```
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
```

FILEID**LOGIN

LOG
V04

```
LL              000000      GGGGGGGG    IIIIII    NN      NN
LL              000000      GGGGGGGG    IIIIII    NN      NN
LL            00      00  GG              II      NN      NN
LL            00      00  GG              II      NN      NN
LL            00      00  GG              II      NNNN    NN
LL            00      00  GG              II      NNNN    NN
LL            00      00  GG              II      NN  NN  NN
LL            00      00  GG              II      NN  NN  NN
LL            00      00  GG  GGGGGG      II      NN    NNNN
LL            00      00  GG  GGGGGG      II      NN    NNNN
LL            00      00  GG      GG      II      NN      NN      ....
LL            00      00  GG      GG      II      NN      NN      ....
LLLLLLLLLL      000000      GGGGGG      IIIIII    NN      NN      ....
LLLLLLLLLL      000000      GGGGGG      IIIIII    NN      NN      ....

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 login (IDENT = 'V04-000',
;     2          0002  0                MAIN = start,
;     3          0003  0                ADDRESSING_MODE(EXTERNAL = GENERAL)) =
;     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:  Login
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This image initializes the process context for a particular
;    36          0036  1 !       user and transfers control the the command language interpreter.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:
;    39          0039  1 !
;    40          0040  1 !       VAX/VMS operating system.
;    41          0041  1 !
;    42          0042  1 ! AUTHOR:  Tim Halvorsen, March 1981
;    43          0043  1 !
;    44          0044  1 ! Modified by:
;    45          0045  1 !
;    46          0046  1 !       V03-033 ACG0436         Andrew C. Goldstein,    23-Jul-1984  16:35
;    47          0047  1 !               Add central check for DISUSER flag; set DISUSER on break-in
;    48          0048  1 !               if enabled.
;    49          0049  1 !
;    50          0050  1 !       V03-032 ACG0432         Andrew C. Goldstein,    9-Jul-1984  11:52
;    51          0051  1 !               Implement PRC$V_NOPASSWORD flag; disable LOG_IO privilege
;    52          0052  1 !               before logout for correct /HANGUP control (image is now
;    53          0053  1 !               installed with LOG_IO).
;    54          0054  1 !
;    55          0055  1 !       V03-031 JRL0008         John R. Lawson, Jr.     25-Jun-1984 10:07
;    56          0056  1 !               Add support for AUTHORIZE/PWDEXPIRED which creates
;    57          0057  1 !               pre-expired passwords ... Set UAF$V_PWD_EXPIRED if
```

```
;    58          0058  1 !               UAF$Q_PWD_DATE lss 0.
;    59          0059  1 !
;    60          0060  1 !       V03-030 MHB0143         Mark Bramhall           2-May-1984
;    61          0061  1 !               Changes for new account name conventions.
;    62          0062  1 !               Limit failing password to NSA$S_PKT_PASSWORD.
;    63          0063  1 !               Use LGI$_NOSUCHUSER and LGI$_INVPWD instead of -2 and -4.
;    64          0064  1 !
;    65          0065  1 !       V03-029 MHB0124         Mark Bramhall           11-Apr-1984
;    66          0066  1 !               Reset creator's CLI and table names if using UAF.
;    67          0067  1 !               Add security auditing for LOGIN failures.
;    68          0068  1 !               Security audit breakin attempts.
;    69          0069  1 !               Give advance warning on password expirations.
;    70          0070  1 !               Add routine to set a terminal's hangup state.
;    71          0071  1 !               Always hang up the terminal on login failures.
;    72          0072  1 !               Move UAF$V_AUDIT to PCB_STS [PCB$V_SECAUDIT] when UAF is read.
;    73          0073  1 !               Change initial logic that gathers terminal inforation.
;    74          0074  1 !               Cancel CLI AST request(s) on PPD$W_INPCHAN during LOGOUTs.
;    75          0075  1 !
;    76          0076  1 !       V03-028 MHB0116         Mark Bramhall           23-Mar-1984
;    77          0077  1 !               Add security auditing for successful LOGINs.
;    78          0078  1 !               Add security auditing for LOGOUTs.
;    79          0079  1 !
;    80          0080  1 !       V03-027 MHB0113         Mark Bramhall           21-Mar-1984
;    81          0081  1 !               Remove temporary job type fetch kludge.
;    82          0082  1 !               Use LNM services for logical names.
;    83          0083  1 !               Add first cut at automated re-connection.
;    84          0084  1 !               Make LOGOUT use current CLI's result parse via SYS$CLI.
;    85          0085  1 !               Reference PCB_STS as a BITVECTOR.
;    86          0086  1 !               Set TERM_NAME, PHY_TERM_NAME, and CLU_TERM_NAME on entry.
;    87          0087  1 !
;    88          0088  1 !       V03-026 LJK0262         Lawrence J. Kenah       28-Feb-1984
;    89          0089  1 !               Use LGI$CMSUPR to rundown CLI.
;    90          0090  1 !
;    91          0091  1 !       V03-025 PCG0001         Peter George            31-Jan-1984  12:51
;    92          0092  1 !               Add check for account expiration.
;    93          0093  1 !               Do not open UAF for subprocesses.
;    94          0094  1 !               Add FAB and RAB arguments to call to lgi$searchuser.
;    95          0095  1 !               Update appropriate UAF fields.
;    96          0096  1 !
;    97          0097  1 !       V03-024 ACG0391         Andrew C. Goldstein,    18-Jan-1984  18:13
;    98          0098  1 !               Specify proper length when protecting the PPD
;    99          0099  1 !
;   100          0100  1 !       V03-023 ACG0385         Andrew C. Goldstein,    28-Dec-1983  17:41
;   101          0101  1 !               Handle extended username and account in UAF,
;   102          0102  1 !               implement new restriction flags
;   103          0103  1 !
;   104          0104  1 !       V03-022 ACG0379         Andrew C. Goldstein,    6-Dec-1983  19:33
;   105          0105  1 !               Make GET_UAFREC return status to handle the OPA0: case
;   106          0106  1 !
;   107          0107  1 !       V03-021 ACG0376         Andrew C. Goldstein,    18-Nov-1983  19:11
;   108          0108  1 !               Process /[NO]HANGUP only if SYS$INPUT is a terminal;
;   109          0109  1 !               use physical terminal name for OPA0 check. Interface
;   110          0110  1 !               cleanups in UAF validation and process type decision.
;   111          0111  1 !               Cancel CLI's exit handlers before doing logout or job
;   112          0112  1 !               step processing.
;   113          0113  1 !
;   114          0114  1 !       V03-020 ACG0363         Andrew C. Goldstein,    27-Sep-1983  18:09
```

```
;  115     0115  1 !               Use $GETDVI to check for OPA0: for device name independence
;  116     0116  1 !               in UAF presence check
;  117     0117  1 !
;  118     0118  1 !       V03-019 GAS0188         Gerry Smith                     22-Sep-1983
;  119     0119  1 !               Change the sense of the call to CIA_SCAN for the case of
;  120     0120  1 !               successful logins.
;  121     0121  1 !
;  122     0122  1 !       V03-018 GAS0183         Gerry Smith                     16-Sep-1983
;  123     0123  1 !               Add support for breakin evasion.  This is done by making
;  124     0124  1 !               checks in VALIDATE_UAFREC.
;  125     0125  1 !
;  126     0126  1 !       V03-017 GAS0169         Gerry Smith                     23-Aug-1983
;  127     0127  1 !               For interactive processes, open SYS$INPUT and SYS$OUTPUT
;  128     0128  1 !               unconditionally, rather than trying to figure out if we only
;  129     0129  1 !               need to open one and fake the other fab out.  The rationale
;  130     0130  1 !               for this is that interactive processes may not have both
;  131     0131  1 !               input and output to the same place -- that is the choice of
;  132     0132  1 !               the individual who requested that the process be created.
;  133     0133  1 !               The additional overhead involved is somewhat less than one
;  134     0134  1 !               page in P1 space, which is not enough to be considered
;  135     0135  1 !               significant at present.
;  136     0136  1 !
;  137     0137  1 !       V03-016 GAS0163         Gerry Smith                     30-Jul-1983
;  138     0138  1 !               Make the device class and device characteristics of
;  139     0139  1 !               SYS$INPUT global.  These fields are referenced in a number
;  140     0140  1 !               of different modules, and $GETDVI should not have to be
;  141     0141  1 !               called in each module.
;  142     0142  1 !
;  143     0143  1 !       V03-015 MLJ0115         Martin L. Jack, 29-Jul-1983  10:30
;  144     0144  1 !               Update for new log file error handling.
;  145     0145  1 !
;  146     0146  1 !       V03-014 GAS0146                 Gerry Smith             22-Jun-1983
;  147     0147  1 !               Add support for PCB$V_INTER.  This bit declares that the
;  148     0148  1 !               process is interactive.  Thus, there are now five distinct
;  149     0149  1 !               classes of process: batch, network, subprocess, interactive,
;  150     0150  1 !               and detached.  From now on, any process which is to be
;  151     0151  1 !               considered an interactive top-level process, must have the
;  152     0152  1 !               INTER bit set in PCB$L_STS, otherwise it will be seen as a
;  153     0153  1 !               detached process, and treated accordingly.  This does away
;  154     0154  1 !               with all the kludge of $ASSIGN/$PARSE/compare.
;  155     0155  1 !
;  156     0156  1 !       V03-013 WMC0001                 Wayne Cardoza           26-May-1983
;  157     0157  1 !               Add SYS$LOGIN_DEVICE.
;  158     0158  1 !
;  159     0159  1 !       V03-012 GAS0123                 Gerry Smith             19-Apr-1983
;  160     0160  1 !               Change to use new $SNDJBC interface.
;  161     0161  1 !
;  162     0162  1 !       V03-011 TMH0011                 Tim Halvorsen           07-Feb-1983
;  163     0163  1 !               Make process status flags global, so that they can be
;  164     0164  1 !               referenced in another module.
;  165     0165  1 !
;  166     0166  1 !       V03-010 GAS0090                 Gerry Smith             15-Oct-1982
;  167     0167  1 !               Finish GAS49169.  If network or batch process, skip the
;  168     0168  1 !               parsing, since the parsing is only performed to determine
;  169     0169  1 !               whether the process is interactive.
;  170     0170  1 !
;  171     0171  1 !       V03-009 GAS0089                 Gerry Smith              5-Oct-1982
```

```
;  172     0172  1 !               If the /DISK qualifier was specified, then use that device
;  173     0173  1 !               for both SYS$SCRATCH and SYS$LOGIN.  Otherwise use whatever
;  174     0174  1 !               is in the UAF as default device.
;  175     0175  1 !
;  176     0176  1 !       V03-008 GAS49169                Gerry Smith             30-Sep-1982
;  177     0177  1 !               Fix the error path for the $PARSE, so that if either
;  178     0178  1 !               $PARSE fails, an error is signaled.  This fixes the problem
;  179     0179  1 !               of someone accessing a terminal after the job-controller
;  180     0180  1 !               initiates a login, but before LOGINOUT can grab the terminal.
;  181     0181  1 !
;  182     0182  1 !       V03-007 GAS49016                Gerry Smith             16-Sep-1982
;  183     0183  1 !               When checking for DISDIALUP, make the check for terminals
;  184     0184  1 !               whose MODEM bit is set, rather than the REMOTE bit.
;  185     0185  1 !
;  186     0186  1 !       V03-006 GAS0082         Gerry Smith             14-May-1982
;  187     0187  1 !               For network processes, if no-authorization bit set,
;  188     0188  1 !               allow login if no UAF record found.
;  189     0189  1 !
;  190     0190  1 !       V03-005 GAS0081         Gerry Smith             12-May-1982
;  191     0191  1 !               After network init, check to see if there is a UAF
;  192     0192  1 !               record present.  If not, signal an authorization
;  193     0193  1 !               error and exit.
;  194     0194  1 !
;  195     0195  1 !       V03-004 GAS0078         Gerry Smith              3-May-1982
;  196     0196  1 !               Have the check for job quotas be for interactive jobs
;  197     0197  1 !               only.
;  198     0198  1 !
;  199     0199  1 !       V03-003 GAS0074         Gerry Smith             14-Apr-1982
;  200     0200  1 !               Complete the fix for GAS0032.  It is possible
;  201     0201  1 !               to turn off hidden devices, so that either _OPA0: or
;  202     0202  1 !                 OPA0: is the legal name for the operator console.
;  203     0203  1 !               Also fix security hole in which an invalid username
;  204     0204  1 !               results in an immediate error message.  The correct
;  205     0205  1 !               behavior is to prompt for password, then signal an
;  206     0206  1 !               error.
