This is the routine for the SHOW AUDIT command.  It is called
;  161     0157  2 !       from the SHOW command processor, and displays the class of events
;  162     0158  2 !       for which security audits and alarms are enabled.  If there is a
;  163     0159  2 !       security journal available, it also displays the device on which
;  164     0160  2 !       the journal file resides.
;  165     0161  2 !
;  166     0162  2 ! Inputs
;  167     0163  2 !       None
;  168     0164  2 !
;  169     0165  2 ! Outputs
;  170     0166  2 !       None
;  171     0167  2 !
;  172     0168  2 !--
;  173     0169  2
;  174     0170  2 LOCAL
;  175     0171  2 !   status,
;  176     0172  2     flags : VECTOR[nsa$k_evt_length,BYTE];
;  177     0173  2 !   arglist : VECTOR[2],
;  178     0174  2 !   device : VECTOR[2],
;  179     0175  2 !   dev_string : VECTOR[10];
;  180     0176  2
;  181     0177  2 !
;  182     0178  2 ! See if the user has the SECURITY privilege.
;  183     0179  2 !
;  184     0180  2 IF NOT .ctl$gq_procpriv[prv$v_security]
;  185     0181  2 THEN
;  186     0182  3     BEGIN
;  187     0183  3     SIGNAL(ss$_nosecurity);
;  188     0184  3     RETURN;
;  189     0185  2     END;
;  190     0186  2
;  191     0187  2 !
;  192     0188  2 ! Get and decode the bits in the alarm vector.
;  193     0189  2 !
;  194     0190  2 CH$MOVE(nsa$k_evt_length,                       ! Copy the alarm bits
;  195     0191  2         nsa$gr_alarmvec,                        ! to program local
;  196     0192  2         flags);                                 ! storage.
;  197     0193  2 IF CH$EQL(nsa$k_evt_length,                     ! If no bits are
;  198     0194  2           flags,                                ! set,
;  199     0195  2           nsa$k_evt_length,
;  200     0196  2           UPLIT BYTE(REP nsa$k_evt_length OF (0)))
;  201     0197  2 THEN show$write_line(%ASCID 'Security alarms currently disabled',
;  202     0198  2                      %REF(0))
;  203     0199  2 ELSE
;  204     0200  3     BEGIN
;  205     0201  3     show$write_line(%ASCID 'Security alarms currently enabled for:',
;  206     0202  3                     %REF(0));
;  207     0203  3     decode_bits(flags);
;  208     0204  3     show$write_line(%ASCID ' ', %REF(0));
;  209     0205  2     END;
;  210     0206  2
```

```
;  211          0207  2 !
;  212          0208  2 ! Get and decode the bits in the journal vector.
;  213          0209  2 !
;  214          0210  2 !CH$MOVE(nsa$k_evt_length,                        ! Copy the journal bits
;  215          0211  2 !       nsa$gr_journvec,                         ! to program local
;  216          0212  2 !       flags);                                  ! storage.
;  217          0213  2 !IF CH$EQL(nsa$k_evt_length,                      ! If no bits are
;  218          0214  2 !          flags,                                ! set,
;  219          0215  2 !          nsa$k_evt_length,
;  220          0216  2 !          UPLIT BYTE(REP nsa$k_evt_length OF (0)))
;  221          0217  2 !THEN show$write_line(%ASCID 'Security journaling currently disabled',
;  222          0218  2 !                     %REF(0))
;  223          0219  2 !ELSE
;  224          0220  2 !    BEGIN
;  225          0221  2 !    show$write_line(%ASCID 'Security journaling currently enabled for:',
;  226          0222  2 !                    %REF(0));
;  227          0223  2 !    decode_bits(flags);
;  228          0224  2 !    show$write_line(%ASCID ' ', %REF(0));
;  229          0225  2 !
;  230          0226  2 !
;  231          0227  2 ! If there are journal bits set, try to find out what device the journal
;  232          0228  2 ! is writing to.
;  233          0229  2 !
;  234          0230  2 !    device[0] = %ALLOCATION(dev_string);
;  235          0231  2 !    device[1] = dev_string;
;  236          0232  2 !    arglist[0] = 1;
;  237          0233  2 !    arglist[1] = device;
;  238          0234  2 !    status = $CMEXEC(ROUTIN = get_device,
;  239          0235  2 !                     ARGLST = arglist);
;  240          0236  2 !    IF .status
;  241          0237  2 !    THEN show$write_line(%ASCID 'Journaling file resides on !AS', %REF(device))
;  242          0238  2 !    ELSE SIGNAL(show$_auddeverr,
;  243          0239  2 !                0,
;  244          0240  2 !                .status);
;  245          0241  2 !    END;
;  246          0242  2
;  247          0243  2 RETURN;
;  248          0244  1 END;


