```
   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**SETAUDIT

```
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT    AAAAAA    UU      UU  DDDDDDDD      IIIIII    TTTTTTTTTT
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT    AAAAAA    UU      UU  DDDDDDDD      IIIIII    TTTTTTTTTT
SS          EE              TT      AA    AA  UU      UU  DD      DD      II            TT
SS          EE              TT      AA    AA  UU      UU  DD      DD      II            TT
SS          EE              TT      AA    AA  UU      UU  DD      DD      II            TT
SS          EE              TT      AA    AA  UU      UU  DD      DD      II            TT
  SSSSSS    EEEEEEEE        TT      AA    AA  UU      UU  DD      DD      II            TT
  SSSSSS    EEEEEEEE        TT      AA    AA  UU      UU  DD      DD      II            TT
        SS  EE              TT      AAAAAAAAAA  UU      UU  DD      DD      II            TT
        SS  EE              TT      AAAAAAAAAA  UU      UU  DD      DD      II            TT
        SS  EE              TT      AA    AA  UU      UU  DD      DD      II            TT          ....
        SS  EE              TT      AA    AA  UU      UU  DD      DD      II            TT          ....
SSSSSSSS    EEEEEEEEEE      TT      AA    AA  UUUUUUUUUU  DDDDDDDD      IIIIII        TT          ....
SSSSSSSS    EEEEEEEEEE      TT      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 setaudit ( IDENT = 'V04-000',
;     2          0002  0                   ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL=LONG_RELATIVE)
;     3          0003  0                   ) =
;     4          0004  1 BEGIN
;     5          0005  1
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:  SET Command
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module implements the DCL command SET AUDIT.
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT:
;    38          0038  1 !
;    39          0039  1 !       VAX/VMS operating system, user and kernel mode
;    40          0040  1 !
;    41          0041  1 ! AUTHOR:  Gerry Smith                              29-Jun-1983
;    42          0042  1 !
;    43          0043  1 ! Modified by:
;    44          0044  1 !
;    45          0045  1 !       V03-007 DAS0001         David Solomon           09-Jul-1984
;    46          0046  1 !               Fix truncation errors; make nonexternal refs LONG_RELATIVE.
;    47          0047  1 !
;    48          0048  1 !       V03-006 RSH0104         R. Scott Hanna          17-Feb-1984
;    49          0049  1 !               Fix field test 1 problems, comment out journaling code
;    50          0050  1 !               and make changes due to new layout of $NSAEVTDEF.
;    51          0051  1 !
;    52          0052  1 !       V03-005 RSH0101         R. Scott Hanna          05-Feb-1984
;    53          0053  1 !               Temporarily disable SET AUDIT.
;    54          0054  1 !
;    55          0055  1 !       V03-004 MKL0208         Mary Kay Lyons          30-Nov-1983
;    56          0056  1 !               Change ITMLST in CRENUV call to LSTADR.
;    57          0057  1 !
```

```
;    58          0058  1 !        V03-003 GAS0176         Gerry Smith             30-Aug-1983
;    59          0059  1 !                Add more comments.  Also remove some of the kludges that
;    60          0060  1 !                were  necessary to make journal calls work.
;    61          0061  1 !
;    62          0062  1 !        V03-002 GAS0173         Gerry Smith             26-Aug-1983
;    63          0063  1 !                Remove reference to (now) non-existent literal.  sigh...
;    64          0064  1 !
;    65          0065  1 !        V03-001 GAS0171         Gerry Smith             24-Aug-1983
;    66          0066  1 !                Remove reference to mailbox and terminal I/O, add
;    67          0067  1 !                interactive and remote login/out.
;    68          0068  1 !
;    69          0069  1 !--
```

```
;    71          0070  1 !
;    72          0071  1 ! Include files
;    73          0072  1 !
;    74          0073  1 LIBRARY 'SYS$LIBRARY:LIB';                      ! VAX/VMS common definitions
;    75          0074  1
;    76          0075  1 !
;    77          0076  1 ! Define some flag bits.  THE FIRST TWO BIT POSITIONS MUST NOT CHANGE.
;    78          0077  1 !
;    79          0078  1 MACRO
;    80          0079  1 !    v_enable   = 0%,                           ! /ENABLE
;    81          0080  1 !    v_disable  = 1%,                           ! /DISABLE
;    82          0081  1 !    v_alarm    = 2%,                           ! /ALARM
;    83          0082  1 !    v_journal  = 3%,                           ! /JOURNAL
;    84          0083  1 !    v_log      = 4%,                           ! /LOG
;    85          0084  1 !    v_new      = 5%;                           ! /NEW_VERSION
;    86          0085  1
;    87          0086  1 LITERAL
;    88          0087  1     num_sys_events      = 3,
;    89          0088  1     num_loginout_events = 4,
;    90          0089  1     num_loginout_types  = 8,
;    91          0090  1     num_file_events     = 8,
;    92          0091  1     num_access_types    = 6,
;    93          0092  1
;    94          0093  1     arm$m_all           = arm$m_control OR
;    95          0094  1                           arm$m_delete OR
;    96          0095  1                           arm$m_execute OR
;    97          0096  1                           arm$m_read OR
;    98          0097  1                           arm$m_write,
;    99          0098  1
;   100          0099  1     nsa$m_evt_log_all   = nsa$m_evt_log_bat OR
;   101          0100  1                           nsa$m_evt_log_det OR
;   102          0101  1                           nsa$m_evt_log_dia OR
;   103          0102  1                           nsa$m_evt_log_loc OR
;   104          0103  1                           nsa$m_evt_log_net OR
;   105          0104  1                           nsa$m_evt_log_rem OR
;   106          0105  1                           nsa$m_evt_log_sub;
```

```
;  108          0106  1 !
;  109          0107  1 ! Declare some storage for tables
;  110          0108  1 !
;  111          0109  1 OWN
;  112          0110  1     option_string       : VECTOR[2]                     ! ASCII storage for
;  113          0111  1                           INITIAL(%ASCID 'ENABLE',      ! qualifiers
;  114          0112  1                                   %ASCID 'DISABLE'),
;  115          0113  1
;  116          0114  1     sys_events          : VECTOR[num_sys_events]        ! System events
;  117          0115  1                           INITIAL(%ASCID '.ACL',
;  118          0116  1                                   %ASCID '.AUTHORIZATION',
;  119          0117  1                                   %ASCID '.MOUNT'),
;  120          0118  1
;  121          0119  1     sys_bits            : VECTOR[num_sys_events]        ! System event bit masks
;  122          0120  1                           INITIAL(nsa$m_evt_acl,
;  123          0121  1                                   nsa$m_evt_uaf,
;  124          0122  1                                   nsa$m_evt_mount),
;  125          0123  1 !
;  126          0124  1 ! loginout_events must be in the same order as the corresponding bytes in $NSAEVTDEF
;  127          0125  1 !
;  128          0126  1     loginout_events     : VECTOR[num_loginout_events]   ! Loginout events
;  129          0127  1                           INITIAL(%ASCID '.BREAKIN',
;  130          0128  1                                   %ASCID '.LOGIN',
;  131          0129  1                                   %ASCID '.LOGFAILURE',
;  132          0130  1                                   %ASCID '.LOGOUT'),
;  133          0131  1
;  134          0132  1     loginout_types      : VECTOR[num_loginout_types]    ! Types of login/logout
;  135          0133  1                           INITIAL(%ASCID '.ALL',
;  136          0134  1                                   %ASCID '.DETACHED',
;  137          0135  1                                   %ASCID '.DIALUP',
;  138          0136  1                                   %ASCID '.LOCAL',
;  139          0137  1                                   %ASCID '.NETWORK',
;  140          0138  1                                   %ASCID '.REMOTE',
;  141          0139  1                                   %ASCID '.BATCH',
;  142          0140  1                                   %ASCID '.SUBPROCESS'),
;  143          0141  1
;  144          0142  1     loginout_bits       : VECTOR [num_loginout_types] ! Loginout bit masks
;  145          0143  1                           INITIAL(nsa$m_evt_log_all,
;  146          0144  1                                   nsa$m_evt_log_det,
;  147          0145  1                                   nsa$m_evt_log_dia,
;  148          0146  1                                   nsa$m_evt_log_loc,
;  149          0147  1                                   nsa$m_evt_log_net,
;  150          0148  1                                   nsa$m_evt_log_rem,
;  151          0149  1                                   nsa$m_evt_log_bat,
;  152          0150  1                                   nsa$m_evt_log_sub),
;  153          0151  1 !
;  154          0152  1 ! file_events must be in the same order as the corresponding longwords in $NSAEVTDEF
;  155          0153  1 !
;  156          0154  1     file_events         : VECTOR[num_file_events]       ! File access events
;  157          0155  1                           INITIAL(%ASCID '.FAILURE',
;  158          0156  1                                   %ASCID '.SUCCESS',
;  159          0157  1                                   %ASCID '.SYSPRV',
;  160          0158  1                                   %ASCID '.BYPASS',
;  161          0159  1                                   %ASCID '.UPGRADE',
;  162          0160  1                                   %ASCID '.DOWNGRADE',
;  163          0161  1                                   %ASCID '.GRPPRV',
;  164          0162  1                                   %ASCID '.READALL'),
```