;  207     0207  1 !
;  208     0208  1 !       V03-002 GAS0072         Gerry Smith             13-Apr-1982
;  209     0209  1 !               Assign a channel to SYS$INPUT before $PARSEing the
;  210     0210  1 !               input and output fields.  This is due to the fact
;  211     0211  1 !               that REMACP triggers on the "last de-assign" of
;  212     0212  1 !               RTAn to determine that the process should terminate.
;  213     0213  1 !               The $ASSIGN keeps a channel open for the duration of
;  214     0214  1 !               the login procedure, until SYS$INPUT and SYS$OUTPUT
;  215     0215  1 !               get opened as process-permanent files.
;  216     0216  1 !
;  217     0217  1 !       V03-001 GAS0070         Gerry Smith              5-Apr-1982
;  218     0218  1 !               Modify the way that a process is deemed interactive.
;  219     0219  1 !               Instead of simply comparing the equivalence names for
;  220     0220  1 !               SYS$INPUT and SYS$OUTPUT, perform a $PARSE on the two,
;  221     0221  1 !               and compare the DVI fields in the NAM blocks.  If the
;  222     0222  1 !               two are exactly equal, and if SYS$INPUT is a terminal,
;  223     0223  1 !               open only one file for both input and output, and set
;  224     0224  1 !               the process to be interactive.
;  225     0225  1 !
;  226     0226  1 !       V03-008 GAS0051         Gerry Smith             23-Feb-1982
;  227     0227  1 !               Fix bug that caused terminals not to get hung up if
;  228     0228  1 !               /HANGUP was specified.  Also, increase size of device
```

```
;  229     0229  1 !                     characteristics buffer, to keep it from tromping on
;  230     0230  1 !                     other data.
;  231     0231  1 !
;  232     0232  1 !               V03-007 GAS0036         Gerry Smith             25-Jan-1982
;  233     0233  1 !                     Add /[NO]HANGUP to LOGOUT.  This causes the terminal
;  234     0234  1 !                     to drop DTR when the last charnel is deassigned from
;  235     0235  1 !                     it during logout.
;  236     0236  1 !
;  237     0237  1 !               V03-006 PCG0001         Peter George            18-Jan-1982
;  238     0238  1 !                     Call CLI$END_PARSE after LOGOUT command parsing.
;  239     0239  1 !
;  240     0240  1 !               V03-005 GAS0032         Gerry Smith             07-Jan-1982
;  241     0241  1 !                     If no SYSUAF.DAT exists, change the check to see if
;  242     0242  1 !                     the user is logging in on __OPA0: instead of _OPA0:
;  243     0243  1 !
;  244     0244  1 !               V03-004 HRJ0033         Herb Jacobs             11-Dec-1981
;  245     0245  1 !                     Add time of day and terminal type verification.
;  246     0246  1 !                     Add support of WSEXTENT from batch queues.
;  247     0247  1 !                     Fix spooling of batch job if error opening primary input.
;  248     0248  1 !                     Fix maximization of WSEXTENT from authorize record.
;  249     0249  1 !
;  250     0250  1 !               V003    TMH0003         Tim Halvorsen   11-Nov-1981
;  251     0251  1 !                     Display different message when logging off from a
;  252     0252  1 !                     subprocess vs. the root process.
;  253     0253  1 !                     Set SYS$LOGIN and SYS$SCRATCH correctly for a subprocess
;  254     0254  1 !                     by setting it to the directory specified in the UAF record
;  255     0255  1 !                     rather than the current default directory when the process
;  256     0256  1 !                     was created.
;  257     0257  1 !                     Fix translation of SYS$ERROR so it handles the case where
;  258     0258  1 !                     it has no translation (makes it a null string).
;  259     0259  1 !
;  260     0260  1 !               V002    TMH0002         Tim Halvorsen   27-Oct-1981
;  261     0261  1 !                     Allocate LGI region during PPD initialization.  Move
;  262     0262  1 !                     initialization of CLIREG and ORIGUIC here from inituser.
;  263     0263  1 !                     Do not re-define SYS$ERROR if the original SYS$ERROR
;  264     0264  1 !                     translation was not the same as the original SYS$OUTPUT
;  265     0265  1 !                     translation when the process was created, so that a user
;  266     0266  1 !                     can pass an "argument" to a subprocess via SYS$ERROR
;  267     0267  1 !                     without it getting overwritten.  Instead, the original
;  268     0268  1 !                     SYS$ERROR translation is passed as a supervisor mode
;  269     0269  1 !                     logical name, and the new SYS$ERROR (same as SYS$OUTPUT)
;  270     0270  1 !                     is passed as an executive mode logical name.
;  271     0271  1 !
;  272     0272  1 !               V001    TMH0001         Tim Halvorsen   29-Jun-1981
;  273     0273  1 !                     Close output file for non-batch jobs (specifically
;  274     0274  1 !                     network jobs) so that the file is truncated.
;  275     0275  1 !--
;  276     0276  1
;  277     0277  1 !
;  278     0278  1 !  Include files
;  279     0279  1 !
;  280     0280  1
;  281     0281  1 LIBRARY 'SYS$LIBRARY:LIB';                      ! VAX/VMS system definitions
;  282     0282  1 REQUIRE 'SHRLIB$:UTILDEF';                      ! Common BLISS definitions
;  283     0467  1 REQUIRE 'SHRLIB$:INTDEF';                       ! CLI common "internal" definitions
;  284     0493  1 REQUIRE 'LIB$:PPDDEF';                          ! Process permanent data region
;  285     0640  1 REQUIRE 'LIB$:LGIDEF';                          ! LOGINOUT-private permanent storage
```

```
;  287     0711  1 !
;  288     0712  1 ! Table of contents
;  289     0713  1 !
;  290     0714  1
;  291     0715  1 FORWARD ROUTINE
;  292     0716  1     start,                          ! Main routine
;  293     0717  1     login:              NOVALUE,    ! Login portion
;  294     0718  1     logout:             NOVALUE,    ! Logout portion
;  295     0719  1     check_job_quotas:   NOVALUE,    ! Check interactive job quotas
;  296     0720  1     check_user_quotas,              ! Check user and account job quotas
;  297     0721  1     check_user_hours:   NOVALUE,    ! Check user hourly restrictions
;  298     0722  1     check_account_expiration: NOVALUE,  ! Check account expiration
;  299     0723  1     update_uaf_record:  NOVALUE,    ! Update the UAF record
;  300     0724  1     set_ppd_prot,                   ! Set page protection on PPD structure
;  301     0725  1     set_sysprv:         NOVALUE,    ! Set SYSPRV privilege
;  302     0726  1     clear_sysprv:       NOVALUE,    ! Clear SYSPRV privilege
;  303     0727  1     clear_log_io:       NOVALUE,    ! Clear LOG_IO privilege
;  304     0728  1     validate_uafrec:    NOVALUE,    ! Read/validate UAF record
;  305     0729  1     get_uafrec,                     ! Read UAF record without validiation
;  306     0730  1     rundown_cli,                    ! Eliminate supervisor mode stuff
;  307     0731  1     logout_message:     NOVALUE,    ! Write logout message
;  308     0732  1     handler,                        ! Condition handler
;  309     0733  1     exit_process:       NOVALUE,    ! Exit the process
;  310     0734  1     clear_creator_cli:  NOVALUE,    ! Clear creator's CLI and CLI table
;  311     0735  1     set_terminal_hangup: NOVALUE;   ! Set/clear terminal's hangup state
;  312     0736  1
;  313     0737  1 !
;  314     0738  1 ! External routines
;  315     0739  1 !
;  316     0740  1
;  317     0741  1 EXTERNAL ROUTINE
;   18     0742  1     open_input:         NOVALUE,    ! Open primary input file
;   19     0743  1     close_input:        NOVALUE,    ! Close primary input file
;  _20     0744  1     open_output:        NOVALUE,    ! Open primary output file
;  321     0745  1     close_output:       NOVALUE,    ! Close primary output file
;  322     0746  1     set_account:        NOVALUE,    ! Set account name in JIB and P1 space
;  323     0747  1     init_interactive:   NOVALUE,    ! Initialize interactive job
;  324     0748  1     check_connection:   NOVALUE,    ! Check for (re-)connection(s)
;  325     0749  1     announce:           NOVALUE,    ! Print welcome message
;  326     0750  1     init_batch:         NOVALUE,    ! Initialize batch job step
;  327     0751  1     init_network:       NOVALUE,    ! Initialize network job
;  328     0752  1     init_user:          NOVALUE,    ! Initialize user process quotas, etc.
;  329     0753  1     init_cli:           NOVALUE,    ! Initialize CLI image
;  330     0754  1     execute_cli:        NOVALUE,    ! Call the CLI image at its entry point
;  331     0755  1     create_logical,                 ! Create logical name with LNM services
;  332     0756  1     write_fao:          NOVALUE,    ! Write formatted message to output
;  333     0757  1     write_output,                   ! Write to primary output stream
;  334     0758  1     terminate_batch:    NOVALUE,    ! Stop batch job, optionally spool log
;  335     0759  1     sys$setddir,                    ! Set default directory
;  336     0760  1     lgi$check_pass,                 ! Validate password against UAF record
;  337     0761  1     lgi$searchuser,                 ! Read UAF record
;  338     0762  1     security_audit:     NOVALUE,    ! Perform a security audit
;  339     0763  1     cia_scan,                       ! Check for suspect/intruders
;  340     0764  1     lgi$cmsupr,                     ! Change mode to supervisor
;  341     0765  1     ascic_day_of_week,              ! Return ASCIC day of week
;  342     0766  1     lib$day_of_week,                ! Find which day of the week this is
;  343     0767  1     lib$hour_of_day,                ! Find which hour of the day this is
```

```
;  344     0768  1     lib$get_vm;                             . Allocate virtual memory
;  345     0769  1
;  346     0770  1 !
;  347     0771  1 ! External storage
;  348     0772  1 !
;  349     0773  1
;  350     0774  1 EXTERNAL
;  351     0775  1     write_output_status,                    ! WRITE_OUTPUT's last status
;  352     0776  1     input_fab:          BBLOCK,             ! Input FAB
;  353     0777  1     input_nam:          BBLOCK,             ! Input NAM
;  354     0778  1     output_fab:         BBLOCK,             ! Output FAB
;  355     0779  1     output_rab:         BBLOCK,             ! Output RAB
;  356     0780  1     output_nam:         BBLOCK,             ! Output NAM
;  357     0781  1     disk_name:          VECTOR,             ! Diskname descriptor
;  358     0782  1     ctl$gl_creprc_flags:BBLOCK,             ! Flags specified to $CREPRC
;  359     0783  1     ctl$ag_climage,                         ! Address of CLI code in control region
;  360     0784  1     ctl$ag_clidata,                         ! Process permanent data region
;  361     0785  1     exe$gl_dynamic_flags: BITVECTOR;        ! Dynamic system flags
;  362     0786  1
;  363     0787  1 EXTERNAL LITERAL
;  364     0788  1         exe$v_brk_disuser : UNSIGNED (6); ! Set DISUSER if breakin
;  365     0789  1
;  366     0790  1 BIND
;  367     0791  1     ppd = ctl$ag_clidata: BBLOCK;           ! Address of PPD structure
;  368     0792  1
;  369     0793  1 !
;  370     0794  1 ! Define message codes
;  371     0795  1 !
;  372     0796  1
;  373     0797  1 EXTERNAL LITERAL
;  374     0798  1     cli$_present,
;  375     0799  1     cli$_negated,
;  376     0800  1     lgi$_evade,
;  377     0801  1     lgi$_logdisabl,
;  378     0802  1     lgi$_exquota,
;  379     0803  1     lgi$_notvalid,
;  380     0804  1     lgi$_fileacc,
;  381     0805  1     lgi$_userauth,
;  382     0806  1     lgi$_userexc,
;  383     0807  1     lgi$_acntexc,
;  384     0808  1     lgi$_badhour,
;  385     0809  1     lgi$_badday,
;  386     0810  1     lgi$_restrict,
;  387     0811  1     lgi$_inputerr,
;  388     0812  1     lgi$_acntexpir,
;  389     0813  1     lgi$_pwdexpir,
;  390     0814  1     lgi$_nosuchuser,
;  391     0815  1     lgi$_invpwd;
;  392     0816  1
;  393     0817  1 !
;  394     0818  1 ! Macro to set the processor interrupt priority level register
;  395     0819  1 !
;  396     0820  1
;  397     0821  1 BUILTIN
;  398     0822  1         MTPR;
;  399     0823  1
;  400     0824  1 MACRO                                       ! set processor IPL
```

```
;  401          0825  1          SET_IPL (LEVEL) = MTPR (%REF (LEVEL), PR$_IPL)%;
;  402          0826  1
;  403          0827  1 !