                                                                .TITLE  SHOWAUDIT
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                00  4C  43  41  00000 P.AAB:   .ASCII  \ACL\<0>                                ;
                                                  010E0003  00004 P.AAA:   .LONG   17694723                                   ;
                                                  00000000' 00008          .ADDRESS P.AAB                                     ;
                                        00  00  00  54  4E  55  4F  4D  0000C P.AAD:   .ASCII  \MOUNT\<0><0><0>                  ;
                                                  010E0005  00014 P.AAC:   .LONG   17694725                                   ;
                                                  00000000' 00018          .ADDRESS P.AAD                                     ;
00  00  4E  4F  49  54  41  5A  49  52  4F  48  54  55  41  0001C P.AAF:   .ASCII  \AUTHORIZATION\<0><0><0>                     ;
                                                      00  0002B                                                              ;
                                                  010E000D  0002C P.AAE:   .LONG   17694733                                   ;
                                                  00000000' 00030          .ADDRESS P.AAF                                     ;
                                        00  45  52  55  4C  49  41  46  00034 P.AAH:   .ASCII  \FAILURE\<0>                   ;
                                                  010E0007  0003C P.AAG:   .LONG   17694727                                   ;
```

```
                                00000000'  00040                .ADDRESS P.AAH
                00  53  53  45  43  43  55  53  00044 P.AAJ:    .ASCII  \SUCCESS\<0>
                                010E0007   0004C P.AAI:    .LONG   17694727
                                00000000'  00050                .ADDRESS P.AAJ
                00  00  56  52  50  53  59  53  00054 P.AAL:    .ASCII  \SYSPRV\<0><0>
                                010E0006   0005C P.AAK:    .LONG   17694726
                                00000000'  00060                .ADDRESS P.AAL
                00  00  53  53  41  50  59  42  00064 P.AAN:    .ASCII  \BYPASS\<0><0>
                                010E0006   0006C P.AAM:    .LONG   17694726
                                00000000'  00070                .ADDRESS P.AAN
                00  45  44  41  52  47  50  55  00074 P.AAP:    .ASCII  \UPGRADE\<0>
                                010E0007   0007C P.AAO:    .LONG   17694727
                                00000000'  00080                .ADDRESS P.AAP
    00  00  00  45  44  41  52  47  4E  57  4F  44  00084 P.AAR:    .ASCII  \DOWNGRADE\<0><0><0>
                                010E0009   00090 P.AAQ:    .LONG   17694729
                                00000000'  00094                .ADDRESS P.AAR
                00  00  56  52  50  50  52  47  00098 P.AAT:    .ASCII  \GRPPRV\<0><0>
                                010E0006   000A0 P.AAS:    .LONG   17694726