```
;  165          0163  1
;  166          0164  1     access_types        : VECTOR[num_access_types]      ! File access types
;  167          0165  1                           INITIAL(%ASCID '.ALL',
;  168          0166  1                                   %ASCID '.CONTROL',
;  169          0167  1                                   %ASCID '.DELETE',
;  170          0168  1                                   %ASCID '.EXECUTE',
;  171          0169  1                                   %ASCID '.READ',
;  172          0170  1                                   %ASCID '.WRITE'),
;  173          0171  1
;  174          0172  1     access_bits         : VECTOR[num_access_types]      ! File access bit masks
;  175          0173  1                           INITIAL(arm$m_all,
;  176          0174  1                                   arm$m_control,
;  177          0175  1                                   arm$m_delete,
;  178          0176  1                                   arm$m_execute,
;  179          0177  1                                   arm$m_read,
;  180          0178  1                                   arm$m_write);
```

```
;  182          0179  1 !
;  183          0180  1 ! Table of contents
;  184          0181  1 !
;  185          0182  1
;  186          0183  1 FORWARD ROUTINE
;  187          0184  1     set$audit   : NOVALUE,          ! Main module of SE  AUDIT
;  188          0185  1     get_values  : NOVALUE,          ! Parse ENABLE/DISABLE list
;  189          0186  1     set_bits    : NOVALUE;          ! Set alarm/journal bits
;  190          0187  1 !   start_journal,                  ! Start a journal
;  191          0188  1 !   new_version,                    ! Make a new version of the journal file
;  192          0189  1 !   create_journal;                 ! Create a new journal
;  193          0190  1
;  194          0191  1 !
;  195          0192  1 ! External routines
;  196          0193  1 !
;  197          0194  1 EXTERNAL ROUTINE
;  198          0195  1     str$concat,                     ! Make a string
;  199          0196  1     cli$get_value,                  ! Get value from CLI
;  200          0197  1     cli$present;                    ! See if qualifier is present
;  201          0198  1
;  202          0199  1 !
;  203          0200  1 ! Declare literals defined elsewhere
;  204          0201  1 !
;  205          0202  1 !EXTERNAL LITERAL
;  206          0203  1 !   set$_jnlaccerr,                 ! Error accessing journal
;  207          0204  1 !   set$_jnlerr,                    ! Error creating journal
;  208          0205  1 !   set$_newjnl,                    ! Journal successfully created
;  209          0206  1 !   set$_jnlsysdev,                 ! Using SYS$SYSDEVICE for journal
;  210          0207  1 !   set$_vrsnerr,                   ! Error creating new version
;  211          0208  1 !   set$_newvrsn;                   ! New version created
;  212          0209  1
;  213          0210  1 !
;  214          0211  1 ! Declare the shared messages
;  215          0212  1 !
;  216          0213  1 !$SHR_MSGDEF (SET, 119, LOCAL,
;  217          0214  1 !           (syntax, error));
;  218          0215  1
;  219          0216  1 !
;  220          0217  1 ! Declare some cells in the exec
;  221          0218  1 !
;  222          0219  1 EXTERNAL
;  223          0220  1     ctl$gq_procpriv : $BBLOCK,
;  224          0221  1 !   nsa$gr_journvec,
;  225          0222  1     nsa$gr_alarmvec;
;  226          0223  1
```

```
;  228          0224  1 GLOBAL ROUTINE set$audit : NOVALUE =
;  229          0225  2 BEGIN
;  230          0226  2
;  231          0227  2 !++
;  232          0228  2 ! Functional description
;  233          0229  2 !
;  234          0230  2 !       This is the routine for the SET AUDIT command.  It is called
;  235          0231  2 !       from the SET command processor, and enables/disables security
;  236          0232  2 !       alarms and security journals, for various event classes.  In
;  237          0233  2 !       addition, a new journal file may be created.
;  238          0234  2 !
;  239          0235  2 ! Inputs
;  240          0236  2 !       None
;  241          0237  2 !
;  242          0238  2 ! Outputs
;  243          0239  2 !       None
;  244          0240  2 !
;  245          0241  2 !----
;  246          0242  2
;  247          0243  2 LOCAL
;  248          0244  2 !   status,
;  249          0245  2 !   chan : WORD,
;  250          0246  2     arglist : VECTOR[3],
;  251          0247  2     mask : VECTOR[2*nsa$k_evt_length,BYTE] ! Enable/disable flags
;  252          0248  2             VOLATILE,
;  253          0249  2 !   disk : $BBLOCK[dsc$c_s_bln],        ! Journal device name
;  254          0250  2     flags : BITVECTOR[8]                ! Flags to tell what we're doing
;  255          0251  2             INITIAL(BYTE(0));
;  256          0252  2
;  257          0253  2 !
;  258          0254  2 ! See if the user has the SECURITY privilege.
;  259          0255  2 !
;  260          0256  2 IF NOT .ctl$gq_procpriv[prv$v_security]
;  261          0257  2 THEN
;  262          0258  3     BEGIN
;  263          0259  3     SIGNAL(ss$_nosecurity);
;  264          0260  3     RETURN;
;  265          0261  2     END;
;  266          0262  2
;  267          0263  2 !
;  268          0264  2 ! See if logging is required.   Also check for  /ALARM or /JOURNAL
;  269          0265  2 !
;  270          0266  2 !IF cli$present(%ASCID 'LOG')
;  271          0267  2 !THEN flags[v_log] = 1;
;  272          0268  2
;  273          0269  2 !IF cli$present(%ASCID 'ALARM')
;  274          0270  2 !THEN flags[v_alarm] = 1;
;  275          0271  2
;  276          0272  2 !IF cli$present(%ASCID 'JOURNAL')
;  277          0273  2 !THEN flags[v_journal] = 1;
;  278          0274  2
;  279          0275  2 !IF cli$present(%ASCID 'NEW_VERSION')
;  280          0276  2 !THEN flags[v_new] = 1;
;  281          0277  2
;  282          0278  2 !$init_dyndesc(disk);
;  283          0279  2 !cli$get_value(%ASCID 'DEVICE_NAME', disk);
;  284          0280  2
```