;  404          0828  1 ! Macro to setup a GETJPI item list or GETDVI item list
;  405          0829  1 !
;  406          0830  1
;  407          0831  1 MACRO
;  408        M 0832  1     setup_jpidvi_list2 (ptr) [item_code, buflen, bufadr] =
;  409        M 0833  1          BEGIN
;  410        M 0834  1          ptr [0,0,16,0] = buflen;
;  411        M 0835  1          ptr [2,0,16,0] = item_code;
;  412        M 0836  1          ptr [4,0,32,0] = bufadr;
;  413        M 0837  1          ptr [8,0,32,0] = 0;
;  414        M 0838  1          ptr = .ptr + 12;
;  415          0839  1          END%,
;  416          0840  1
;  417        M 0841  1     setup_jpidvi_list (buffer) =
;  418        M 0842  1          BEGIN
;  419        M 0843  1          LOCAL ptr: REF BBLOCK;
;  420        M 0844  1          ptr = buffer;
;  421        M 0845  1          setup_jpidvi_list2(ptr, %REMAINING);
;  422          0846  1          END%;
;  423          0847  1
;  424          0848  1 !
;  425          0849  1 ! Flags
;  426          0850  1 !
;  427          0851  1
;  428          0852  1 GLOBAL
;  429          0853  1     pcb_sts:            BITVECTOR[32],          ! PCB status flags
;  430          0854  1     job_type,                                   ! Job type from JIB
;  431          0855  1     terminal_device:    BYTE INITIAL(false),    ! True if SYS$INPUT is terminal
;  432          0856  1     dev_char_2:         $BBLOCK[4],             ! Dev char #2 of SYS$INPUT
;  433          0857  1     dev_dep_2:          $BBLOCK[4],             ! Dev dep #2 of SYS$INPUT
;  434          0858  1     subprocess:         BYTE INITIAL(false),    ! True if subprocess
;  435          0859  1     image_activate:     BYTE INITIAL(false),    ! True if image to be chained
;  436          0860  1     break_attempt:      BYTE INITIAL(false),    ! True if breakin attempt
;  437          0861  1     dummy:              VECTOR [1, BYTE];       ! Alignment filler
;  438          0862  1
;  439          0863  1 OWN
;  440          0864  1     doing_login:        BYTE INITIAL(false),    ! True if doing a login
;  441          0865  1     full_logout:        BYTE INITIAL(false),    ! Give full logout message
;  442          0866  1     dummy_1:            VECTOR [2, BYTE];       ! Alignment filler
;  443          0867  1
;  444          0868  1 !
;  445          0869  1 ! Own storage
;  446          0870  1 !
;  447          0871  1
;  448          0872  1 OWN
;  449          0873  1     fail_password_buff:                         ! Buffer for failing password
;  450          0874  1                 VECTOR [nsa$s_pkt_password,BYTE],
;  451          0875  1     term_buff: VECTOR [8,BYTE],                 ! Buffer for terminal name
;  452          0876  1     phy_term_buff: VECTOR [8,BYTE],             ! Buffer for terminal name
;  453          0877  1     clu_term_buff: VECTOR [16,BYTE],            ! Buffer for terminal name
;  454          0878  1     sys$input_buffer: VECTOR [128,BYTE],        ! SYS$INPUT buffer
;  455          0879  1     sys$output_buffer: VECTOR [128,BYTE],       ! SYS$OUTPUT buffer
;  456          0880  1     sys$error_buffer: VECTOR [128,BYTE];        ! SYS$ERROR buffer
;  457          0881  1
```

```
;  458          0882  1 GLOBAL
;  459          0883  1     uaf_fab: $FAB_DECL,                       ! FAB to open UAF
;  460          0884  1     uaf_rab: $RAB_DECL;                       ! RAB to read UAF record
;  461          0885  1
;  462          0886  1 OWN
;  463          0887  1     rundown_list: VECTOR [3]                  ! Argument list for LGI$CMSUPR
;  464          0888  1                   INITIAL (2, rundown_cl:, 0);
;  465          0889  1
;  466          0890  1 GLOBAL
;  467          0891  1     uaf_record: REF BBLOCK,                   ! Address of user authorization record
;  468          0892  1     org_username: VECTOR [uaf$s_username, BYTE], ! Username proc was started with
;  469          0893  1     parent_pid,                               ! PID of parent process
;  470          0894  1     creator_username: VECTOR [2]              ! Username of creating process
;  471          0895  1                   INITIAL(0, org_username),
;  472          0896  1     fail_password: VECTOR [2]                 ! Descriptor of failing password
;  473          0897  1                   INITIAL(0, fail_password_buff),
;  474          0898  1     term_name:  VECTOR [2]                    ! Terminal name string descriptor
;  475          0899  1                 INITIAL(0, term_buff),
;  476          0900  1     phy_term_name: VECTOR [2]                 ! Phys terminal name string descriptor
;  477          0901  1                 INITIAL(0, phy_term_buff),
;  478          0902  1     clu_term_name: VECTOR [2]                 ! Cluster terminal name string descriptor
;  479          0903  1                 INITIAL(0, clu_term_buff),
;  480          0904  1     sys$input_attr,                           ! SYS$INPUT logical name attributes
;  481          0905  1     sys$input:  VECTOR [2]                    ! SYS$INPUT descriptor
;  482          0906  1                 INITIAL(%ALLOCATION(sys$input_buffer),
;  483          0907  1                         sys$input_buffer),
;  484          0908  1     sys$output_attr,                          ! SYS$OUTPUT logical name attributes
;  485          0909  1     sys$output: VECTOR [2]                    ! SYS$OUTPUT descriptor
;  486          0910  1                 INITIAL(%ALLOCATION(sys$output_buffer),
;  487          0911  1                         sys$output_buffer),
;  488          0912  1     sys$error_attr,                           ! SYS$ERROR logical name attributes
;  489          0913  1     sys$error:  VECTOR [2]                    ! SYS$ERROR descriptor
;  490          0914  1                 INITIAL(%ALLOCATION(sys$error_buffer),
;  491          0915  1                         sys$error_buffer);
;  492          0916  1
;  493          0917  1 LITERAL
;  494          0918  1     bell = 7;
```

```
;  496     0919  1 ROUTINE start =
;  497     0920  1
;  498     0921  1 !---
;  499     0922  1 !
;  500     0923  1 !       This is the main entry point to the image.  The login data structure
;  501     0924  1 !       is examined in the control region to determine if this is a login or
;  502     0925  1 !       a logout operation.
;  503     0926  1 !
;  504     0927  1 ! Inputs:
;  505     0928  1 !
;  506     0929  1 !       None
;  507     0930  1 !
;  508     0931  1 ! Outputs:
;  509     0932  1 !
;  510     0933  1 !       None
;  511     0934  1 !---
;  512     0935  1
;  513     0936  2 BEGIN
;  514     0937  2
;  515     0938  2 BUILTIN FP;
;  516     0939  2
;  517     0940  2 !
;  518     0941  2 ! Establish condition handler
;  519     0942  2 !
;  520     0943  2 .fp = handler;                          ! Establish condition handler
;  521     0944  2
;  522     0945  2 !
;  523     0946  2 ! Get job status flags to determine type of job.
;  524     0947  2 !
;  525     0948  3 BEGIN
;  526     0949  3
;  527     0950  3 LOCAL
;  528     0951  3     dev_class,                          ! Device class of SYS$INPUT
;  529     0952  3     item_list:  $ITMLST_DECL(ITEMS=5);  ! GETJPI/GETDVI item list
;  530     0953  3
;  531   P 0954  3 $ITMLST_INIT(ITMLST = item_list,
;  532   P 0955  3               (ITMCOD = jpi$_username,
;  533   P 0956  3                BUFSIZ = uaf$s_username,
;  534   P 0957  3                BUFADR = org_username,
;  535   P 0958  3                RETLEN = creator_username),
;  536   P 0959  3               (ITMCOD = jpi$_sts,
;  537   P 0960  3                BUFADR = pcb_sts),
;  538   P 0961  3               (ITMCOD = jpi$_jobtype,
;  539   P 0962  3                BUFADR = job_type),
;  540   P 0963  3               (ITMCOD = jpi$_owner,
;  541     0964  3                BUFADR = parent_pid));
;  542     0965  3
;  543     0966  3 pcb_sts = 0;
;  544     0967  3 job_type = 0;
;  545     0968  3 parent_pid = 0;
;  546     0969  3 ch$fill (' ', uaf$s_username, org_username);
;  547     0970  3 return_if_error($GETJPIW(ITMLST = item_list));  ! Obtain PCB flags
;  548     0971  3
;  549     0972  3 subprocess = (.parent_pid NEQ 0);       ! Mark if this is a subprocess or not
;  550     0973  3
;  551     0974  3 !
;  552     0975  3 ! Get characteristics and names of the SYS$INPUT device.  Believe the
```

```
;  553          0976  3 ! SYS$INPUT logical for logins, but use the name from the PPD for logouts.
;  554          0977  3 ! This information is for later use throughout LOGINOUT.
;  555          0978  3 !
;  556        P 0979  3 $ITMLST_INIT(ITMLST = item_list,
;  557        P 0980  3                (ITMCOD = dvi$_devclass,
;  558        P 0981  3                 BUFADR = dev_class),
;  559        P 0982  3                (ITMCOD = dvi$_devchar2,
;  560        P 0983  3                 BUFADR = dev_char_2),
;  561        P 0984  3                (ITMCOD = dvi$_devdepend2,
;  562        P 0985  3                 BUFADR = dev_dep_2),
;  563        P 0986  3                (ITMCOD = dvi$_devnam,
;  564        P 0987  3                 BUFSIZ = %ALLOCATION(term_buff),
;  565        P 0988  3                 BUFADR = term_buff,
;  566        P 0989  3                 RETLEN = term_name[0]),
;  567        P 0990  3                (ITMCOD = dvi$_tt_phydevnam,
;  568        P 0991  3                 BUFSIZ = %ALLOCATION(phy_term_buff),
;  569        P 0992  3                 BUFADR = phy_term_buff,
;  570          0993  3                 RETLEN = phy_term_name[0]));
;  571          0994  3
;  572          0995  3 dev_class = 0;
;  573        P 0996  3 IF $GETDVIW(ITMLST = item_list,
;  574        P 0997  3                DEVNAM = (IF .ppd[ppd$w_size] EQL 0
;  575        P 0998  3                          THEN %ASCID 'SYS$INPUT'
;  576        P 0999  3                          ELSE
;  577        P 1000  3                          (LOCAL devnam_desc : VECTOR[2];
;  578        P 1001  3                           devnam_desc[0] = .ppd[$BYTEOFFSET(ppd$t_inpdvi),0,8,0];
;  579        P 1002  3                           devnam_desc[1] = ppd[ppd$t_inpdvi] + 1;
;  580        P 1003  3                           devnam_desc)
;  581          1004  4                         ))
;  582          1005  3 AND .dev_class EQL dc$_term
;  583          1006  3 THEN
;  584          1007  4     BEGIN
;  585          1008  4     terminal_device = true;              ! It is a terminal device
;  586          1009  4     IF .phy_term_name[0] EQL 0           ! If no physical name,
;  587          1010  4     THEN CH$MOVE((phy_term_name[0] = .term_name[0]), ! Set one
;  588          1011  4                  term_buff, phy_term_buff); !  from original
;  589        P 1012  4     $ITMLST_INIT(ITMLST = item_list,
;  590        P 1013  4                 (ITMCOD = dvi$_fulldevnam,
;  591        P 1014  4                  BUFSIZ = %ALLOCATION(clu_term_buff),
;  592        P 1015  4                  BUFADR = clu_term_buff,
;  593          1016  4                  RETLEN = clu_term_name[0]));
;  594        P 1017  4     return_if_error($GETDVIW(ITMLST = item_list,
;  595          1018  4                              DEVNAM = phy_term_name));
;  596          1019  4     END
;  597          1020  3 ELSE
;  598          1021  4     BEGIN
;  599          1022  4     terminal_device = false;             ! Not a terminal device
;  600          1023  4     dev_char_2 = 0;                      ! Ensure characteristics
;  601          1024  4     dev_dep_2 = 0;                       !  and dependent bits are null
;  602          1025  4     term_name[0] = 0;                    ! Ensure
;  603          1026  4     phy_term_name[0] = 0;                !  terminal names
;  604          1027  4     clu_term_name[0] = 0;                !   are null
;  605          1028  3     END;
;  606          1029  3
;  607          1030  2 END;
;  608          1031  2
;  609          1032  2 !