                                00000000'  000A4                .ADDRESS P.AAT
                00  4C  4C  41  44  41  45  52  000A8 P.AAV:    .ASCII  \READALL\<0>
                                010E0007   000B0 P.AAU:    .LONG   17694727
                                00000000'  000B4                .ADDRESS P.AAV
                00  4E  49  4B  41  45  52  42  000B8 P.AAX:    .ASCII  \BREAKIN\<0>
                                010E0007   000C0 P.AAW:    .LONG   17694727
                                00000000'  000C4                .ADDRESS P.AAX
                00  00  00  4E  49  47  4F  4C  000C8 P.AAZ:    .ASCII  \LOGIN\<0><0><0>
                                010E0005   000D0 P.AAY:    .LONG   17694725
                                00000000'  000D4                .ADDRESS P.AAZ
    00  00  45  52  55  4C  49  41  46  47  4F  4C  000D8 P.ABB:    .ASCII  \LOGFAILURE\<0><0>
                                010E000A   000E4 P.ABA:    .LONG   17694730
                                00000000'  000E8                .ADDRESS P.ABB
                00  00  54  55  4F  47  4F  4C  000EC P.ABD:    .ASCII  \LOGOUT\<0><0>
                                010E0006   000F4 P.ABC:    .LONG   17694726
                                00000000'  000F8                .ADDRESS P.ABD
                00  00  2C  48  43  54  41  42  000FC P.ABF:    .ASCII  \BATCH,\<0><0>
                                010E0006   00104 P.ABE:    .LONG   17694726
                                00000000'  00108                .ADDRESS P.ABF
                00  2C  50  55  4C  41  49  44  0010C P.ABH:    .ASCII  \DIALUP,\<0>
                                010E0007   00114 P.ABG:    .LONG   17694727
                                00000000'  00118                .ADDRESS P.ABH
                00  00  2C  4C  41  43  4F  4C  0011C P.ABJ:    .ASCII  \LOCAL,\<0><0>
                                010E0006   00124 P.ABI:    .LONG   17694726
                                00000000'  00128                .ADDRESS P.ABJ
                00  2C  45  54  4F  4D  45  52  0012C P.ABL:    .ASCII  \REMOTE,\<0>
                                010E0007   00134 P.ABK:    .LONG   17694727
                                00000000'  00138                .ADDRESS P.ABL
                2C  4B  52  4F  57  54  45  4E  0013C P.ABN:    .ASCII  \NETWORK,\
                                010E0008   00144 P.ABM:    .LONG   17694728
                                00000000'  00148                .ADDRESS P.ABN
    00  2C  53  53  45  43  4F  52  50  42  55  53  0014C P.ABP:    .ASCII  \SUBPROCESS,\<0>
                                010E000B   00158 P.ABO:    .LONG   17694731
                                00000000'  0015C                .ADDRESS P.ABP
    00  00  00  2C  44  45  48  43  41  54  45  44  00160 P.ABR:    .ASCII  \DETACHED,\<0><0><0>
                                010E0009   0016C P.ABQ:    .LONG   17694729
                                00000000'  00170                .ADDRESS P.ABR
                00  00  00  2C  44  41  45  52  00174 P.ABT:    .ASCII  \READ,\<0><0><0>
                                010E0005   0017C P.ABS:    .LONG   17694725
```