```
;  285          0281  2
;  286          0282  2 !
;  287          0283  2 ! See if anything is to be enabled or disabled.
;  288          0284  2 !
;  289          0285  2 CH$FILL(0, %ALLOCATION(mask), mask);            ! Zero the masks
;  290          0286  2
;  291          0287  2 INCR i FROM 0 TO 1 DO                          ! Loop thru, once for enable,
;  292          0288  3     BEGIN                                      ! once for disable.
;  293          0289  3     IF cli$present(.option_string[.i])         ! If something to do,
;  294          0290  3     THEN
;  295          0291  4         BEGIN
;  296          0292  4         flags[.i] = 1;                         ! set a bit saying so, and
;  297          0293  4         get_values(.option_string[.i],         ! call routine to put new values
;  298          0294  4                    mask[.i*nsa$k_evt_length])  ! in the appropriate mask.
;  299          0295  3         END;
;  300          0296  2     END;
;  301          0297  2
;  302          0298  2 !
;  303          0299  2 ! If something is supposed to be enabled/disabled, go do it.
;  304          0300  2 !
;  305          0301  2 !IF .flags[v_enable]                           ! See why ENABLE and
;  306          0302  2 !OR .flags[v_disable]                          ! DISABLE are first in the
;  307          0303  2 !THEN                                          ! bit list?
;  308          0304  2 !   BEGIN                                      ! If something is supposed
;  309          0305  2     arglist[0] = 2;                            ! to be set/cleared, set
;  310          0306  2     arglist[1] = mask;                         ! up the argument list
;  311          0307  2     arglist[2] = flags;                        ! and call the routine
;  312          0308  2     $CMKRNL(ROUTIN = set_bits, ARGLST = arglist); ! to do the bit tweaking.
;  313          0309  2 !   END;
;  314          0310  2
;  315          0311  2 !
;  316          0312  2 ! If /JOURNAL, then start/create the journal.
;  317          0313  2 !
;  318          0314  2 !IF NOT .flags[v_journal]                      ! If not a journal,
;  319          0315  2 !THEN RETURN;                                  ! leave.
;  320          0316  2 !IF NOT (.flags[v_enable]                      ! If not either
;  321          0317  2 !       OR .flags[v_new])                      ! enabling or new_version,
;  322          0318  2 !THEN RETURN;                                  ! leave.
;  323          0319  2 !arglist[0] = 1;                                    ! Otherwise set up the
;  324          0320  2 !arglist[1] = chan;                            ! argument list and
;  325          0321  2 !status = $CMEXEC(ROUTIN = start_journal,      ! call the routine to
;  326          0322  2 !                 ARGLST = arglist);           ! start the journal.
;  327          0323  2 !
;  328          0324  2 !
;  329          0325  2 ! If there was an error trying to start the journal, tell the
;  330          0326  2 ! user and attempt to create a new one.
;  331          0327  2 !
;  332          0328  2 !IF NOT .status
;  333          0329  2 !THEN
;  334          0330  2 !   BEGIN
;  335          0331  2 !   SIGNAL(set$_jnlaccerr, 0,                  ! Error trying to start journal
;  336          0332  2 !          .status);                           ! and here's why
;  337          0333  2 !   SIGNAL(set$_jnlsysdev);                    ! and that we'll try on SYS$SYSDEVICE
;  338          0334  2 !   arglist[0] = 2;
;  339          0335  2 !   arglist[1] = chan;
;  340          0336  2 !   arglist[2] = disk;
;  341          0337  2 !   status = $CMEXEC(ROUTIN = create_journal,
```

```
;  342          0338  2 !                       ARGLST = arglist);
;  343          0339  2
;  344          0340  2 !
;  345          0341  2 ! If we can't create a journal on SYS$SYSDEVICE, give up.
;  346          0342  2 !
;  347          0343  2 !    IF NOT .status
;  348          0344  2 !    THEN SIGNAL(set$_jnlerr, 0,
;  349          0345  2 !                .status)
;  350          0346  2 !    ELSE IF .flags[v_log]
;  351          0347  2 !    THEN SIGNAL(set$_newjnl);
;  352          0348  2 !    END;
;  353          0349  2
;  354          0350  2 !
;  355          0351  2 ! If /NEW_VERSION was requested, try that.
;  356          0352  2 !
;  357          0353  2 IF .flags[v_new]
;  358          0354  2 THEN
;  359          0355  2 !    BEGIN
;  360          0356  2 !    LOCAL
;  361          0357  2 !       arglist : VECTOR[3];
;  362          0358  2 !    arglist[0] = 2;
;  363          0359  2 !    arglist[1] = chan;
;  364          0360  2 !    arglist[2] = disk;
;  365          0361  2 !    status = $CMEXEC(ROUTIN = new_version,
;  366          0362  2 !                     ARGLST = arglist);
;  367          0363  2 !    IF NOT .status
;  368          0364  2 !    THEN SIGNAL(set$_vrsnerr, 0,
;  369          0365  2 !                .status)
;  370          0366  2 !    ELSE IF .flags[v_log]
;  371          0367  2 !    THEN SIGNAL(set$_newvrsn);
;  372          0368  2 !    END;
;  373          0369  2
;  374          0370  2 RETURN;
;  375          0371  1 END;


                                                                .TITLE  SETAUDIT
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                  00  00  45  4C  42  41  4E  45  00000 P.AAB:   .ASCII  \ENABLE\<0><0>
                                                      010E0006  00008 P.AAA:   .LONG   17694726
                                                      00000000' 0000C         .ADDRESS P.AAB
                                  00  45  4C  42  41  53  49  44  00010 P.AAD:   .ASCII  \DISABLE\<0>
                                                      010E0007  00018 P.AAC:   .LONG   17694727
                                                      00000000' 0001C         .ADDRESS P.AAD
                                              4C  43  41  2E  00020 P.AAF:   .ASCII  \.ACL\
                                                      010E0004  00024 P.AAE:   .LONG   17694724
                                                      00000000' 00028         .ADDRESS P.AAF
00  4E  4F  49  54  41  5A  49  52  4F  48  54  55  41  2E  0002C P.AAH:   .ASCII  \.AUTHORIZATION\<0><0>
                                                          00  0003B
                                                      010E000E  0003C P.AAG:   .LONG   17694734
                                                      00000000' 00040         .ADDRESS P.AAH
                                  00  00  54  4E  55  4F  4D  2E  00044 P.AAJ:   .ASCII  \.MOUNT\<0><0>
                                                      010E0006  0004C P.AAI:   .LONG   17694726
                                                      00000000' 00050         .ADDRESS P.AAJ
```