```

```
;  610          1033  2 ! Drop the installed SYSPRV privilege, unless the parent is authorized,
;  611          1034  2 ! so that random caller's of this program cannot use its privileges to
;  612          1035  2 ! create files in a privileged directory.  Note that the caller may be
;  613          1036  2 ! the job controller, which initializes the process with all privileges
;  614          1037  2 ! in order to ensure that the terminal is accessible even though it has
;  615          1038  2 ! device protection.
;  616          1039  2 !
;  617          1040  2
;  618          1041  2 clear_sysprv();                          ! Drop SYSPRV privilege
;  619          1042  2
;  620          1043  2 !
;  621          1044  2 ! Assume that when the process is initially created, the PPD is a demand
;  622          1045  2 ! zero page.  We can simply test the length of the structure to determine
;  623          1046  2 ! if its already been set up.  If so, its a logout operation
;  624          1047  2 !
;  625          1048  2
;  626          1049  3 IF NOT (doing_login = (.ppd [ppd$w_size] EQL 0)) ! If already initialized,
;  627          1050  2 THEN
;  628          1051  2     logout()                             ! then invoke logout portion
;  629          1052  2 ELSE
;  630          1053  2     login();                             ! else, initialize user job
;  631          1054  2
;  632          1055  2 RETURN true;
;  633          1056  2
;  634          1057  1 END;


                                                        .TITLE  LOGIN
                                                        .IDENT  \V04-000\

                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

00  00  00  54  55  50  4E  49  24  53  59  53  00000 P.AAB:  .ASCII  \SYS$INPUT\<0><0><0>                    ;
                            010E0009  0000C P.AAA:  .LONG   17694729                                        ;
                            00000000'  00010        .ADDRESS P.AAB                                          ;

                                                        .PSECT  $OWN$,NOEXE,2

                                  00  00000 DOING_LOGIN:
                                                        .BYTE   0                                           ;
                                  00  00001 FULL_LOGOUT:
                                                        .BYTE   0                                           ;
                                      00002             .BLKB   2
                                      00004 DUMMY_1:    .BLKB   2
                                      00006             .BLKB   2
                                      00008 FAIL_PASSWORD_BUFF:
                                                        .BLKB   31
                                      00027             .BLKB   1
                                      00028 TERM_BUFF:
                                                        .BLKB   8
                                      00030 PHY_TERM_BUFF:
                                                        .BLKB   8
                                      00038 CLU_TERM_BUFF:
                                                        .BLKB   16
                                      00048 SYS$INPUT_BUFFER:
                                                        .BLKB   128
                                      000C8 SYS$OUTPUT_BUFFER:
```

```
                                .BLKB   128
                00148 SYS$ERROR_BUFFER:
                                .BLKB   128
00000002        001C8 RUNDOWN_LIST:
                                .LONG   2                                       ;
00000000V       001CC           .ADDRESS RUNDOWN_CLI                            ;
00000000        001D0           .LONG   0                                       ;

                                .PSECT  $GLOBAL$,NOEXE,2

                00000 PCB_STS::
                                .BLKB   4
                00004 JOB_TYPE::
                                .BLKB   4
              00 00008 TERMINAL_DEVICE::
                                .BYTE   0                                       ;
                00009           .BLKB   3
                0000C DEV_CHAR_2::
                                .BLKB   4
                00010 DEV_DEP_2::
                                .BLKB   4
              00 00014 SUBPROCESS::
                                .BYTE   0                                       ;
              00 00015 IMAGE_ACTIVATE::
                                .BYTE   0                                       ;
              00 00016 BREAK_ATTEMPT::
                                .BYTE   0                                       ;
                00017           .BLKB   1
                00018 DUMMY::   .BLKB   1
                00019           .BLKB   3
                0001C UAF_FAB::
                                .BLKB   80
                0006C UAF_RAB::
                                .BLKB   68
                000B0 UAF_RECORD::
                                .BLKB   4
                000B4 ORG_USERNAME::
                                .BLKB   32
                000D4 PARENT_PID::
                                .BLKB   4
00000000        000D8 CREATOR_USERNAME::
                                .LONG   0
00000000'       000DC           .ADDRESS ORG_USERNAME                           ;
00000000        000E0 FAIL_PASSWORD::
                                .LONG   0
00000000'       000E4           .ADDRESS FAIL_PASSWORD_BUFF                     ;
00000000        000E8 TERM_NAME::
                                .LONG   0
00000000'       000EC           .ADDRESS TERM_BUFF                              ;
00000000        000F0 PHY_TERM_NAME::
                                .LONG   0
00000000'       000F4           .ADDRESS PHY_TERM_BUFF                          ;
00000000        000F8 CLU_TERM_NAME::
                                .LONG   0
00000000'       000FC           .ADDRESS CLU_TERM_BUFF                          ;
                00100 SYS$INPUT_ATTR::
                                .BLKB   4
```

```
00000080  00104 SYS$INPUT::
                        .LONG   128
00000000' 00108         .ADDRESS SYS$INPUT_BUFFER
          0010C SYS$OUTPUT_ATTR::
                        .BLKB   4
00000080  00110 SYS$OUTPUT::
                        .LONG   128
00000000' 00114         .ADDRESS SYS$OUTPUT_BUFFER
          00118 SYS$ERROR_ATTR::
                        .BLKB   4
00000080  0011C SYS$ERROR::
                        .LONG   128
00000000' 00120         .ADDRESS SYS$ERROR_BUFFER

                        .EXTRN  OPEN_INPUT, CLOSE_INPUT
                        .EXTRN  OPEN_OUTPUT, CLOSE_OUTPUT
                        .EXTRN  SET_ACCOUNT, INIT_INTERACTIVE
                        .EXTRN  CHECK_CONNECTION
                        .EXTRN  ANNOUNCE, INIT_BATCH
                        .EXTRN  INIT_NETWORK, INIT_USER
                        .EXTRN  INIT_CLI, EXECUTE_CLI
                        .EXTRN  CREATE_LOGICAL, WRITE_FAO
                        .EXTRN  WRITE_OUTPUT, TERMINATE_BATCH
                        .EXTRN  SYS$SETDDIR, LGI$CHECK_PASS
                        .EXTRN  LGI$SEARCHUSER, SECURITY_AUDIT
                        .EXTRN  CIA_SCAN, LGI$CMSUPR
                        .EXTRN  ASCIC_DAY_OF_WEEK
                        .EXTRN  LIB$DAY_OF_WEEK
                        .EXTRN  LIB$HOUR_OF_DAY
                        .EXTRN  LIB$GET_VM, WRITE_OUTPUT_STATUS
                        .EXTRN  INPUT_FAB, INPUT_NAM
                        .EXTRN  OUTPUT_FAB, OUTPUT_RAB
                        .EXTRN  OUTPUT_NAM, DISK_NAME
                        .EXTRN  CTL$GL_CREPRC_FLAGS
                        .EXTRN  CTL$AG_CLIMAGE, CTL$AG_CLIDATA
                        .EXTRN  EXE$GL_DYNAMIC_FLAGS
                        .EXTRN  EXE$V_BRK_DISUSER
                        .EXTRN  CLI$_PRESENT, CLI$_NEGATED
                        .EXTRN  LGI$_EVADE, LGI$_LOGDISABL
                        .EXTRN  LGI$_EXQUOTA, LGI$_NOTVALID
                        .EXTRN  LGI$_FILEACC, LGI$_USERAUTH
                        .EXTRN  LGI$_USEREXC, LGI$_ACNTEXC
                        .EXTRN  LGI$_BADHOUR, LGI$_BADDAY
                        .EXTRN  LGI$_RESTRICT, LGI$_INPUTERR
                        .EXTRN  LGI$_ACNTEXPIR, LGI$_PWDEXPIR
                        .EXTRN  LGI$_NOSUCHUSER
                        .EXTRN  LGI$_INVPWD, SYS$GETJPIW
                        .EXTRN  SYS$GETDVIW

                        .PSECT  $CODE$,NOWRT,2

                  03FC 00000 START:     .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9            ; 0919
59 00000000G   00   9E 00002            MOVAB   SYS$GETDVIW, R9                         ;
58      0000'  CF   9E 00009            MOVAB   TERM_BUFF, R8                           ;
57 00000000G   00   9E 0000E            MOVAB   PPD, R7                                 ;
56      0000'  CF   9E 00015            MOVAB   PHY_TERM_NAME, R6                       ;
5E        B4   AE   9E 0001A            MOVAB   -76(SP), SP                             ;
```

```
                        6D      0000V   CF  9E 0001E          MOVAB   HANDLER, (FP)                                    ; 0943
                        50        0C    AE  9E 00023          MOVAB   ITEM_LIST, $$ITMBLKPTR                          ; 0964
                        80 02020020     8F  D0 00027          MOVL    #33685536, ($$ITMBLKPTR)+                       ;
                        80        C4    A6  9E 0002E          MOVAB   ORG_USERNAME, ($$ITMBLKPTR)+                    ;
                        80        E8    A6  9E 00032          MOVAB   CREATOR_USERNAME, ($$ITMBLKPTR)+                ;
                        80 03050004     8F  D0 00036          MOVL    #50659332, ($$ITMBLKPTR)+                       ;
                        80      FF10    C6  9E 0003D          MOVAB   PCB_STS, ($$ITMBLKPTR)+                         ;
                                        80  D4 00042          CLRL    ($$ITMBLKPTR)+                                  ;
                        80 03230004     8F  D0 00044          MOVL    #52625412, ($$ITMBLKPTR)+                       ;
                        80      FF14    C6  9E 0004B          MOVAB   JOB_TYPE, ($$ITMBLKPTR)+                        ;
                                        80  D4 00050          CLRL    ($$ITMBLKPTR)+                                  ;
                        80 03030004     8F  D0 00052          MOVL    #50528260, ($$ITMBLKPTR)+                       ;
                        80        E4    A6  9E 00059          MOVAB   PARENT_PID, ($$ITMBLKPTR)+                      ;
                                        80  7C 0005D          CLRQ    ($$ITMBLKPTR)+                                  ;
                              FF10      C6  7C 0005F          CLRQ    PCB_STS                                         ; 0966
                                E4      A6  D4 00063          CLRL    PARENT_PID                                      ; 0968
        20              20      6E          00  2C 00066      MOVC5   #0, (SP), #32, #32, ORG_USERNAME                ; 0969
                                C4      A6      0006B
                                        7E  7C 0006D          CLRQ    -(SP)                                           ; 0970
                                        7E  D4 0006F          CLRL    -(SP)                                           ;
                                18      AE  9F 00071          PUSHAB  ITEM_LIST                                       ;
                                        7E  7C 00074          CLRQ    -(SP)                                           ;
                                        7E  D4 00076          CLRL    -(SP)                                           ;
                    00000000G   00          07  FB 00078      CALLS   #7, SYS$GETJPIW                                 ;
                                01          50  E8 0007F      BLBS    STATUS, 1$                                      ;
                                            04  00082          RET                                                     ;
                                            50  D4 00083 1$:   CLRL    R0                                              ; 0972
                                E4      A6      D5 00085      TSTL    PARENT_PID                                      ;
                                            02  13 00088      BEQL    2$                                              ;
                                            50  D6 0008A      INCL    R0                                              ;
                FF24        C6              50  90 0008C 2$:   MOVB    R0, SUBPROCESS                                  ;
                            50        0C    AE  9E 00091      MOVAB   ITEM_LIST, $$ITMBLKPTR                          ; 0993
                            80 00040004     8F  D0 00095      MOVL    #262T48, ($$ITMBLKPTR)+                         ;
                            80              6E  9E 0009C      MOVAB   DEV_CLASS, ($$ITMBLKPTR)+                       ;
                                            80  D4 0009F      CLRL    ($$ITMBLKPTR)+                                  ;
                            80 00E60004     8F  D0 000A1      MOVL    #15073284, ($$ITMBLKPTR)+                       ;
                            80        FF1C  C6  9E 000A8      MOVAB   DEV_CHAR_2, ($$ITMBLKPTR)+                      ;
                                            80  D4 000AD      CLRL    ($$ITMBLKPTR)+                                  ;
                            80 001C0004     8F  D0 000AF      MOVL    #1835012, ($$ITMBLKPTR)+                        ;