```
                                                00000000'  00180                .ADDRESS P.ABT
                                        00  00  2C  45  54  49  52  57  00184 P.ABV:  .ASCII  \WRITE,\<0><0>
                                                010E0006   0018C P.ABU:  .LONG   17694726
                                                00000000'  00190                .ADDRESS P.ABV
                                        2C  45  54  55  43  45  58  45  00194 P.ABX:  .ASCII  \EXECUTE,\
                                                010E0008   0019C P.ABW:  .LONG   17694728
                                                00000000'  001A0                .ADDRESS P.ABX
                                        00  2C  45  54  45  4C  45  44  001A4 P.ABZ:  .ASCII  \DELETE,\<0>
                                                010E0007   001AC P.ABY:  .LONG   17694727
                                                00000000'  001B0                .ADDRESS P.ABZ
                                        2C  4C  4F  52  54  4E  4F  43  001B4 P.ACB:  .ASCII  \CONTROL,\
                                                010E0008   001BC P.ACA:  .LONG   17694728
                                                00000000'  001C0                .ADDRESS P.ACB
                                                        00#  001C4 P.ACC:  .BYTE   0[40]
73  6D  72  61  6C  61  20  79  74  69  72  75  63  65  53  001EC P.ACE:  .ASCII  \Security alarms currently disabled\<0>
61  73  69  64  20  79  6C  74  6E  65  72  72  75  63  20  001FB
                                            00  64  65  6C  62  0020A
                                                        00  0020F            .ASCII  <0>
                                                010E0022   00210 P.ACD:  .LONG   17694754
                                                00000000'  00214                .ADDRESS P.ACE
73  6D  72  61  6C  61  20  79  74  69  72  75  63  65  53  00218 P.ACG:  .ASCII  \Security alarms currently enabled for:-
62  61  6E  65  20  79  6C  74  6E  65  72  72  75  63  20  00227                \<0>
                        00  3A  72  6F  66  20  64  65  6C  00236
                                                        00  0023F            .ASCII  <0>
                                                010E0026   00240 P.ACF:  .LONG   17694758
                                                00000000'  00244                .ADDRESS P.ACG
                                            00  00  00  20  00248 P.ACI:  .ASCII  \ \<0><0><0>
                                                010E0001   0024C P.ACH:  .LONG   17694721
                                                00000000'  00250                .ADDRESS P.ACI

                                                                                .PSECT  $OWN$,NOEXE,2

                                00000000'  00000000'  00000000'  00000  SYS_EVENTS:
                                                                                .ADDRESS P.AAA, P.AAC, P.AAE
00000000'  00000000'  00000000'  00000000'  00000000'  00000000'  0000C  FILE_EVENTS:
                                                                                .ADDRESS P.AAG, P.AAI, P.AAK, P.AAM, P.AAO, -
                                00000000'  00000000'  00024                           P.AAQ, P.AAS, P.AAU
                        00000000'  00000000'  00000000'  00000000'  0002C  LOGINOUT_EVENTS:
                                                                                .ADDRESS P.AAW, P.AAY, P.ABA, P.ABC
00000000'  00000000'  00000000'  00000000'  00000000'  00000000'  0003C  LOGINOUT_TYPES:
                                                                                .ADDRESS P.ABE, P.ABG, P.ABI, P.ABK, P.ABM, -
                                                00000000'  00054                      P.ABO, P.ABQ
                00000000'  00000000'  00000000'  00000000'  00000000'  00058  ACCESS_TYPES:
                                                                                .ADDRESS P.ABS, P.ABU, P.ABW, P.ABY, P.ACA

                                                                                .EXTRN  SHOW$WRITE_LINE
                                                                                .EXTRN  STR$APPEND, CTL$GQ_PROCPRIV
                                                                                .EXTRN  NSA$GR_ALARMVEC

                                                                                .PSECT  $CODE$,NOWRT,2

                                                        007C  00000             .ENTRY  SHOW$AUDIT, Save R2,R3,R4,R5,R6  ; 0150
                                        56  00000000G   00  9E  00002          MOVAB   SHOW$WRITE_LINE, R6              ;
                                        5E              2C  C2  00009          SUBL2   #44, SP                          ;
                                0D  00000000G   00      06  E0  0000C          BBS     #6, CTL$GQ_PROCPRIV+4, 1$        ; 0180
                                        7E      2934    8F  3C  00014          MOVZWL  #10548, -(SP)                    ; 0183
                                    00000000G   00      01  FB  00019          CALLS   #1, LIB$SIGNAL                   ;
```