```
                                4E 49 4B 41 45 52 42 2E  00054 P.AAL:   .ASCII  \.BREAKIN\
                                                010E0008 0005C P.AAK:   .LONG   17694728
                                                00000000' 00060         .ADDRESS P.AAL
                                00 00 4E 49 47 4F 4C 2E  00064 P.AAN:   .ASCII  \.LOGIN\<0><0>
                                                010E0006 0006C P.AAM:   .LONG   17694726
                                                00000000' 00070         .ADDRESS P.AAN
                00 45 52 55 4C 49 41 46 47 4F 4C 2E  00074 P.AAP:   .ASCII  \.LOGFAILURE\<0>
                                                010E000B 00080 P.AAO:   .LONG   17694731
                                                00000000' 00084         .ADDRESS P.AAP
                                00 54 55 4F 47 4F 4C 2E  00088 P.AAR:   .ASCII  \.LOGOUT\<0>
                                                010E0007 00090 P.AAQ:   .LONG   17694727
                                                00000000' 00094         .ADDRESS P.AAR
                                                4C 4C 41 2E  00098 P.AAT:   .ASCII  \.ALL\
                                                010E0004 0009C P.AAS:   .LONG   17694724
                                                00000000' 000A0         .ADDRESS P.AAT
                00 00 00 44 45 48 43 41 54 45 44 2E  000A4 P.AAV:   .ASCII  \.DETACHED\<0><0><0>
                                                010E0009 000B0 P.AAU:   .LONG   17694729
                                                00000000' 000B4         .ADDRESS P.AAV
                                00 50 55 4C 41 49 44 2E  000B8 P.AAX:   .ASCII  \.DIALUP\<0>
                                                010E0007 000C0 P.AAW:   .LONG   17694727
                                                00000000' 000C4         .ADDRESS P.AAX
                                00 00 4C 41 43 4F 4C 2E  000C8 P.AAZ:   .ASCII  \.LOCAL\<0><0>
                                                010E0006 000D0 P.AAY:   .LONG   17694726
                                                00000000' 000D4         .ADDRESS P.AAZ
                                4B 52 4F 57 54 45 4E 2E  000D8 P.ABB:   .ASCII  \.NETWORK\
                                                010E0008 000E0 P.ABA:   .LONG   17694728
                                                00000000' 000E4         .ADDRESS P.ABB
                                00 45 54 4F 4D 45 52 2E  000E8 P.ABD:   .ASCII  \.REMOTE\<0>
                                                010E0007 000F0 P.ABC:   .LONG   17694727
                                                00000000' 000F4         .ADDRESS P.ABD
                                00 00 48 43 54 41 42 2E  000F8 P.ABF:   .ASCII  \.BATCH\<0><0>
                                                010E0006 00100 P.ABE:   .LONG   17694726
                                                00000000' 00104         .ADDRESS P.ABF
        00 53 53 45 43 4F 52 50 42 55 53 2E  00108 P.ABH:   .ASCII  \.SUBPROCESS\<0>
                                                010E000B 00114 P.ABG:   .LONG   17694731
                                                00000000' 00118         .ADDRESS P.ABH
                                45 52 55 4C 49 41 46 2E  0011C P.ABJ:   .ASCII  \.FAILURE\
                                                010E0008 00124 P.ABI:   .LONG   17694728
                                                00000000' 00128         .ADDRESS P.ABJ
                                53 53 45 43 43 55 53 2E  0012C P.ABL:   .ASCII  \.SUCCESS\
                                                010E0008 00134 P.ABK:   .LONG   17694728
                                                00000000' 00138         .ADDRESS P.ABL
                                00 56 52 50 53 59 53 2E  0013C P.ABN:   .ASCII  \.SYSPRV\<0>
                                                010E0007 00144 P.ABM:   .LONG   17694727
                                                00000000' 00148         .ADDRESS P.ABN
                                00 53 53 41 50 59 42 2E  0014C P.ABP:   .ASCII  \.BYPASS\<0>
                                                010E0007 00154 P.ABO:   .LONG   17694727
                                                00000000' 00158         .ADDRESS P.ABP
                                45 44 41 52 47 50 55 2E  0015C P.ABR:   .ASCII  \.UPGRADE\
                                                010E0008 00164 P.ABQ:   .LONG   17694728
                                                00000000' 00168         .ADDRESS P.ABR
        00 00 45 44 41 52 47 4E 57 4F 44 2E  0016C P.ABT:   .ASCII  \.DOWNGRADE\<0><0>
                                                010E000A 00178 P.ABS:   .LONG   17694730
                                                00000000' 0017C         .ADDRESS P.ABT
                                00 56 52 50 50 52 47 2E  00180 P.ABV:   .ASCII  \.GRPPRV\<0>
                                                010E0007 00188 P.ABU:   .LONG   17694727
                                                00000000' 0018C         .ADDRESS P.ABV
```

```
                        4C  4C  41  44  41  45  52  2E  00190 P.ABX:   .ASCII  \.READALL\                                    ;
                                                010E0008  00198 P.ABW:   .LONG   17694728                                    ;
                                                00000000' 0019C          .ADDRESS P.ABX                                     ;
                                            4C  4C  41  2E  001A0 P.ABZ:   .ASCII  \.ALL\                                     ;
                                                010E0004  001A4 P.ABY:   .LONG   17694724                                    ;
                                                00000000' 001A8          .ADDRESS P.ABZ                                     ;
                        4C  4F  52  54  4E  4F  43  2E  001AC P.ACB:   .ASCII  \.CONTROL\                                   ;
                                                010E0008  001B4 P.ACA:   .LONG   17694728                                    ;
                                                00000000' 001B8          .ADDRESS P.ACB                                     ;
                        00  45  54  45  4C  45  44  2E  001BC P.ACD:   .ASCII  \.DELETE\<0>                                  ;
                                                010E0007  001C4 P.ACC:   .LONG   17694727                                    ;
                                                00000000' 001C8          .ADDRESS P.ACD                                     ;
                        45  54  55  43  45  58  45  2E  001CC P.ACF:   .ASCII  \.EXECUTE\                                   ;
                                                010E0008  001D4 P.ACE:   .LONG   17694728                                    ;
                                                00000000' 001D8          .ADDRESS P.ACF                                     ;
                        00  00  00  44  41  45  52  2E  001DC P.ACH:   .ASCII  \.READ\<0><0><0>                              ;
                                                010E0005  001E4 P.ACG:   .LONG   17694725                                    ;
                                                00000000' 001E8          .ADDRESS P.ACH                                     ;
                        00  00  45  54  49  52  57  2E  001EC P.ACJ:   .ASCII  \.WRITE\<0><0>                               ;
                                                010E0006  001F4 P.ACI:   .LONG   17694726                                    ;
                                                00000000' 001F8          .ADDRESS P.ACJ                                     ;

                                                                         .PSECT  $OWN$,NOEXE,2

                                    00000000' 00000000' 00000 OPTION_STRING:
                                                                         .ADDRESS P.AAA, P.AAC                              ;
                                00000000' 00000000' 00000000' 00008 SYS_EVENTS:
                                                                         .ADDRESS P.AAE, P.AAG, P.AAI                       ;
                                00000002  00000004  00000001  00014 SYS_BITS:
                                                                         .LONG   1, 4, 2                                    ;
                        00000000' 00000000' 00000000' 00000000' 00020 LOGINOUT_EVENTS:
                                                                         .ADDRESS P.AAK, P.AAM, P.AAO, P.AAQ                ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00030 LOGINOUT_TYPES:
                                                                         .ADDRESS P.AAS, P.AAU, P.AAW, P.AAY, P.ABA, -      ;
                                    00000000' 00000000' 00048                P.ABC, P.ABE, P.ABG                        ;
00000008  00000010  00000004  00000002  00000040  0000007F  00050 LOGINOUT_BITS:
                                                                         .LONG   127, 64, 2, 4, 16, 8, 1, 32                ;
                                    00000020  00000001  00068                                                               ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00070 FILE_EVENTS:
                                                                         .ADDRESS P.ABI, P.ABK, P.ABM, P.ABO, P.ABQ, -      ;
                                    00000000' 00000000' 00088                P.ABS, P.ABU, P.ABW                        ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00090 ACCESS_TYPES:
                                                                         .ADDRESS P.ABY, P.ACA, P.ACC, P.ACE, P.ACG, P.ACI  ;
00000002  00000001  00000004  00000008  00000010  0000001F  000A8 ACCESS_BITS:
                                                                         .LONG   31, 16, 8, 4, 1, 2                         ;

                                                                         .EXTRN  STR$CONCAT, CLI$GET_VALUE
                                                                         .EXTRN  CLI$PRESENT, CTL$GQ_PROCPRIV
                                                                         .EXTRN  NSA$GR_ALARMVEC
                                                                         .EXTRN  SYS$CMKRNL