                            80        FF20  C6  9E 000B6      MOVAB   DEV_DEP_2, ($$ITMBLKPTR)+                       ;
                                            80  D4 000BB      CLRL    ($$ITMBLKPTR)+                                  ;
                            80 00200008     8F  D0 000BD      MOVL    #2097160, ($$ITMBLKPTR)+                        ;
                            80              68  9E 000C4      MOVAB   TERM_BUFF, ($$ITMBLKPTR)+                       ;
                            80        F8    A6  9E 000C7      MOVAB   TERM_NAME, ($$ITMBLKPTR)+                       ;
                            80 01120008     8F  D0 000CB      MOVL    #17956872, ($$ITMBLKPTR)+                       ;
                            80        08    A8  9E 000D2      MOVAB   PHY_TERM_BUFF, ($$ITMBLKPTR)+                   ;
                            80              66  9E 000D6      MOVAB   PHY_TERM_NAME, ($$ITMBLKPTR)+                   ;
                                            80  D4 000D9      CLRL    ($$ITMBLKPTR)+                                  ;
                                            6E  D4 000DB      CLRL    DEV_CLASS                                       ; 0995
                                            7E  7C 000DD      CLRQ    -(SP)                                           ; 1004
                                            7E  7C 000DF      CLRQ    -(SP)                                           ;
                                        1C  AE  9F 000E1      PUSHAB  ITEM_LIST                                       ;
                                            67  B5 000E4      TSTW    PPD                                             ;
                                            07  12 000E6      BNEQ    3$                                              ;
                            50      0000'   CF  9E 000E8      MOVAB   P.AAA, R0                                       ;
                                            0E  11 000ED      BRB     4$                                              ;
                    18      AE        28    A7  9A 000EF 3$:   MOVZBL  PPD+40, DEVNAM_DESC                             ;
```

```
                1C    AE        29    A7  9E 000F4          MOVAB   PPD+41, DEVNAM_DESC+4                   ;
                      50        18    AE  9E 000F9          MOVAB   DEVNAM_DESC, R0                         ;
                                      50  DD 000FD 4$:      PUSHL   R0                                      ;
                                      7E  7C 000FF          CLRQ    -(SP)                                   ;
                      69              08  FB 00101          CALLS   #8, SYS$GETDVIW                         ;
                      45              50  E9 00104          BLBC    R0, 6$                                  ;
          00000042    8F              6E  D1 00107          CMPL    DEV_CLASS, #66                          ; 1005
                                      3C  12 0010E          BNEQ    6$                                      ;
              FF18    C6              01  90 00110          MOVB    #1, TERMINAL_DEVICE                     ; 1008
                                      66  D5 00115          TSTL    PHY_TERM_NAME                           ; 1009
                                      0C  12 00117          BNEQ    5$                                      ;
                      50        F8    A6  D0 00119          MOVL    TERM_NAME, R0                           ; 1010
                      66              50  D0 0011D          MOVL    R0, PHY_TERM_NAME                       ;
          08  A8                     50  28 00120          MOVC3   R0, TERM_BUFF, PHY_TERM_BUFF            ;
                      50        0C    AE  9E 00125 5$:      MOVAB   ITEM_LIST, $$ITMBLKPTR                  ; 1016
                      80 00E80010     8F  D0 00129          MOVL    #15204368, ($$ITMBLKPTR)+               ;
                      80        10    A8  9E 00130          MOVAB   CLU_TERM_BUFF, ($$ITMBLKPTR)+           ;
                      80        08    A6  9E 00134          MOVAB   CLU_TERM_NAME, ($$ITMBLKPTR)+           ;
                                      80  D4 00138          CLRL    ($$ITMBLKPTR)+                          ;
                                      7E  7C 0013A          CLRQ    -(SP)                                   ; 1018
                                      7E  7C 0013C          CLRQ    -(SP)                                   ;
                      1C              AE  9F 0013E          PUSHAB  ITEM_LIST                               ;
                                      56  DD 00141          PUSHL   R6                                      ;
                                      7E  7C 00143          CLRQ    -(SP)                                   ;
                      69              08  FB 00145          CALLS   #8, SYS$GETDVIW                         ;
                      11              50  E8 00148          BLBS    STATUS, 7$                              ;
                                          04 0014B          RET                                             ;
                          FF18        C6  94 0014C 6$:      CLRB    TERMINAL_DEVICE                         ; 1022
                          FF1C        C6  7C 00150          CLRQ    DEV_CHAR_2                              ; 1023
                            F8        A6  D4 00154          CLRL    TERM_NAME                               ; 1025
                                      66  D4 00157          CLRL    PHY_TERM_NAME                           ; 1026
                            08        A6  D4 00159          CLRL    CLU_TERM_NAME                           ; 1027
              0000V         CF          00  FB 0015C 7$:      CALLS   #0, CLEAR_SYS_RV                    ; 1041
                                      50  D4 00161          CLRL    R0                                      ; 1049
                                      67  B5 00163          TSTW    PPD                                     ;
                                      02  12 00165          BNEQ    8$                                      ;
                                      50  D6 00167          INCL    R0                                      ;
                D8    A8              50  90 00169 8$:      MOVB    R0, DOING_LOGIN                         ;
                      07              50  E8 0016D          BLBS    R0, 9$                                  ;
              0000V   CF              00  FB 00170          CALLS   #0, LOGOUT                              ; 1051
                                      05  11 00175          BRB     10$                                     ;
              0000V   CF              00  FB 00177 9$:      CALLS   #0, LOGIN                               ; 1053
                      50              01  D0 0017C 10$:     MOVL    #1, R0                                  ; 1055
                                          04 0017F          RET                                             ; 1057

; Routine Size:  384 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  636          1058  1 ROUTINE login: NOVALUE =
;  637          1059  1
;  638          1060  1 !---
;  639          1061  1 !
;  640          1062  1 !       Determine the username and initialize all process context
;  641          1063  1 !       to correspond to the user authorization limits and quotas.
;  642          1064  1 !
;  643          1065  1 ! Inputs:
;  644          1066  1 !
;  645          1067  1 !       None
;  646          1068  1 !
;  647          1069  1 ! Outputs:
;  648          1070  1 !
;  649          1071  1 !       routine = status (already signaled)
;  650          1072  1 !---
;  651          1073  1
;  652          1074  2 BEGIN
;  653          1075  2
;  654          1076  2 EXTERNAL LITERAL
;  655          1077  2     ctl$c_clidatasz;                    ! Size of storage area in P1 space
;  656          1078  2
;  657          1079  2 LOCAL
;  658          1080  2     status,
;  659          1081  2     time:       VECTOR [2],             ! Buffer for current time
;  660          1082  2     lgi:        REF BBLOCK;             ! Address of LGI area
;  661          1083  2
;  662          1084  2 BIND
;  663          1085  2     clireg = ppd [ppd$q_clireg]: VECTOR; ! Reference as 2 longwords
;  664          1086  2
;  665          1087  2 !
;  666          1088  2 ! Change the page protection on the PPD structure to allow user mode
;  667          1089  2 ! write access, so that this program may run primarily in user mode.
;  668          1090  2 !
;  669        P 1091  2 status = $CMEXEC(ROUTIN = set_ppd_prot, ! Set PPD page protection
;  670          1092  2                  ARGLST = prt$c_uw);
;  671          1093  2 IF NOT .status THEN SIGNAL_STOP (.status);
;  672          1094  2
;  673          1095  2 !
;  674          1096  2 ! Translate SYS$INPUT, SYS$OUTPUT, and SYS$ERROR.
;  675          1097  2 !
;  676          1098  2
;  677          1099  3 BEGIN
;  678          1100  3
;  679          1101  3 LOCAL
;  680          1102  3     trnlnm_item_list : BLOCK[2*3+1,LONG];       ! Item list for 2 items
;  681          1103  3
;  682          1104  3 trnlnm_item_list[0, 0,16,0] = 4;
;  683          1105  3 trnlnm_item_list[0,16,16,0] = lnm$_attributes;  ! Fetch name's attributes
;  684          1106  3 trnlnm_item_list[1, 0,32,0] = sys$input_attr;
;  685          1107  3 trnlnm_item_list[2, 0,32,0] = 0;
;  686          1108  3 trnlnm_item_list[3, 0,16,0] = .sys$input[0];
;  687          1109  3 trnlnm_item_list[3,16,16,0] = lnm$_string;      ! Fetch name's value string
;  688          1110  3 trnlnm_item_list[4, 0,32,0] = .sys$input[1];
;  689          1111  3 trnlnm_item_list[5, 0,32,0] = sys$input[0];
;  690          1112  3 trnlnm_item_list[6, 0,32,0] = 0;
;  691          1113  3
;  692        P 1114  3 status = $TRNLNM(TABNAM = %ASCID 'LNM$PROCESS_TABLE', ! Translate SYS$INPUT
```

```
;  693        P 1115  3                    LOGNAM = %ASCID 'SYS$INPUT',
;  694          1116  3                    ITMLST = trnlnm_item_list);
;  695          1117  3 IF NOT .status
;  696          1118  3 THEN SIGNAL_STOP(.status);
;  697          1119  3
;  698          1120  3 trnlnm_item_list[1,0,32,0] = sys$output_attr;
;  699          1121  3 trnlnm_item_list[3,0,16,0] = .sys$output[0];
;  700          1122  3 trnlnm_item_list[4,0,32,0] = .sys$output[1];
;  701          1123  3 trnlnm_item_list[5,0,32,0] = sys$output[0];
;  702          1124  3
;  703        P 1125  3 status = $TRNLNM(TABNAM = %ASCID 'LNM$PROCESS_TABLE', ! Translate SYS$OUTPUT
;  704        P 1126  3                  LOGNAM = %ASCID 'SYS$OUTPUT',
;  705          1127  3                  ITMLST = trnlnm_item_list);
;  706          1128  3 IF NOT .status
;  707          1129  3 THEN SIGNAL_STOP(.status);
;  708          1130  3
;  709          1131  3 trnlnm_item_list[1,0,32,0] = sys$error_attr;
;  710          1132  3 trnlnm_item_list[3,0,16,0] = .sys$error[0];
;  711          1133  3 trnlnm_item_list[4,0,32,0] = .sys$error[1];
;  712          1134  3 trnlnm_item_list[5,0,32,0] = sys$error[0];
;  713          1135  3
;  714        P 1136  3 status = $TRNLNM(TABNAM = %ASCID 'LNM$PROCESS_TABLE', ! Translate SYS$ERROR
;  715        P 1137  3                  LOGNAM = %ASCID 'SYS$ERROR',
;  716          1138  3                  ITMLST = trnlnm_item_list);
;  717          1139  3 IF NOT .status                                  ! If no translation,
;  718          1140  3 OR .status EQL ss$_notran
;  719          1141  3 THEN
;  720          1142  3     sys$error[0] = 0;                           !  then make a null string
;  721          1143  3
;  722          1144  2 END;
;  723          1145  2
;  724          1146  2 !
;  725          1147  2 ! Inititalize the process permanent data region
;  726          1148  2 !
;  727          1149  2 ppd [ppd$w_size] = ppd$c_length;          ! Initialize with static length
;  728          1150  2 lgi = ppd + ppd$c_length;                ! Allocate LGI area just after PPD
;  729          1151  2 ppd [ppd$l_lgi] = .lgi;                   ! Store pointer in PPD
;  730          1152  2 clireg [1] = .lgi + lgi$c_length;         ! Allocate CLI storage after LGI
;  731          1153  2 clireg [0] = ctl$c_clidatasz - ppd$c_length - lgi$c_length;
;  732          1154  2
;  733          1155  2 !
;  734          1156  2 ! Store the original name of SYS$INPUT in the PPD region
;  735          1157  2 !
;  736          1158  2 VECTOR [ppd[ppd$t_filename], 0; ,BYTE] = .sys$input[0] + 1;
;  737          1159  2 CH$MOVE(.sys$input[0],
;  738          1160  2          .sys$input[1],
;  739          1161  2          ppd[ppd$t_filename] + 1);
;  740          1162  2
;  741          1163  2 !
;  742          1164  2 ! Determine if the job is interactive.
;  743          1165  2 ! If not interactive, set the mode bit.
;  744          1166  2 !
;  745          1167  2 IF NOT .ctl$gl_creprc_flags[prc$v_inter]          ! If not interactive,
;  746          1168  2 THEN ppd [ppd$v_mode] = true;                    ! mark non-interactive job
;  747          1169  2
;  748          1170  2 !
;  749          1171  2 ! Change the STARTUP process's account name to the special start up
```

```
;  750          1172  2 ! account name (a binary null followed by "<start>") in order to
;  751          1173  2 ! generate a SYSINIT accounting record.
;  752          1174  2 !
;  753          1175  2 ! The STARTUP process is detected by:
;  754          1176  2 !
;  755          1177  2 !       Not a subprocess
;  756          1178  2 !       No terminal
;  757          1179  2 !       Username is "SYSTEM"
;  758          1180  2 !       Account name is all binary nulls
;  759          1181  2 !