```
                                        04 00020        RET                                          ; 0182
       04   AE 00000000G  00       28   28 00021 1$:    MOVC3   #40, NSA$GR_ALARMVEC, FLAGS          ; 0190
     0000'  CF         04  AE      28   29 0002A        CMPC3   #40, FLAGS, P.ACC                    ; 0193
                                   0A   12 00031        BNEQ    2$                                   ;
                                   6E   D4 00033        CLRL    (SP)                                 ; 0198
                                   5E   DD 00035        PUSHL   SP                                   ;
                           0000'   CF   9F 00037        PUSHAB  P.ACD                                ; 0197
                                   1B   11 0003B        BRB     3$                                   ;
                                   6E   D4 0003D 2$:    CLRL    (SP)                                 ; 0202
                                   5E   DD 0003F        PUSHL   SP                                   ;
                           0000'   CF   9F 00041        PUSHAB  P.ACF                                ; 0201
                     66            02   FB 00045        CALLS   #2, SHOW$WRITE_LINE                  ;
                                04 AE   9F 00048        PUSHAB  FLAGS                                ; 0203
              0000V  CF            01   FB 0004B        CALLS   #1, DECODE_BITS                      ;
                                   6E   D4 00050        CLRL    (SP)                                 ; 0204
                                   5E   DD 00052        PUSHL   SP                                   ;
                           0000'   CF   9F 00054        PUSHAB  P.ACH                                ;
                     66            02   FB 00058 3$:    CALLS   #2, SHOW$WRITE_LINE                  ;
                                        04 0005B        RET                                          ; 0244

; Routine Size:  92 bytes,     Routine Base:  $CODE$ + 0000
```

```
;  250     0245  1 ROUTINE decode_bits (flags) : NOVALUE =
;  251     0246  2 BEGIN
;  252     0247  2
;  253     0248  2 !++
;  254     0249  2 !
;  255     0250  2 ! Given a set of bits, determine what security events are set.
;  256     0251  2 !
;  257     0252  2 ! Inputs:
;  258     0253  2 !       flags - address of security alarm/journal vector
;  259     0254  2 !
;  260     0255  2 ! Outputs:
;  261     0256  2 !       None.  The appropriate events are displayed.
;  262     0257  2 !
;  263     0258  2 !--
;  264     0259  2
;  265     0260  2 MAP
;  266     0261  2     flags : REF $BBLOCK;
;  267     0262  2
;  268     0263  2 BIND
;  269     0264  2     sys = flags[nsa$l_evt_sys] : BITVECTOR,
;  270     0265  2     file_access = flags[nsa$l_evt_failure] : VECTOR,
;  271     0266  2     log_event = flags[nsa$b_evt_logb] : VECTOR[,BYTE];
;  272     0267  2
;  273     0268  2 !
;  274     0269  2 ! Get the system-wide general events
;  275     0270  2 !
;  276     0271  2 INCR i FROM 0 TO $BITPOSITION(nsa$v_evt_spare)-1 DO
;  277     0272  2     IF .sys[.i] THEN show$write_line(%ASCID '    !AS', sys_events[.i]);
;  278     0273  2
;  279     0274  2 !
;  280     0275  2 ! The loginout events
;  281     0276  2 !
;  282     0277  2 INCR j FROM 0 TO 3 DO
;  283     0278  3     BEGIN
;  284     0279  3     IF .log_event[.j] NEQ 0
;  285     0280  3     THEN
;  286     0281  4         BEGIN
;  287     0282  4         BIND
;  288     0283  4             login = log_event[.j] : BITVECTOR;
;  289     0284  4         LOCAL
;  290     0285  4             desc : $BBLOCK[dsc$c_s_bln],
;  291     0286  4             arglist : VECTOR[2];
;  292     0287  4         $init_dyndesc(desc);
;  293     0288  4         INCR i FROM 0 TO 6 DO
;  294     0289  5             BEGIN
;  295     0290  5             IF .login[.i]
;  296     0291  5             THEN str$append(desc, .loginout_types[.i]);
;  297     0292  4             END;
;  298     0293  4         desc[dsc$w_length] = .desc[dsc$w_length] - 1;
;  299     0294  4         arglist[0] = .loginout_events[.j];
;  300     0295  4         arglist[1] = desc;
;  301     0296  4         show$write_line(%ASCID '    !12<!AS:!>(!AS)', arglist);
;  302     0297  3         END;
;  303     0298  2     END;
;  304     0299  2
;  305     0300  2 !
;  306     0301  2 ! The file access events.
```