                                                                         .PSECT  $CODE$,NOWRT,2

                                                007C 00000      SET$AUDIT:  .ENTRY  SET$AUDIT, Save R2,R3,R4,R5,R6         ; 0224
                                    56 00000000'  EF  9E 00002              MOVAB   OPTION_STRING, R6                      ;
                                    5E        A0  AE  9E 00009              MOVAB   -96(SP), SP                            ;
                                                  6E  94 0000D              CLRB    FLAGS                                  ; 0225
```

```
                     0D 00000000G   00          06 E0 0000F          BBS     #6, CTL$GQ_PROCPRIV+4, 1$                ; 0256
                        7E     2934   8F 3C 00017          MOVZWL  #10548, -(SP)                    ; 0259
                        00000000G   00          01 FB 0001C          CALLS   #1, LIB$SIGNAL
                                                04 00023          RET                              ; 0258
    0050    8F          00          6E          00 2C 00024 1$:  MOVC5   #0, (SP), #0, #80, MASK          ; 0285
                                          04    AE    0002B
                                                52 D4 0002D          CLRL    I                        ; 0287
                                             6642 DD 0002F 2$:   PUSHL   OPTION_STRING[I]         ; 0289
                        00000000G   00          01 FB 00032          CALLS   #1, CLI$PRESENT
                                    16          50 E9 00039          BLBC    R0, 4$
                     00             6E          52 E2 0003C          BBSS    I, FLAGS, 3$             ; 0292
                     50             52          28 C5 00040 3$:  MULL3   #40, I, R0               ; 0294
                                          04 AE40 9F 00044          PUSHAB  MASK[R0]
                                             6642 DD 00048          PUSHL   OPTION_STRING[I]
                        00000000V   EF          02 FB 0004B          CALLS   #2, GET_VALUES
                     D9             52          01 F3 00052 4$:  AOBLEQ  #1, I, 2$                ; 0287
                              54    AE          02 D0 00056          MOVL    #2, ARGLIST              ; 0305
                              58    AE    04    AE 9E 0005A          MOVAB   MASK, ARGLIST+4          ; 0306
                              5C    AE          6E 9E 0005F          MOVAB   FLAGS, ARGLIST+8         ; 0307
                                          54    AE 9F 00063          PUSHAB  ARGLIST                  ; 0308
                                    00000000V   EF 9F 00066          PUSHAB  SET_BITS
                        00000000G   00          02 FB 0006C          CALLS   #2, SYS$CMKRNL
                                                04 00073          RET                              ; 0371

; Routine Size:  116 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  377     0372  1 ROUTINE get_values (option, mask) : NOVALUE =
;  378     0373  2 BEGIN
;  379     0374  2
;  380     0375  2 !+++
;  381     0376  2 !
;  382     0377  2 ! Given an ASCII descriptor and a set of masks, decipher the
;  383     0378  2 ! string into an audit class, and set the appropriate bits in the
;  384     0379  2 ! masks
;  385     0380  2 !
;  386     0381  2 ! Inputs:
;  387     0382  2 !       mask - address of mask area
;  388     0383  2 !       option - address of ASCII descriptor
;  389     0384  2 !
;  390     0385  2 ! Outputs:
;  391     0386  2 !       mask - will have a bit set.
;  392     0387  2 !
;  393     0388  2 !---
;  394     0389  2
;  395     0390  2 MAP
;  396     0391  2     mask : REF $BBLOCK;                         ! The mask is a byte block
;  397     0392  2
;  398     0393  2 BIND                                            ! Locate stuff within the mask:
;  399     0394  2     sys = mask[nsa$l_evt_sys],                  ! The simple system bits,
;  400     0395  2     loginout = mask[nsa$b_evt_logb] : VECTOR[4,BYTE],   ! Loginout bits
;  401     0396  2     file_access = mask[nsa$l_evt_failure] : VECTOR;     ! File access vector
;  402     0397  2
;  403     0398  2 LOCAL
;  404     0399  2     all         : BYTE INITIAL (0),
;  405     0400  2     file_all    : BYTE INITIAL (0),
;  406     0401  2     string      : $BBLOCK[dsc$c_s_bln],
;  407     0402  2     desc        : $BBLOCK[dsc$c_s_bln];
;  408     0403  2
;  409     0404  2 $init_dyndesc(desc);                            ! Get a couple of
;  410     0405  2 $init_dyndesc(string);                          ! dynamic descriptors
;  411     0406  2
;  412     0407  2 str$concat(desc, .option, %ASCID '.ALL');
;  413     0408  2 IF cli$present(desc) THEN all = 1;
;  414     0409  2
;  415     0410  2 !
;  416     0411  2 ! system events
;  417     0412  2 !
;  418     0413  2
;  419     0414  2 INCR i FROM 0 TO num_sys_events-1 DO
;  420     0415  3     BEGIN
;  421     0416  3     str$concat(desc, .option, .sys_events[.i]);
;  422     0417  3     IF .all OR cli$present(desc) THEN sys = .sys OR .sys_bits[.i];
;  423     0418  2     END;
;  424     0419  2
;  425     0420  2 !
;  426     0421  2 ! Loginout events
;  427     0422  2 !
;  428     0423  2
;  429     0424  2 INCR i FROM 0 TO num_loginout_events-1 DO
;  430     0425  3     BEGIN
;  431     0426  3     str$concat(desc, .option, .loginout_events[.i]);
;  432     0427  3     IF .all OR cli$present(desc)
;  433     0428  3     THEN
```

```
;  434          0429  4          BEGIN
;  435          0430  4          INCR j FROM 0 TO num_loginout_types-1 DO
;  436          0431  5              BEGIN
;  437          0432  5              IF (.i EQLU 0) AND (.j EQLU 6) THEN EXITLOOP;
;  438          0433  5              str$concat(desc, .option, .loginout_events[.i], .loginout_types[.j]);
;  439          0434  5              IF .all OR cli$present(desc)
;  440          0435  5              THEN
;  441          0436  5                  loginout[.i] = .loginout[.i] OR .loginout_bits[.j];
;  442          0437  4              END;
;  443          0438  3          END;
;  444          0439  2      END;
;  445          0440  2  loginout[0] = .loginout[0] AND NOT (nsa$m_evt_log_bat OR nsa$m_evt_log_sub);
;  446          0441  2
;  447          0442  2  !
;  448          0443  2  ! file access events
;  449          0444  2  !
;  450          0445  2
;  451          0446  2  str$concat(desc, .option, %ASCID '.FILE_ACCESS', %ASCID '.ALL');
;  452          0447  2  IF cli$present(desc) THEN file_all = 1;
;  453          0448  2
;  454          0449  2  INCR i FROM 0 TO num_file_events-1 DO
;  455          0450  3      BEGIN
;  456          0451  3      str$concat(desc, .option, %ASCID '.FILE_ACCESS', .file_events[.i]);
;  457          0452  3      IF .all OR .file_all OR cli$present(desc)
;  458          0453  3      THEN
;  459          0454  4          BEGIN
;  460          0455  4          INCR j FROM 0 TO num_access_types-1 DO
;  461          0456  5              BEGIN
;  462          0457  5              str$concat(desc, .option, %ASCID '.FILE_ACCESS', .file_events[.i], .access_types[.j]);
;  463          0458  5              IF .all OR .file_all OR cli$present(desc)
;  464          0459  5              THEN
;  465          0460  5                  file_access[.i] = .file_access[.i] OR .access_bits[.j];
;  466          0461  4              END;
;  467          0462  4          IF .file_access[.i] EQLU 0 THEN file_access[.i] = .access_bits[0];
;  468          0463  3          END;
;  469          0464  2      END;
;  470          0465  2  !
;  471          0466  2  ! Upgrade and Downgrade not implemented yet so make sure that ALL has not
;  472          0467  2  ! turned them on.
;  473          0468  2  !
;  474          0469  2  file_access[4] = 0;
;  475          0470  2  file_access[5] = 0;
;  476          0471  1  END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  4C  4C  41  2E  001FC P.ACL:   .ASCII  \.ALL\                          ;
                                                                  00200 P.ACK:   .LONG   17694724                        ;
                                                                  00204          .ADDRESS P.ACL                          ;
                53  53  45  43  43  41  5F  45  4C  49  46  2E  00208 P.ACN:   .ASCII  \.FILE_ACCESS\                   ;
                                                                  00214 P.ACM:   .LONG   17694732                        ;
                                                                  00218          .ADDRESS P.ACN                          ;
                                                  4C  4C  41  2E  0021C P.ACP:   .ASCII  \.ALL\                          ;
                                                                  00220 P.ACO:   .LONG   17694724                        ;
                                                                  00224          .ADDRESS P.ACP                          ;
```