;  760          1182  3 BEGIN
;  761          1183  3 BUILTIN
;  762          1184  3     SKPC;
;  763          1185  3 EXTERNAL
;  764          1186  3     ctl$t_username,
;  765          1187  3     ctl$t_account;
;  766          1188  3 IF NOT .subprocess
;  767          1189  3 AND NOT .terminal_device
;  768          1190  3 AND CH$EQL(jib$s_username, ctl$t_username, 6, UPLIT BYTE('SYSTEM'), ' ')
;  769          1191  3 AND NOT SKPC(%REF(0), %REF(jib$s_account), ctl$t_account)
;  770        P 1192  3 THEN $CMKRNL(ROUTIN = set_account,
;  771          1193  3                ARGLST = %ASCID %STRING(%CHAR(0), '<start>'));
;  772          1194  2 END;
;  773          1195  2
;  774          1196  2 !
;  775          1197  2 ! Perform initializations specific to the type of job.
;  776          1198  2 !
;  777          1199  2 ! The following special account names (all starting with a binary
;  778          1200  2 ! null) are set in order to generate correct LOGFAIL accounting
;  779          1201  2 ! records if there is an authorization failure:
;  780          1202  2 !
;  781          1203  2 !       <batch> Batch job login failure
;  782          1204  2 !       <det>   Detached process login failure
;  783          1205  2 !       <login> Interactive login failure
;  784          1206  2 !       <net>   Network login failure
;  785          1207  2 !
;  786          1208  2 SELECTONE true
;  787          1209  2 OF
;  788          1210  2     SET
;  789          1211  2     !
;  790          1212  2     ! If batch job, ask job controller for job parameters
;  791          1213  2     !
;  792          1214  2     [.ctl$gl_creprc_flags[prc$v_batch]]:
;  793          1215  3         BEGIN
;  794          1216  3         $CMKRNL(ROUTIN = clear_creator_cli);
;  795        P 1217  3         $CMKRNL(ROUTIN = set_account,
;  796          1218  3                 ARGLST = %ASCID %STRING(%CHAR(0), '<batch>'));
;  797          1219  3         init_batch();
;  798          1220  2         END;
;  799          1221  2
;  800          1222  2     !
;  801          1223  2     ! If network job, process NETACP parameters
;  802          1224  2     !
;  803          1225  2     [.ctl$gl_creprc_flags[prc$v_netwrk]]:
;  804          1226  3         BEGIN
;  805          1227  3         IF NOT .ctl$gl_creprc_flags[prc$v_nouaf]
;  806          1228  3         THEN
```

```
;  807          1229  4              BEGIN
;  808          1230  4              $CMKRNL(ROUTIN = clear_creator_cli);
;  809        P 1231  4              $CMKRNL(ROUTIN = set_account,
;  810          1232  4                      ARGLST = %ASCID %STRING(%CHAR(0), '<net>'));
;  811          1233  3              END;
;  812          1234  3          init_network();
;  813          1235  2          END;
;  814          1236  2
;  815          1237  2      !
;  816          1238  2      ! If sub-process, open SYS$INPUT and SYS$OUTPUT.
;  817          1239  2      ! P1 cells will be used later on to get the CLI name, CLI table
;  818          1240  2      ! name, and UAF flags.
;  819          1241  2      ! Note that this test must preceed the following two.
;  820          1242  2      !
;  821          1243  2      [.subprocess<0,1>]:
;  822          1244  3          BEGIN
;  823          1245  3          $CMEXEC(ROUTIN = open_output);  ! Create output file first so that
;  824          1246  3                                          !  error messages can be written
;  825          1247  3          $CMEXEC(ROUTIN = open_input);   ! Create input file
;  826          1248  2          END;
;  827          1249  2
;  828          1250  2      !
;  829          1251  2      ! If interactive job initiated by unsolicited input from a terminal,
;  830          1252  2      ! prompt the terminal for job parameters, and check for job quotas.
;  831          1253  2      !
;  832          1254  2      [.ctl$gl_creprc_flags[prc$v_inter]
;  833          1255  2       and not .ctl$gl_creprc_flags[prc$v_nopassword]]:
;  834          1256  3          BEGIN
;  835          1257  3          $CMKRNL(ROUTIN = clear_creator_cli);
;  836        P 1258  3          $CMKRNL(ROUTIN = set_account,
;  837          1259  3                  ARGLST = %ASCID %STRING(%CHAR(0), '<login>'));
;  838          1260  3          init_interactive();
;  839          1261  3          check_connection();
;  840          1262  3          check_job_quotas();
;  841          1263  2          END;
;  842          1264  2
;  843          1265  2      !
;  844          1266  2      ! Any other process is considered a detached one.
;  845          1267  2      !
;  846          1268  2      [OTHERWISE]:
;  847          1269  3          BEGIN
;  848          1270  3          IF NOT .ctl$gl_creprc_flags[prc$v_nouaf]
;  849          1271  3          THEN
;  850          1272  4              BEGIN
;  851          1273  4              $CMKRNL(ROUTIN = clear_creator_cli);
;  852        P 1274  4              $CMKRNL(ROUTIN = set_account,
;  853          1275  4                      ARGLST = %ASCID %STRING(%CHAR(0), '<det>'));
;  854          1276  4              get_uafrec();
;  855          1277  4              IF .uaf_record EQL 0
;  856          1278  4              THEN SIGNAL_STOP(lgi$_userauth);    ! signal fatal error
;  857          1279  3              END;
;  858          1280  2          END;
;  859          1281  2      TES;
;  860          1282  2
;  861          1283  2 !
;  862          1284  2 ! Check user job limits based on current process counts
;  863          1285  2 !
```

```
;  864     1286  2 IF .uaf_record NEQ 0                    ! If a UAF record
;  865     1287  2 THEN
;  866     1288  2   IF .uaf_record [uaf$w_maxjobs] NEQ 0  ! with job limits
;  867     1289  2   OR .uaf_record [uaf$w_maxacctjobs] NEQ 0
;  868     1290  2 THEN
;  869     1291  3     BEGIN                                ! check limits now, at raised IPL
;  870     1292  3     status = $CMKRNL(ROUTIN = check_user_quotas);
;  871     1293  3     IF NOT .status
;  872     1294  3     THEN SIGNAL_STOP(.status);           ! signal fatal error
;  873     1295  2     END;
;  874     1296  2
;  875     1297  2 !
;  876     1298  2 ! Check account expiration, password expriation, user hourly restrictions,
;  877     1299  2 ! DISUSER flag, and terminal line types. All these checks are waived for
;  878     1300  2 ! the system manager logging in on the console terminal.
;  879     1301  2 !
;  880     1302  2 IF .uaf_record NEQ 0                    ! If a UAF record
;  881     1303  2 THEN
;  882     1304  3     BEGIN
;  883     1305  3     $GETTIM (TIMADR = time);             ! Get current time
;  884     1306  4     IF  (
;  885     1307  4         CH$NEQ(uaf$s_username, uaf_record [uaf$t_username],
;  886     1308  4                6, UPLIT BYTE('SYSTEM'), ' ')
;  887     1309  4       OR
;  888     1310  4         CH$NEQ(.phy_term_name [0], .phy_term_name [1],
;  889     1311  4                6, UPLIT BYTE('_OPA0:'))
;  890     1312  4         )
;  891     1313  3     THEN
;  892     1314  4         BEGIN
;  893     1315  5         IF  (                            ! Check password expiration
;  894     1316  5             .uaf_record [uaf$v_pwd_expired]
;  895     1317  5           OR
;  896     1318  5             .uaf_record [uaf$v_pwd2_expired]
;  897     1319  5             )
;  898     1320  4         THEN SIGNAL_STOP (lgi$_pwdexpir);
;  899     1321  4
;  900     1322  4         IF .uaf_record[uaf$v_disacnt]    ! Check DISUSER flag
;  901     1323  4         THEN SIGNAL_STOP (lgi$_notvalid);
;  902     1324  4         check_account_expiration(time); ! Check account expiration
;  903     1325  4         check_user_hours(time);          ! Check hourly restrictions if any
;  904     1326  3         END;
;  905     1327  2     END;
;  906     1328  2
;  907     1329  2 !
;  908     1330  2 ! If interactive login write a messages announcing successful login.
;  909     1331  2 !
;  910     1332  2 IF .ctl$gl_creprc_flags[prc$v_inter]
;  911     1333  2 AND NOT .ctl$gl_creprc_flags[prc$v_nopassword]
;  912     1334  2 AND .uaf_record NEQ 0
;  913     1335  2 THEN
;  914     1336  2     announce();
;  915     1337  2
;  916     1338  2 !
;  917     1339  2 ! Initialize the CLI image into P1 space
;  918     1340  2 !
;  919     1341  2
;  920     1342  2 IF NOT .image_activate                   ! If not activating a single image,
```

```
;  921     1343  2 THEN
;  922     1344  2     init_cli();                            ! Initialize CLI image
;  923     1345  2
;  924     1346  2 !
;  925     1347  2 ! Set the process quotas, privileges and UIC from the UAF record
;  926     1348  2 !
;  927     1349  2
;  928     1350  2 IF .uaf_record NEQ 0                       ! If UAF record is valid,
;  929     1351  2     AND NOT .ctl$gl_creprc_flags[prc$v_nouaf] ! and uaf init. enabled,
;  930     1352  2     AND NOT .subprocess                    ! or if this is not a sub-process,
;  931     1353  2 THEN
;  932     1354  3     BEGIN
;  933     1355  3     lgi [lgi$l_origuic] = .uaf_record [uaf$l_uic]; ! Save original UIC
;  934     1356  3     init_user();                           ! Initialize user context
;  935     1357  2     END;
;  936     1358  2
;  937     1359  2 !
;  938     1360  2 ! Open the input and output files under the user's UIC and privileges.
;  939     1361  2 !
;  940     1362  2
;  941     1363  2 IF .input_fab [fab$w_ifi] EQL 0            ! If not already done,
;  942     1364  2 THEN
;  943     1365  3     BEGIN
;  944     1366  3     $CMEXEC(ROUTIN = open_output);         ! Create output file first so that
;  945     1367  3                                            ! error messages can be written
;  946     1368  3     $CMEXEC(ROUTIN = open_input);          ! Create input file
;  947     1369  2     END;
;  948     1370  2
;  949     1371  2 !
;  950     1372  2 ! If the creating process passed SYS$ERROR as something different than
;  951     1373  2 ! SYS$OUTPUT, then the original value of SYS$ERROR is passed as
;  952     1374  2 ! a supervisor mode SYS$ERROR logical name, so that this process
;  953     1375  2 ! can obtain the value.  This is used by the DCL SPAWN command
;  954     1376  2 ! to pass a context argument to the subprocess - but yet it wants
;  955     1377  2 ! the executive mode SYS$ERROR (the permanent one) to be set equal
;  956     1378  2 ! to SYS$OUTPUT (so that we don't get two output streams on SPAWNed
;  957     1379  2 ! jobs).
;  958     1380  2 !
;  959     1381  2
;  960     1382  2 IF .subprocess                                 ! For subprocesses
;  961     1383  2 AND .sys$error [0] NEQ 0                      !  with non-null SYS$ERROR,
;  962     1384  2 AND (H$NEQ(.sys$error [0], .sys$error [1],    ! If original SYS$ERROR
;  963     1385  2              .sys$output [0], .sys$output [1], 0) !  NEQ original SYS$OUTPUT,
;  964     1386  2 THEN
;  965     1387  2     create_logical(%ASCID 'SYS$ERROR',        ! Define supervisor SYS$ERROR
;  966     1388  2                    sys$error,                !   as original SYS$ERROR
;  967     1389  2                    psl$c_super);
;  968     1390  2
;  969     1391  2 !
;  970     1392  2 ! Create SYS$LOGIN, SYS$LOGIN_DEVICE, SYS$SCRATCH logical names
;  971     1393  2 !