```
;  307          0302  2 !
;  308          0303  2 sys[0] = 0;                                    ! Cheat, use SYS[0] as a flag.
;  309          0304  2 INCR i FROM 0 TO 7 DO                          ! Look at each type of file
;  310          0305  3     BEGIN                                      ! access cell
;  311          0306  3     IF .file_access[.i] NEQ 0                  ! If something is set,
;  312          0307  3     THEN                                       ! display the info...
;  313          0308  4         BEGIN
;  314          0309  4         BIND
;  315          0310  4             access = file_access[.i] : BITVECTOR;      ! Look at the cell
;  316          0311  4         LOCAL                                          ! as a set of bits.
;  317          0312  4             desc : $BBLOCK[dsc$c_s_bln],
;  318          0313  4             arglist : VECTOR[2];
;  319          0314  4         IF NOT .sys[0]                             ! If not yet done,
;  320          0315  4         THEN                                       ! print a header.
;  321          0316  5             BEGIN
;  322          0317  5             show$write_line(%ASCID '    FILE_ACCESS:', %REF(0));
;  323          0318  5             sys[0] = 1;                            ! Show that header
;  324          0319  4             END;                                   ! has been written.
;  325          0320  4         $init_dyndesc(desc);                       ! Setup a descriptor
;  326          0321  4         INCR j FROM 0 TO ($BITPOSITION(arm$v_fill) - 1) DO     ! Go thru each
;  327          0322  5             BEGIN                                              ! type of access
;  328          0323  5             IF .access[.j]                                     ! type; if set,
;  329          0324  5             THEN str$append(desc, .access_types[.j]);          ! add to display
;  330          0325  4             END;
;  331          0326  4         desc[dsc$w_length] = .desc[dsc$w_length] -1;   ! Strip trailing comma
;  332          0327  4         arglist[0] = .file_events[.i];                 ! Get file option
;  333          0328  4         arglist[1] = desc;                             ! and string of accesses
;  334          0329  4         show$write_line(%ASCID '        !11<!AS:!>(!AS)', arglist);
;  335          0330  3         END;
;  336          0331  2     END;
;  337          0332  2
;  338          0333  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                00  53  41  21  20  20  20  20  00254 P.ACK:   .ASCII  \     !AS\<0>
                                                    010E0007    0025C P.ACJ:   .LONG   17694727
                                                    00000000'   00260          .ADDRESS P.ACK
28  3E  21  3A  53  41  21  3C  32  31  21  20  20  20  20  20  00264 P.ACM:   .ASCII  \     !12<!AS:!>(!AS)\<0>
                                            00  29  53  41  21  00273
                                                    010E0013    00278 P.ACL:   .LONG   17694739
                                                    00000000'   0027C          .ADDRESS P.ACM
53  53  45  43  43  41  5F  45  4C  49  46  20  20  20  20  20  00280 P.ACO:   .ASCII  \     FILE_ACCESS:\
                                                        3A      0028F
                                                    010E0010    00290 P.ACN:   .LONG   17694736
                                                    00000000'   00294          .ADDRESS P.ACO
53  41  21  3C  31  31  21  20  20  20  20  20  20  20  20  20  00298 P.ACQ:   .ASCII  \         !11<!AS:!>(!AS)\<0>
                            00  29  53  41  21  28  3E  21  3A  002A7
                                                    010E0017    002B0 P.ACP:   .LONG   17694743
                                                    00000000'   002B4          .ADDRESS P.ACQ