```
53 53 45 43 43 41 5F 45 4C 49 46 2E  00228 P.ACR:   .ASCII   \.FILE_ACCESS\                          ;
                           010E000C  00234 P.ACQ:   .LONG    17694732                                ;
                          00000000'  00238          .ADDRESS P.ACR                                   ;
53 53 45 43 43 41 5F 45 4C 49 46 2E  0023C P.ACT:   .ASCII   \.FILE_ACCESS\                          ;
                           010E000C  00248 P.ACS:   .LONG    17694732                                ;
                          00000000'  0024C          .ADDRESS P.ACT                                   ;


                                                    .PSECT   $CODE$,NOWRT,2

                               0FFC  00000 GET_VALUES:
                                                    .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11   ; 0372
           5B 00000000G   00    9E  00002          MOVAB    CLI$PRESENT, R11                        ;
           5A 00000000G   00    9E  00009          MOVAB    STR$CONCAT, R10                         ;
           59 00000000'   EF    9E  00010          MOVAB    LOGINOUT_EVENTS, R9                     ;
           5E             0C    C2  00017          SUBL2    #12, SP                                 ;
   54   08 AC             04    C1  0001A          ADDL3    #4, MASK, R4                            ; 0395
   55   08 AC             08    C1  0001F          ADDL3    #8, MASK, R5                            ; 0396
                          58    94  00024          CLRB     ALL                                     ;
                          57    94  00026          CLRB     FILE_ALL                                ;
              020E0000    8F    DD  00028          PUSHL    #34471936                               ; 0404
                    04    AE    D4  0002E          CLRL     DESC+4                                  ;
        08 AE 020E0000    8F    D0  00031          MOVL     #34471936, STRING                       ; 0405
                    0C    AE    D4  00039          CLRL     STRING+4                                ;
             00000000'    EF    9F  0003C          PUSHAB   P.ACK                                   ; 0407
           56       04    AC    D0  00042          MOVL     OPTION, R6                              ;
                          56    DD  00046          PUSHL    R6                                      ;
                    08    AE    9F  00048          PUSHAB   DESC                                    ;
           6A             03    FB  0004B          CALLS    #3, STR$CONCAT                          ;
                          5E    DD  0004E          PUSHL    SP                                      ; 0408
           6B             01    FB  00050          CALLS    #1, CLI$PRESENT                         ;
           03             50    E9  00053          BLBC     R0, 1$                                  ;
           58             01    90  00056          MOVB     #1, ALL                                 ;
                          52    D4  00059 1$:      CLRL     I                                       ; 0417
                  E8 A942       DD  0005B 2$:      PUSHL    SYS_EVENTS[I]                           ; 0416
                          56    DD  0005F          PUSHL    R6                                      ;
                    08    AE    9F  00061          PUSHAB   DESC                                    ;
           6A             03    FB  00064          CALLS    #3, STR$CONCAT                          ;
           08             58    E8  00067          BLBS     ALL, 3$                                 ; 0417
                          5E    DD  0006A          PUSHL    SP                                      ;
           6B             01    FB  0006C          CALLS    #1, CLI$PRESENT                         ;
           06             50    E9  0006F          BLBC     R0, 4$                                  ;
        08 BC        F4 A942    C8  00072 3$:      BISL2    SYS_BITS[I], @MASK                      ;
   DF      52             02    F3  00078 4$:      AOBLEQ   #2, I, 2$                               ; 0414
                          53    D4  0007C          CLRL     I                                       ; 0424
                        6943    DD  0007E 5$:      PUSHL    LOGINOUT_EVENTS[I]                      ; 0426
                          56    DD  00081          PUSHL    R6                                      ;
                    08    AE    9F  00083          PUSHAB   DESC                                    ;
           6A             03    FB  00086          CALLS    #3, STR$CONCAT                          ;
           08             58    E8  00089          BLBS     ALL, 6$                                 ; 0427
                          5E    DD  0008C          PUSHL    SP                                      ;
           6B             01    FB  0008E          CALLS    #1, CLI$PRESENT                         ;
           31             50    E9  00091          BLBC     R0, 11$                                 ;
                          52    D4  00094 6$:      CLRL     J                                       ; 0430
                          53    D5  00096 7$:      TSTL     I                                       ; 0432
                          05    12  00098          BNEQ     8$                                      ;
```

```
                06              52  D1 0009A          CMPL    J, #6                                   ;
                                26  13 0009D          BEQL    11$                                     ;
                            10 A942  DD 0009F  8$:     PUSHL   LOGINOUT_TYPES[J]                       ; 0433
                               6943  DD 000A3          PUSHL   LOGINOUT_EVENTS[I]                      ;
                                  56  DD 000A6          PUSHL   R6                                      ;
                                0C    AE 9F 000A8       PUSHAB  DESC                                    ;
                              6A        04  FB 000AB    CALLS   #4, STR$CONCAT                          ;
                              08        58  E8 000AE    BLBS    ALL, 9$                                 ; 0434
                                        5E  DD 000B1    PUSHL   SP                                      ;
                              6B        01  FB 000B3    CALLS   #1, CLI$PRESENT                         ;
                              08        50  E9 000B6    BLBC    R0, 10$                                 ;
                            30 A942  DF 000B9  9$:     PUSHAL  LOGINOUT_BITS[J]                        ; 0436
                          6344        9E  88 000BD    BISB2   @(SP)+, (I)[R4]                         ;
                D1            52        07  F3 000C1  10$:    AOBLEQ  #7, J, 7$                       ; 0430
                B5            53        03  F3 000C5  11$:    AOBLEQ  #3, I, 5$                       ; 0424
                              64        21  8A 000C9    BICB2   #33, (R4)                               ; 0440
                    00000000'   EF 9F 000CC       PUSHAB  P.ACO                                   ; 0446
                    00000000'   EF 9F 000D2       PUSHAB  P.ACM                                   ;
                                  56  DD 000D8    PUSHL   R6                                      ;
                                0C    AE 9F 000DA       PUSHAB  DESC                                    ;
                              6A        04  FB 000DD    CALLS   #4, STR$CONCAT                          ;
                                        5E  DD 000E0    PUSHL   SP                                      ; 0447
                              6B        01  FB 000E2    CALLS   #1, CLI$PRESENT                         ;
                              03        50  E9 000E5    BLBC    R0, 12$                                 ;
                              57        01  90 000E8    MOVB    #1, FILE_ALL                            ;
                                        52  D4 000EB  12$:     CLRL    I                               ; 0449
                            50 A942  DD 000ED  13$:     PUSHL   FILE_EVENTS[I]                          ; 0451
                    00000000'   EF 9F 000F1       PUSHAB  P.ACQ                                   ;
                                  56  DD 000F7    PUSHL   R6                                      ;
                                0C    AE 9F 000F9       PUSHAB  DESC                                    ;
                              6A        04  FB 000FC    CALLS   #4, STR$CONCAT                          ;
                              0B        58  E8 000FF    BLBS    ALL, 14$                                ; 0452
                              08        57  E8 00102    BLBS    FILE_ALL, 14$                           ;
                                        5E  DD 00105    PUSHL   SP                                      ;
                              6B        01  FB 00107    CALLS   #1, CLI$PRESENT                         ;
                              3C        50  E9 0010A    BLBC    R0, 18$                                 ;
                                        53  D4 0010D  14$:     CLRL    J                               ; 0455
                            70 A943  DD 0010F  15$:     PUSHL   ACCESS_TYPES[J]                         ; 0457
                            50 A942  DD 00113          PUSHL   FILE_EVENTS[I]                          ;
                    00000000'   EF 9F 00117       PUSHAB  P.ACS                                   ;
                                  56  DD 0011D    PUSHL   R6                                      ;
                                10    AE 9F 0011F       PUSHAB  DESC                                    ;
                              6A        05  FB 00122    CALLS   #5, STR$CONCAT                          ;
                              0B        58  E8 00125    BLBS    ALL, 16$                                ; 0458
                              08        57  E8 00128    BLBS    FILE_ALL, 16$                           ;
                                        5E  DD 0012B    PUSHL   SP                                      ;
                              6B        01  FB 0012D    CALLS   #1, CLI$PRESENT                         ;
                              07        50  E9 00130    BLBC    R0, 17$                                 ;
                            6542     0088 C943  C8 00133  16$:    BISL2   ACCESS_BITS[J], (R5)[I]         ; 0460
                D1            03        05  F3 0013A  17$:    AOBLEQ  #5, J, 15$                      ; 0455
                                      6542  D5 0013E    TSTL    (R5)[I]                                 ; 0462
                                        06  12 00141    BNEQ    18$                                     ;
                            6542     0088   C9 00143    MOVL    ACCESS_BITS, (R5)[I]                    ;
                A0            52        07  F3 00149  18$:    AOBLEQ  #7, I, 13$                      ; 0449
                                10    A5  7C 0014D       CLRQ    16(R5)                                  ; 0469
                                          04 00150    RET                                             ; 0471
```

; Routine Size: 337 bytes,   Routine Base: $CODE$ + 0074