;  972     1394  2
;  973     1395  2 IF .uaf_record NEQ 0                        ! If UAF record valid,
;  974     1396  2 THEN
;  975     1397  3     BEGIN
;  976     1398  3     LOCAL
;  977     1399  3         buffer:         VECTOR [80,BYTE],   ! Buffer for SYS$LOGIN
```

```
;  978          1400  3          bufdesc:        VECTOR [2],             ! Descriptor of above buffer
;  979          1401  3          devdesc:        VECTOR [2];             ! Descriptor for device name
;  980          1402  3
;  981          1403  3      IF .disk_name[0] EQL 0                      ! If no disk specified,
;  982          1404  3      THEN                                        ! use the default in the UAF
;  983          1405  4          BEGIN
;  984          1406  4          bufdesc [0] = CH$RCHAR(uaf_record [uaf$t_defdev])
;  985          1407  4                      + CH$RCHAR(uaf_record [uaf$t_defdir]);
;  986          1408  4          bufdesc [1] = buffer;
;  987          1409  4          devdesc [0] = CH$RCHAR(uaf_record [uaf$t_defdev]);
;  988          1410  4          devdesc [1] = buffer;
;  989          1411  4
;  990          1412  4          CH$COPY(CH$RCHAR(uaf_record [uaf$t_defdev]),
;  991          1413  4                  uaf_record [$BYTEOFFSET(uaf$t_defdev) + 1,0,0,0],
;  992          1414  4                  CH$RCHAR(uaf_record [uaf$t_defdir]),
;  993          1415  4                  uaf_record [$BYTEOFFSET(uaf$t_defdir) + 1,0,0,0],
;  994          1416  4              0, %ALLOCATION(buffer), buffer);
;  995          1417  4          END
;  996          1418  3      ELSE                                        ! Otherwise, use the disk
;  997          1419  4          BEGIN                                   ! specified at login
;  998          1420  4          bufdesc [0] = .disk_name[0]
;  999          1421  4                      + CH$RCHAR(uaf_record [uaf$t_defdir]);
; 1000          1422  4          bufdesc [1] = buffer;
; 1001          1423  4          devdesc [0] = .disk_name[0];
; 1002          1424  4          devdesc [1] = buffer;
; 1003          1425  4
; 1004          1426  4          CH$COPY(.disk_name[0],
; 1005          1427  4                  .disk_name[1],
; 1006          1428  4                  CH$RCHAR(uaf_record [uaf$t_defdir]),
; 1007          1429  4                  uaf_record [$BYTEOFFSET(uaf$t_defdir) + 1,0,0,0],
; 1008          1430  4                  0, %ALLOCATION(buffer), buffer);
; 1009          1431  3          END;
; 1010          1432  3
; 1011          1433  4      BEGIN
; 1012          1434  4
; 1013          1435  4      !+
; 1014          1436  4      ! Normally, we wouldn't have to use $CMEXEC to create executive
; 1015          1437  4      ! mode logicals as LOGINOUT is installed with SYSNAM privilege
; 1016          1438  4      ! but, the INIT_USER() routine above has established the user's
; 1017          1439  4      ! authorized privileges, which may not include SYSNAM.
; 1018          1440  4      !-
; 1019          1441  4
; 1020          1442  4      LOCAL
; 1021          1443  4          create_arglist : VECTOR[6];
; 1022          1444  4
; 1023          1445  4      create_arglist[0] = 5;                      ! 5 args to 'create_logical'
; 1024          1446  4      create_arglist[3] = psl$c_exec;             ! Exec mode definitions
; 1025          1447  4      create_arglist[4] = 0;                      ! No attributes
; 1026          1448  4      create_arglist[5] = %ASCID 'LNM$JOB';       ! Use job table
; 1027          1449  4
; 1028          1450  4      create_arglist[1] = %ASCID 'SYS$LOGIN';     ! Define SYS$LOGIN
; 1029          1451  4      create_arglist[2] = bufdesc;
; 1030        P 1452  4      $CMEXEC(ROUTIN = create_logical,
; 1031          1453  4              ARGLST = create_arglist);
; 1032          1454  4
; 1033          1455  4      create_arglist[1] = %ASCID 'SYS$LOGIN_DEVICE'; ! Define SYS$LOGIN_DEVICE
; 1034          1456  4      create_arglist[2] = devdesc;
```

```
;  1035        P 1457  4     $CMEXEC(RCUTIN = create_logical,
;  1036          1458  4             ARGLST = create_arglist);
;  1037          1459  4
;  1038          1460  4     create_arglist[1] = %ASCID 'SYS$SCRATCH';   ! Define SYS$SCRATCH
;  1039          1461  4     create_arglist[2] = bufdesc;
;  1040        P 1462  4     $CMEXEC(ROUTIN = create_logical,
;  1041          1463  4             ARGLST = create_arglist);
;  1042          1464  4
;  1043          1465  3     END;
;  1044          1466  3
;  1045          1467  2     END;
;  1046          1468  2
;  1047          1469  2 !
;  1048          1470  2 ! Update the UAF record.
;  1049          1471  2 !
;  1050          1472  2 IF .uaf_record NEQ 0
;  1051          1473  2 THEN update_uaf_record(time);
;  1052          1474  2
;  1053          1475  2 !
;  1054          1476  2 ! Security audit successful LOGINs
;  1055          1477  2 !
;  1056          1478  2
;  1057          1479  2 security_audit(nsa$k_rectyp_logi);      ! Security audit successful LOGINs
;  1058          1480  2
;  1059          1481  2 !
;  1060          1482  2 ! Transfer control to the command language interpreter
;  1061          1483  2 !
;  1062          1484  2
;  1063          1485  2 IF .image_activate                      ! If activating a single image,
;  1064          1486  2 THEN
;  1065          1487  3     $CMEXEC(ROUTIN = .ctl$ag_climage)   ! then activate the image
;  1066          1488  2 ELSE
;  1067          1489  2     $CMEXEC(ROUTIN = execute_cli);      ! Call the CLI image
;  1068          1490  2
;  1069          1491  2 RETURN true;
;  1070          1492  2
;  1071          1493  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

42 41 54 5F 53 53 45 43 4F 52 50 24 4D 4E 4C  00014 P.AAD:  .ASCII  \LNM$PROCESS_TABLE\<0><0><0>
                                  00 00 00 45 4C  00023
                                                  010E0011  00028 P.AAC:  .LONG   17694737
                                                  00000000' 0002C         .ADDRESS P.AAD
                00 00 00 54 55 50 4E 49 24 53 59 53  00030 P.AAF:  .ASCII  \SYS$INPUT\<0><0><0>
                                                  010E0009  0003C P.AAE:  .LONG   17694729
                                                  00000000' 00040         .ADDRESS P.AAF
42 41 54 5F 53 53 45 43 4F 52 50 24 4D 4E 4C  00044 P.AAH:  .ASCII  \LNM$PROCESS_TABLE\<0><0><0>
                                  00 00 00 45 4C  00053
                                                  010E0011  00058 P.AAG:  .LONG   17694737
                                                  00000000' 0005C         .ADDRESS P.AAH
                00 00 54 55 50 54 55 4F 24 53 59 53  00060 P.AAJ:  .ASCII  \SYS$OUTPUT\<0><0>
                                                  010E000A  0006C P.AAI:  .LONG   17694730
                                                  00000000' 00070         .ADDRESS P.AAJ
42 41 54 5F 53 53 45 43 4F 52 50 24 4D 4E 4C  00074 P.AAL:  .ASCII  \LNM$PROCESS_TABLE\<0><0><0>
```

```
                                        00  00  00  45  4C  00083
                                                    010E0011  00088 P.AAK:  .LONG   17694737
                                                    00000000' 0008C         .ADDRESS P.AAL
        00  00  00  52  4F  52  52  45  24  53  59  53  00090 P.AAN:  .ASCII  \SYS$ERROR\<0><0><0>
                                                    010E0009  0009C P.AAM:  .LONG   17694729
                                                    00000000' 000A0         .ADDRESS P.AAN
                                    4D  45  54  53  59  53  000A4 P.AAO:  .ASCII  \SYSTEM\
                                                            000AA         .BLKB   2
                                3E  74  72  61  74  73  3C  00  000AC P.AAQ:  .ASCII  <0>\<start>\
                                                    010E0008  000B4 P.AAP:  .LONG   17694728
                                                    00000000' 000B8         .ADDRESS P.AAQ
                                3E  68  63  74  61  62  3C  00  000BC P.AAS:  .ASCII  <0>\<batch>\
                                                    010E0008  000C4 P.AAR:  .LONG   17694728
                                                    00000000' 000C8         .ADDRESS P.AAS
                                00  00  3E  74  65  6E  3C  00  000CC P.AAU:  .ASCII  <0>\<net>\<0><0>
                                                    010E0006  000D4 P.AAT:  .LONG   17694726
                                                    00000000' 000D8         .ADDRESS P.AAU
                                3E  6E  69  67  6F  6C  3C  00  000DC P.AAW:  .ASCII  <0>\<login>\
                                                    010E0008  000E4 P.AAV:  .LONG   17694728
                                                    00000000' 000E8         .ADDRESS P.AAW
                                00  00  3E  74  65  64  3C  00  000EC P.AAY:  .ASCII  <0>\<det>\<0><0>
                                                    010E0006  000F4 P.AAX:  .LONG   17694726
                                                    00000000' 000F8         .ADDRESS P.AAY
                                        4D  45  54  53  59  53  000FC P.AAZ:  .ASCII  \SYSTEM\
                                        3A  30  41  50  4F  5F  00102 P.ABA:  .ASCII  \_OPA0:\
        00  00  00  52  4F  52  52  45  24  53  59  53  00108 P.ABC:  .ASCII  \SYS$ERROR\<0><0><0>
                                                    010E0009  00114 P.ABB:  .LONG   17694729
                                                    00000000' 00118         .ADDRESS P.ABC
                                00  42  4F  4A  24  4D  4E  4C  0011C P.ADE:  .ASCII  \LNM$JOB\<0>
                                                    010E0007  00124 P.ABD:  .LONG   17694727
                                                    00000000' 00128         .ADDRESS P.ABE
            00  00  00  4E  49  47  4F  4C  24  53  59  53  0012C P.ABG:  .ASCII  \SYS$LOGIN\<0><0><0>
                                                    010E0009  00138 P.ABF:  .LONG   17694729
                                                    00000000' 0013C         .ADDRESS P.ABG
43  49  56  45  44  5F  4E  49  47  4F  4C  24  53  59  53  00140 P.ABI:  .ASCII  \SYS$LOGIN_DEVICE\
                                                        45  0014F
                                                    010E0010  00150 P.ABH:  .LONG   17694736
                                                    00000000' 00154         .ADDRESS P.ABI
            00  48  43  54  41  52  43  53  24  53  59  53  00158 P.ABK:  .ASCII  \SYS$SCRATCH\<0>
                                                    010E000B  00164 P.ABJ:  .LONG   17694731
                                                    00000000' 00168         .ADDRESS P.ABK

                                                                            .EXTRN  CTL$C_CLIDATASZ
                                                                            .EXTRN  SYS$CMEXEC, SYS$TRNLNM
                                                                            .EXTRN  CTL$T_USERNAME, CTL$T_ACCOUNT
                                                                            .EXTRN  SYS$CMKRNL, SYS$GETTIM

                                                                            .PSECT  $CODE$,NOWRT,2

                                                        0FFC 00000 LOGIN:   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1058
                                    5B      0000'   CF  9E 00002            MOVAB   UAF_RECORD, R11
                                    5E        80    AE  9E 00007            MOVAB   -128(SP), SP
                                                    04  DD 0000B            PUSHL   #4                                          ; 1092
                                                0000V CF  9F 0000D            PUSHAB  SET_PPD_PROT
                                00000000G   00      02  FB 00011            CALLS   #2, SYS$CMEXEC
                                            56      50  D0 00018            MOVL    R0, STATUS
                                            09      56  E8 0001B            BLBS    STATUS, 1$                                  ; 1093
```

```
                                        56  DD 0001E         PUSHL   STATUS                                   ;
               00000000G   00           01  FB 00020         CALLS   #1, LIB$STOP                             ;
                      5C   AE 00030004  8F  D0 00027 1$:     MOVL    #196612, TRNLNM_ITEM_LIST                ; 1104
                      60   AE       50  AB  9E 0002F         MOVAB   SYS$INPUT_ATTR, TRNLNM_ITEM_LIST+4       ; 1106
                                    64  AE  D4 00034         CLRL    TRNLNM_ITEM_LIST+8                       ; 1107
                      68   AE       54  AB  B0 00037         MOVW    SYS$INPUT, TRNLNM_ITEM_LIST+12           ; 1108
                      6A   AE           02  B0 0003C         MOVW    #2, TRNLNM_ITEM_LIST+14                  ; 1109
                      6C   AE       58  AB  D0 00040         MOVL    SYS$INPUT+4, TRNLNM_ITEM_LIST+16         ; 1110
                      70   AE       54  AB  9E 00045         MOVAB   SYS$INPUT, TRNLNM_ITEM_LIST+20           ; 1111
                                    74  AE  D4 0004A         CLRL    TRNLNM_ITEM_LIST+24                      ; 1112
                                    5C  AE  9F 0004D         PUSHAB  TRNLNM_ITEM_LIST                         ; 1116
                                        7E  D4 00050         CLRL    -(SP)                                    ;
                                 0000'  CF  9F 00052         PUSHAB  P.AAE                                    ;
                                 0000'  CF  9F 00056         PUSHAB  P.AAC                                    ;
                                        7E  D4 0005A         CLRL    -(SP)                                    ;
               00000000G   00           05  FB 0005C         CALLS   #5, SYS$TRNLNM                           ;
                           56           50  D0 00063         MOVL    R0, STATUS                               ;
                           09           56  E8 00066         BLBS    STATUS, 2$                               ; 1117
                                        56  DD 00069         PUSHL   STATUS                                   ; 1118