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

```
                                03FC 00000 DECODE_BITS:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 0245
                59 00000000G   00   9E 00002       MOVAB   STR$APPEND, R9
                58     0000'   CF   9E 00009       MOVAB   SYS_EVENTS, R8
                57 00000000G   00   9E 0000E       MOVAB   SHOW$WRITE_LINE, R7
                5E             14   C2 00015       SUBL2   #20, SP
                55         04  AC   D0 00018       MOVL    FLAGS, R5                               ; 0264
                56         08  A5   9E 0001C       MOVAB   8(R5), R6                               ; 0265
                54         04  A5   9E 00020       MOVAB   4(R5), R4                               ; 0266
                               52   D4 00024       CLRL    I                                       ; 0271
            0A                 65   E1 00026 1$:    BBC     I, (R5), 2$                             ; 0272
                           6842     DF 0002A       PUSHAL  SYS_EVENTS[I]
                       0000'   CF   9F 0002D       PUSHAB  P.ACJ
                67             02   FB 00031       CALLS   #2, SHOW$WRITE_LINE
            EE                 52   F3 00034 2$:    AOBLEQ  #2, I, 1$
                               53   D4 00038       CLRL    J                                       ; 0277
                           6344     95 0003A 3$:    TSTB    (J)[R4]                                 ; 0279
                               38   13 0003D       BEQL    6$
            0C  AE 020E0000    8F   D0 0003F       MOVL    #34471936, DESC                         ; 0287
                           10  AE   D4 00047       CLRL    DESC+4
                               52   D4 0004A       CLRL    I                                       ; 0288
            0A             6344     E1 0004C 4$:    BBC     I, (J)[R4], 5$                          ; 0290
                           3C A842  DD 00051       PUSHL   LOGINOUT_TYPES[I]                       ; 0291
                           10  AE   9F 00055       PUSHAB  DESC
                69             02   FB 00058       CALLS   #2, STR$APPEND
            ED                 52   F3 0005B 5$:    AOBLEQ  #6, I, 4$                               ; 0288
                           0C  AE   B7 0005F       DECW    DESC                                    ; 0293
            04  AE         2C A843  D0 00062       MOVL    LOGINOUT_EVENTS[J], ARGLIST             ; 0294
            08  AE         0C  AE   9E 00068       MOVAB   DESC, ARGLIST+4                         ; 0295
                            04  AE   9F 0006D       PUSHAB  ARGLIST                                 ; 0296
                       0000'   CF   9F 00070       PUSHAB  P.ACL
                67             02   FB 00074       CALLS   #2, SHOW$WRITE_LINE
            BF                 53   F3 00077 6$:    AOBLEQ  #3, J, 3$                               ; 0277
                               65   8A 0007B       BICB2   #1, (R5)                                ; 0303
                               52   D4 0007E       CLRL    I                                       ; 0304
                           6642     D5 00080 7$:    TSTL    (R6)[I]                                 ; 0306
                               4B   13 00083       BEQL    11$
                               65   E8 00085       BLBS    (R5), 8$                                ; 0314
                               6E   D4 00088       CLRL    (SP)                                    ; 0317
                               5E   DD 0008A       PUSHL   SP
                       0000'   CF   9F 0008C       PUSHAB  P.ACN
                67             02   FB 00090       CALLS   #2, SHOW$WRITE_LINE
                               65   88 00093       BISB2   #1, (R5)                                ; 0318
            0C  AE 020E0000    8F   D0 00096 8$:    MOVL    #34471936, DESC                         ; 0320
                           10  AE   D4 0009E       CLRL    DESC+4
                               53   D4 000A1       CLRL    J                                       ; 0321
                           6642     DF 000A3 9$:    PUSHAL  (R6)[I]                                 ; 0323
            0A                 9E   E1 000A6       BBC     J, @(SP)+, 10$
                           58 A843  DD 000AA       PUSHL   ACCESS_TYPES[J]                         ; 0324