```
1 ROUTINE set_bits (mask, flags) : NOVALUE =
2 BEGIN
2
2 !+++
2 !
2 ! Reset the bits for either security alarms, or security journaling.
2 !
2 ! Inputs:
2 !       mask - address of vector of bits to enable and disable
2 !       flags - address of flags (alarm, journal)
2 !
2 ! Outputs:
2 !       None.  The auditing bits are set.
2 !
2 !---
2
2 MAP
2     mask : REF VECTOR[,BYTE],
2     flags : REF BITVECTOR;
2
2 BIND
2     enab = mask[0] : VECTOR[,BYTE],
2     disab = mask[nsa$k_evt_length] : VECTOR[,BYTE];
2
2 LOCAL
2     bits : REF VECTOR[,BYTE];
2
2 !
2 ! Determine whether to modify the journal or alarm bits.
2 !
2 !IF .flags[v_alarm]
2 !THEN bits = nsa$gr_alarmvec
2 !ELSE bits = nsa$gr_journvec;
2 bits = nsa$gr_alarmvec;
2
2 !
2 ! Step through the vector in SYSCOMMON and enable bits.
2 !
2 INCR i FROM 0 TO nsa$k_evt_length-1 DO
2     bits[.i] = (.bits[.i] OR .enab[.i]) AND NOT .disab[.i];
2
2 RETURN;
1 END;
```

```
                                  001C 00000 SET_BITS:
                                                     .WORD   Save R2,R3,R4            ; 0472
 53       04     AC              28  C1 00002        ADDL3   #40, MASK, R3            ; 0494
                 51 00000000G    00  9E 00007        MOVAB   NSA$GR_ALARMVEC, BITS    ; 0505
                                 50  D4 0000E        CLRL    I                        ; 0511
                 52            6041  9A 00010 1$:    MOVZBL  (I)[BITS], R2            ;
                 54         04 BC40  9A 00014        MOVZBL  @MASK[I], R4             ;
                 52              54  C8 00019        BISL2   R4, R2                   ;
6041             52            6043  8B 0001C        BICB3   (I)[R3], R2, (I)[BITS]   ;
```

```
                    EA          50          27  F3 00022         AOBLEQ  #39, I, 1$                ;
                                                04 00026         RET                                ; 0514

; Routine Size:  39 bytes,    Routine Base:  $CODE$ + 01C5
```

```
;  521          0515  1 !
;  522          0516  1 !ROUTINE start_journal (chan) =
;  523          0517  1 !BEGIN
;  524          0518  1
;  525          0519  1 !+++
;  526          0520  1 !
;  527          0521  1 ! Attempt to assign a channel to the journal.
;  528          0522  1 !
;  529          0523  1 ! Inputs:
;  530          0524  1 !       chan - address of word to store channel number
;  531          0525  1 !
;  532          0526  1 ! Outputs:
;  533          0527  1 !       chan - will be filled in.
;  534          0528  1 !
;  535          0529  1 !---
;  536          0530  1
;  537          0531  1 !MAP
;  538          0532  1 !    chan : REF VECTOR[,WORD];
;  539          0533  1
;  540          0534  1 !RETURN $ASSJNL(CHAN   = chan[0],                ! Try to assign a channel,
;  541          0535  1 !               JNLTYP = dt$_atjnl,              ! perhaps causing the journal
;  542          0536  1 !               ACMODE = UPLIT BYTE(jsb$c_exec), ! to get started on the
;  543          0537  1 !               JNLNAM = %ASCID 'SECURITY');     ! system.
;  544          0538  1
;  545          0539  1 !END;
```

```
;  546          0540  1 !
;  547          0541  1 !ROUTINE create_journal (chan, disk) =
;  548          0542  1 !BEGIN
;  549          0543  1
;  550          0544  1 !+++
;  551          0545  1 !
;  552          0546  1 ! Try to create a journal.
;  553          0547  1 !
;  554          0548  1 ! Inputs:
;  555          0549  1 !       chan - address of word, where to put channel number
;  556          0550  1 !       disk - address of descriptor for device to journal to
;  557          0551  1 !
;  558          0552  1 ! Outputs:
;  559          0553  1 !       chan - will be filled in.
;  560          0554  1 !
;  561          0555  1 !---
;  562          0556  1
;  563          0557  1 !MAP
;  564          0558  1 !    chan : REF VECTOR[,WORD],
;  565          0559  1 !    disk : REF $BBLOCK;
;  566          0560  1
;  567          0561  1 !LOCAL
;  568          0562  1 !    device_list : VECTOR[3],
;  569          0563  1 !    data : $BBLOCK[jsb$c_length];
;  570          0564  1
;  571          0565  1 !
;  572          0566  1 ! Set up the list of descriptors pointing to the device(s) on which the
;  573          0567  1 ! journal file resides.
;  574          0568  1 !
;  575          0569  1 !IF .disk[dsc$w_length] NEQ 0                    ! If there's a valid
;  576          0570  1 !THEN                                            ! device specified by the
;  577          0571  1 !    BEGIN                                       ! user, transfer the length
;  578          0572  1 !    device_list[0] = .disk[dsc$w_length];       ! and address to DEVICE_LIST.
;  579          0573  1 !    device_list[1] = .disk[dsc$a_pointer];
;  580          0574  1 !    END
;  581          0575  1 !ELSE                                                ! Otherwise, use
;  582          0576  1 !    BEGIN                                           ! SYS$SYSDEVICE.
;  583          0577  1 !    device_list[0] = %CHARCOUNT('SYS$SYSDEVICE');
;  584          0578  1 !    device_list[1] = UPLIT BYTE ('SYS$SYSDEVICE');
;  585          0579  1 !    END;
;  586          0580  1 !device_list[2] = 0;                              ! Zero-terminate the list
;  587          0581  1
;  588          0582  1
;  589          0583  1 !
;  590          0584  1 ! Set up the Journal Specification Block, which contains all the
;  591          0585  1 ! information about the journal to be opened/created.
;  592          0586  1 !
;  593          0587  1 !CH$FILL(0, jsb$c_length, data);                        ! Initialize
;  594          0588  1 !data[jsb$w_jnlnamlen] = %CHARCOUNT('SECURITY');        ! Journal name is 8 bytes,
;  595          0589  1 !data[jsb$l_jnlnam] = UPLIT BYTE('SECURITY');    ! name is "SECURITY"
;  596          0590  1 !data[jsb$b_jnltyp] = jsb$c_at;                  ! AT journal
;  597          0591  1 !data[jsb$b_jnldev] = jsb$c_disk;                ! always a disk journal
;  598          0592  1 !data[jsb$w_maxsiz] = nsa$s_idt_record_buf;      ! and can be this big
;  599          0593  1 !data[jsb$w_filext] = 10;                        ! extend size
;  600          0594  1 !data[jsb$w_bufsiz] = 1;                                 ! (Do this to stop crashing)
;  601          0595  1 !data[jsb$b_acmode] = jsb$c_exec;                ! do everything from EXEC
;  602          0596  1 !data[jsb$w_prot] = %B'1101T1011T001T00';        ! Protection=(S:RW,O:RW,G:W,W:W)
```

```
;  603          0597  1 !data[jsb$l_uic] = %X'00010004';                    ! Owner = [1,4]
;  604          0598  1 !data[jsb$l_flags] = jsb$m_known                     ! Known journal, and create a
;  605          0599  1 !                    OR jsb$m_cif;                  ! journal file if not there
;  606          0600  1 !data[jsb$b_copies] = 1;                            ! Only one copy of the file
;  607          0601  1 !data[jsb$l_prinamdes] = device_list;               ! Disk names here
;  608          0602  1
;  609          0603  1 !RETURN $CREJNL(CHAN   = chan[0],                   ! Give it a try.
;  610          0604  1 !               FLAGS  = cjf$m_read,
;  611          0605  1 !               ACMODE = UPLIT BYTE(jsb$c_exec),
;  612          0606  1 !               JSB    = data);
;  613          0607  1
;  614          0608  1 !END;
```

```
;  615     0609  1 !
;  616     0610  1 !ROUTINE new_version (chan, disk) =
;  617     0611  1 !BEGIN
;  618     0612  1
;  619     0613  1 !+++
;  620     0614  1 !
;  621     0615  1 ! Create a new version of the journal file.
;  622     0616  1 !
;  623     0617  1 ! Inputs:
;  624     0618  1 !       chan - address of channel to use
;  625     0619  1 !       disk - address of device descriptor, telling what device to
;  626     0620  1 !              locate the new file.
;  627     0621  1 !
;  628     0622  1 ! Outputs:
;  629     0623  1 !       None.
;  630     0624  1 !
;  631     0625  1 !---
;  632     0626  1
;  633     0627  1 !MAP
;  634     0628  1 !    chan : REF VECTOR[,WORD],
;  635     0629  1 !    disk : REF $BBLOCK;
;  636     0630  1
;  637     0631  1 !LOCAL
;  638     0632  1 !    item_list : VECTOR[12],
;  639     0633  1 !    curdev_len,
;  640     0634  1 !    curdev : VECTOR[10];
;  641     0635  1
;  642     0636  1 !
;  643     0637  1 ! Get the current device name.
;  644     0638  1 !
;  645     0639  1 !BEGIN
;  646     0640  1 !LOCAL
;  647     0641  1 !    status,                                    ! Declare these here,
;  648     0642  1 !    list : $ITMLST_DECL(ITEMS = 1);            ! because they're temporary.
;  649     0643  1
;  650     0644  1 !$ITMLST_INIT(ITMLST = list,                    ! Set up this list,
;  651     0645  1 !             (ITMCOD = cji$_fildsknam,         ! want device name
;  652     0646  1 !              BUFADR = curdev,                 ! store it here,
;  653     0647  1 !              BUFSIZ = %ALLOCATION(curdev),    ! up to this big,
;  654     0648  1 !              RETLEN = curdev_len)             ! actual size returned here.
;  655     0649  1 !             );
;  656     0650  1 !IF NOT (status = $GETCJI(ITMLST = list,                 ! Do it.
;  657     0651  1 !                         CHAN   = .chan[0]))    ! If a problem,
;  658     0652  1 !THEN RETURN .status;                            ! stop now.
;  659     0653  1 !END;
;  660     0654  1
;  661     0655  1
;  662     0656  1 !
;  663     0657  1 ! The $CRENWV service takes as its argument a list of itemlist addresses.
;  664     0658  1 ! Set that up now.
;  665     0659  1 !
;  666     0660  1 !item_list[0] = item_list[2];                    ! Point to "real" itemlist
;  667     0661  1 !item_list[1] = 0;                               ! Zero-terminate.
;  668     0662  1
;  669     0663  1 !
;  670     0664  1 ! The journaling facility uses a pseudo-itemlist, which actually looks more
;  671     0665  1 ! like a descriptor list.  In the first longword is the itemcode and the
```

```
;  672          0666  1 ! length of the item; the second longword contains the address of the item.
;  673          0667  1 ! This list must be zero-terminated as well.  In all cases, we want to list
;  674          0668  1 ! the current device name, and the "close this file" flag.
;  675          0669  1 !
;  676          0670  1 !item_list[2] = (cnv$_flags^16) OR 4;           ! Flags are a longword
;  677          0671  1 !item_list[3] = UPLIT (cnv$m_close);            ! and say "close the file"
;  678          0672  1 !
;  679          0673  1 !item_list[4] = (cnv$_curdevnam^16) OR .curdev_len;     ! Current devname
;  680          0674  1 !item_list[5] = curdev;
;  681          0675  1 !
;  682          0676  1 !
;  683          0677  1 ! If the user wants the journal file on a different device, then put that
;  684          0678  1 ! item in the list, followed by the zero terminator.
;  685          0679  1 !
;  686          0680  1 !IF .disk[dsc$w_length] NEQ 0                  ! If to a different disk,
;  687          0681  1 !THEN                                          ! put that item in.
;  688          0682  1 !    BEGIN
;  689          0683  1 !    item_list[6] = (cnv$_newdevnam^16) OR .disk[dsc$w_length];
;  690          0684  1 !    item_list[7] = .disk[dsc$a_pointer];
;  691          0685  1 !    item_list[8] = 0;
;  692          0686  1 !    END
;  693          0687  1 !
;  694          0688  1 !
;  695          0689  1 ! Otherwise, simply zero-terminate the list.
;  696          0690  1 !
;  697          0691  1 !ELSE item_list[6] = 0;
;  698          0692  1 !
;  699          0693  1 !RETURN $CRENWV(CHAN   = .chan[0],              ! Do it.
;  700          0694  1 !               LSTADR = item_list);
;  701          0695  1 !
;  702          0696  1 !END;
```

```
;  703          0697  1 !
;  704          0698  1 END
;  705          0699  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $PLIT$                         592  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                          192  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         492  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        33         0      1000          00:01.8


;                                   COMMAND QUALIFIERS

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

; Size:          492 code + 784 data bytes
; Run Time:          00:15.4
; Elapsed Time:      00:49.9
; Lines/CPU Min:      2723
; Lexemes/CPU-Min:  14758
; Memory Used:   160 pages
; Compilation Complete
```

0052 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SETDEVICE.LIS

SETDEVS.LIS

SETCLUSUB.LIS

SETDIR.LIS

SETACT.LIS

SETAUDIT.LIS

SETCLUSTR.LIS