                           10  AE   9F 000AE       PUSHAB  DESC
                69             02   FB 000B1       CALLS   #2, STR$APPEND
            EB                 53   F3 000B4 10$:   AOBLEQ  #4, J, 9$                               ; 0321
                           0C  AE   B7 000B8       DECW    DESC                                    ; 0326
            04  AE         0C A842  D0 000BB       MOVL    FILE_EVENTS[I], ARGLIST                 ; 0327
            08  AE         0C  AE   9E 000C1       MOVAB   DESC, ARGLIST+4                         ; 0328
                            04  AE   9F 000C6       PUSHAB  ARGLIST                                 ; 0329
                       0000'   CF   9F 000C9       PUSHAB  P.ACP
```

```
                      67          02  FB 000CD          CALLS   #2, SHOW$WRITE_LINE                      ;
                  AC  52          07  F3 000D0 11$:     AOBLEQ  #7, I, 7$                               ; 0304
                                      04 000D4          RET                                             ; 0333

; Routine Size:  213 bytes,    Routine Base:  $CODE$ + 005C
```

```
;  339     0334  1 !
;  340     0335  1 !ROUTINE get_device (desc) =
;  341     0336  1 !BEGIN
;  342     0337  1
;  343     0338  1 !+++
;  344     0339  1 !
;  345     0340  1 ! Get the name of the device to which the security journal is writing.
;  346     0341  1 !
;  347     0342  1 ! Inputs:
;  348     0343  1 !       desc - address of a descriptor, where to put the device name
;  349     0344  1 !
;  350     0345  1 ! Outputs:
;  351     0346  1 !       desc - will be filled in with the ASCII string,
;  352     0347  1 !              the name of the journal file device.
;  353     0348  1 !
;  354     0349  1 !---
;  355     0350  1
;  356     0351  1 !MAP
;  357     0352  1 !    desc : REF VECTOR;
;  358     0353  1
;  359     0354  1 !LOCAL
;  360     0355  1 !    status,
;  361     0356  1 !    chan : WORD,
;  362     0357  1 !    item_list : $ITMLST_DECL(ITEMS=2);
;  363     0358  1
;  364     0359  1 !
;  365     0360  1 ! Attempt to access the journal device.
;  366     0361  1 !
;  367     0362  1 !status = $ASSJNL(CHAN   = chan,
;  368     0363  1 !                 ACMODE = UPLIT BYTE(jsb$c_exec),
;  369     0364  1 !                 FLAGS  = cjf$m_read OR cjf$m_write,
;  370     0365  1 !                 JNLTYP = dt$_atjnl,
;  371     0366  1 !                 JNLNAM = %ASCID 'SECURITY');
;  372     0367  1
;  373     0368  1 !IF NOT .status                          ! If the assign failed,
;  374     0369  1 !THEN RETURN .status;                    ! give up now.
;  375     0370  1
;  376     0371  1 !
;  377     0372  1 ! Now, using the channel, obtain the name of the device to which
;  378     0373  1 ! the journal writes data.
;  379     0374  1 !
;  380     0375  1 !$ITMLST_INIT(ITMLST = item_list,        ! Set up an item list
;  381     0376  1 !             (ITMCOD = cji$_fildsknam,  ! asking for the device name
;  382     0377  1 !              BUFADR = .desc[1],        ! Store it here,
;  383     0378  1 !              BUFSIZ = .desc[0],
;  384     0379  1 !              RETLEN = desc[0]));       ! and return the length here.
;  385     0380  1
;  386     0381  1 !status = $GETCJI(CHAN = .chan,          ! Do it.
;  387     0382  1 !                 ITMLST = item_list);
;  388     0383  1
;  389     0384  1 !$DEASJNL(CHAN = .chan);
;  390     0385  1
;  391     0386  1 !RETURN .status;                              ! Return the final status.
;  392     0387  1 !END;
```

```
;  393          0388  1 !
;  394          0389  1
;  395          0390  1 END
;  396          0391  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  $PLIT$                        696  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                         108  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        305  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]LIB.L32;1     18619         19         0        1000          00:01.8



;                                   COMMAND QUALIFIERS

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

; Size:          305 code + 804 data bytes
; Run Time:          00:10.8
; Elapsed Time:      00:34.8
; Lines/CPU Min:      2174
; Lexemes/CPU-Min: 15519
; Memory Used:  121 pages
; Compilation Complete
```

SHOMSGUTL LIS

SHONET LIS

SHOWAUDIT LIS

SHOWID LIS

SHOWLOG LIS

SHOWERROR LIS

SHOWFILES LIS

SHOMEMORY LIS