               00000000G   00           01  FB 0006B         CALLS   #1, LIB$STOP                             ;
                      60   AE       5C  AB  9E 00072 2$:     MOVAB   SYS$OUTPUT_ATTR, TRNLNM_ITEM_LIST+4      ; 1120
                      68   AE       60  AB  B0 00077         MOVW    SYS$OUTPUT, TRNLNM_ITEM_LIST+12          ; 1121
                      6C   AE       64  AB  D0 0007C         MOVL    SYS$OUTPUT+4, TRNLNM_ITEM_LIST+16        ; 1122
                      70   AE       60  AB  9E 00081         MOVAB   SYS$OUTPUT, TRNLNM_ITEM_LIST+20          ; 1123
                                    5C  AE  9F 00086         PUSHAB  TRNLNM_ITEM_LIST                         ; 1127
                                        7E  D4 00089         CLRL    -(SP)                                    ;
                                 0000'  CF  9F 0008B         PUSHAB  P.AAI                                    ;
                                 0000'  CF  9F 0008F         PUSHAB  P.AAG                                    ;
                                        7E  D4 00093         CLRL    -(SP)                                    ;
               00000000G   00           05  FB 00095         CALLS   #5, SYS$TRNLNM                           ;
                           56           50  D0 0009C         MOVL    R0, STATUS                               ;
                           09           56  E8 0009F         BLBS    STATUS, 3$                               ; 1128
                                        56  DD 000A2         PUSHL   STATUS                                   ; 1129
               00000000G   00           01  FB 000A4         CALLS   #1, LIB$STOP                             ;
                      60   AE       68  AB  9E 000AB 3$:     MOVAB   SYS$ERROR_ATTR, TRNLNM_ITEM_LIST+4       ; 1131
                      68   AE       6C  AB  B0 000B0         MOVW    SYS$ERROR, TRNLNM_ITEM_LIST+12           ; 1132
                      6C   AE       70  AB  D0 000B5         MOVL    SYS$ERROR+4, TRNLNM_ITEM_LIST+16         ; 1133
                      70   AE       6C  AB  9E 000BA         MOVAB   SYS$ERROR, TRNLNM_ITEM_LIST+20           ; 1134
                                    5C  AE  9F 000BF         PUSHAB  TRNLNM_ITEM_LIST                         ; 1138
                                        7E  D4 000C2         CLRL    -(SP)                                    ;
                                 0000'  CF  9F 000C4         PUSHAB  P.AAM                                    ;
                                 0000'  CF  9F 000C8         PUSHAB  P.AAK                                    ;
                                        7E  D4 000CC         CLRL    -(SP)                                    ;
               00000000G   00           05  FB 000CE         CALLS   #5, SYS$TRNLNM                           ;
                           56           50  D0 000D5         MOVL    R0, STATUS                               ;
                           09           56  E9 000D8         BLBC    STATUS, 4$                               ; 1139
               00000629    8F           56  D1 000DB         CMPL    STATUS, #1577                            ; 1140
                                        03  12 000E2         BNEQ    5$                                       ;
                                    6C  AB  D4 000E4 4$:     CLRL    SYS$ERROR                                ; 1142
               00000000G   00     0168  8F  B0 000E7 5$:     MOVW    #360, PPD                                ; 1149
                           57 00000000G 00  9E 000F0         MOVAB   PPD+360, LGI                             ; 1150
               00000000G   00           57  D0 000F7         MOVL    LGI, PPD+20                              ; 1151
               00000000G   00       18  A7  9E 000FE         MOVAB   24(R7), CLIREG+4                         ; 1152
               00000000G   00 00000000G 8F  D0 00106         MOVL    #CTL$C_CLIDATASZ-384, CLIREG             ; 1153
  00000000G  00       54   AB           01  81 00111         ADDB3   #1, SYS$INPUT, PPD+104                   ; 1158
  00000000G  00       58   BB       54  AB  28 0011A         MOVC3   SYS$INPUT, @SYS$INPUT+4, PPD+105         ; 1161
            07 00000000G   00           02  E0 00124         BBS     #2, CTL$GL_CREPRC_FLAGS+1, 6$            ; 1167
```

```
                     00000000G  00          02  88 0012C       BISB2   #2, PPD+2                                  ; 1168
                                2E    FF64  CB  E8 00133 6$:   BLBS    SUBPROCESS, 7$                             ; 1188
                                29    FF58  CB  E8 00138       BLBS    TERMINAL_DEVICE, 7$                        ; 1189
        06                20 00000000G  00  0C  2D 0013D       CMPC5   #12, CTL$T_USERNAME, #32, #6, P.AAO        ; 1190
                                     0000'  CF     00146                                                      ;
                                            1B  12 00149       BNEQ    7$                                         ;
           00000000G  00                08  00  3B 0014B       SKPC    #0, #8, CTL$T_ACCOUNT                      ; 1191
                                            11  12 00153       BNEQ    7$                                         ;
                                     0000'  CF  9F 00155       PUSHAB  P.AAP                                      ; 1193
                                 00000000G  00  9F 00159       PUSHAB  SET_ACCOUNT                                ;
                     00000000G  00          02  FB 0015F       CALLS   #2, SYS$CMKRNL                             ;
                  27 00000000G  00          04  E1 00166 7$:   BBC     #4, CTL$GL_CREPRC_FLAGS, 8$                ; 1214
                                            7E  D4 0016E       CLRL    -(SP)                                      ; 1216
                                     0000V  CF  9F 00170       PUSHAB  CLEAR_CREATOR_CLI                          ;
                     00000000G  00          02  FB 00174       CALLS   #2, SYS$CMKRNL                             ;
                                     0000'  CF  9F 0017B       PUSHAB  P.AAR                                      ; 1218
                                 00000000G  00  9F 0017F       PUSHAB  SET_ACCOUNT                                ;
                     00000000G  00          02  FB 00185       CALLS   #2, SYS$CMKRNL                             ;
                     00000000G  00          00  FB 0018C       CALLS   #0, INIT_BATCH                             ; 1219
                                            5A  11 00193       BRB     11$                                        ; 1208
                                 00000000G  00  95 00195 8$:   TSTB    CTL$GL_CREPRC_FLAGS                        ; 1225
                                            2F  18 0019B       BGEQ    10$                                        ;
                  1E 00000000G  00          06  E0 0019D       BBS     #6, CTL$GL_CREPRC_FLAGS, 9$                ; 1227
                                            7E  D4 001A5       CLRL    -(SP)                                      ; 1230
                                     0000V  CF  9F 001A7       PUSHAB  CLEAR_CREATOR_CLI                          ;
                     00000000G  00          02  FB 001AB       CALLS   #2, SYS$CMKRNL                             ;
                                     0000'  CF  9F 001B2       PUSHAB  P.AAT                                      ; 1232
                                 00000000G  00  9F 001B6       PUSHAB  SET_ACCOUNT                                ;
                     00000000G  00          02  FB 001BC       CALLS   #2, SYS$CMKRNL                             ;
                     00000000G  00          00  FB 001C3 9$:   CALLS   #0, INIT_NETWORK                           ; 1234
                                            70  11 001CA       BRB     13$                                        ; 1208
                                20    FF64  CB  E9 001CC 10$:  BLBC    SUBPROCESS, 12$                            ; 1243
                                            7E  D4 001D1       CLRL    -(SP)                                      ; 1245
                                 00000000G  00  9F 001D3       PUSHAB  OPEN_OUTPUT                                ;
                     00000000G  00          02  FB 001D9       CALLS   #2, SYS$CMEXEC                             ;
                                            7E  D4 001E0       CLRL    -(SP)                                      ; 1247
                                 00000000G  00  9F 001E2       PUSHAB  OPEN_INPUT                                 ;
                     00000000G  00          02  FB 001E8       CALLS   #2, SYS$CMEXEC                             ;
                                            4B  11 001EF 11$:  BRB     13$                                        ; 1208
        50 00000000G  00                01  02  EF 001F1 12$:  EXTZV   #2, #1, CTL$GL_CREPRC_FLAGS+1, R0          ; 1255
        51 00000000G  00                01  05  EF 001FA       EXTZV   #5, #1, CTL$GL_CREPRC_FLAGS+1, R1          ;
                                        50  51  CA 00203       BICL2   R1, R0                                     ;
                                        01  50  D1 00206       CMPL    R0, #1                                     ;
                                            33  12 00209       BNEQ    14$                                        ;
                                            7E  D4 0020B       CLRL    -(SP)                                      ; 1257
                                     0000V  CF  9F 0020D       PUSHAB  CLEAR_CREATOR_CLI                          ;
                     00000000G  00          02  FB 00211       CALLS   #2, SYS$CMKRNL                             ;
                                     0000'  CF  9F 00218       PUSHAB  P.AAV                                      ; 1259
                                 00000000G  00  9F 0021C       PUSHAB  SET_ACCOUNT                                ;
                     00000000G  00          02  FB 00222       CALLS   #2, SYS$CMKRNL                             ;
                     00000000G  00          00  FB 00229       CALLS   #0, INIT_INTERACTIVE                       ; 1260
                     00000000G  00          00  FB 00230       CALLS   #0, CHECK_CONNECTION                       ; 1261
                         0000V  CF          00  FB 00237       CALLS   #0, CHECK_JOB_QUOTAS                       ; 1262
                                            3C  11 0023C 13$:  BRB     15$                                        ; 1208
                  34 00000000G  00          06  E0 0023E 14$:  BBS     #6, CTL$GL_CREPRC_FLAGS, 15$               ; 1270
                                            7E  D4 00246       CLRL    -(SP)                                      ; 1273
                                     0000V  CF  9F 00248       PUSHAB  CLEAR_CREATOR_CLI                          ;
```

```
                    00000000G  00          02  FB  0024C          CALLS   #2, SYS$CMKRNL                                   ;
                          0000'  CF         9F  00253          PUSHAB  P.AAX                                           ; 1275
                00000000G  00               00  9F  00257          PUSHAB  SET_ACCOUNT                                     ;
                    00000000G  00           02  FB  0025D          CALLS   #2, SYS$CMKRNL                                  ;
                         0000V  CF          00  FB  00264          CALLS   #0, GET_UAFREC                                  ; 1276
                                            6B  D5  00269          TSTL    UAF_RECORD                                      ; 1277
                                            0D  12  0026B          BNEQ    15$                                             ;
                        00000000G           8F  DD  0026D          PUSHL   #LGI$_USERAUTH                                  ; 1278
                    00000000G  00           01  FB  00273          CALLS   #1, LIB$STOP                                    ;
                               50           6B  D0  0027A  15$:     MOVL    UAF_RECORD, R0                                  ; 1286
                                            28  13  0027D          BEQL    17$                                             ;
                                  0206      C0  B5  0027F          TSTW    518(R0)                                         ; 1288
                                            06  12  00283          BNEQ    16$                                             ;
                                  0208      C0  B5  00285          TSTW    520(R0)                                         ; 1289
                                            1C  13  00289          BEQL    17$                                             ;
                                            7E  D4  0028B  16$:     CLRL    -(SP)                                           ; 1292
                                         0000V  CF  9F  0028D          PUSHAB  CHECK_USER_QUOTAS                               ;
                    00000000G  00           02  FB  00291          CALLS   #2, SYS$CMKRNL                                  ;
                               56           50  D0  00298          MOVL    R0, STATUS                                      ;
                               09           56  E8  0029B          BLBS    STATUS, 17$                                     ; 1293
                                            56  DD  0029E          PUSHL   STATUS                                          ; 1294
                    00000000G  00           01  FB  002A0          CALLS   #1, LIB$STOP                                    ;
                                            6B  D5  002A7  17$:     TSTL    UAF_RECORD                                      ; 1302
                                            63  13  002A9          BEQL    22$                                             ;
                                     78     AE  9F  002AB          PUSHAB  TIME                                            ; 1305
                    00000000G  00           01  FB  002AE          CALLS   #1, SYS$GETTIM                                  ;
                               54           6B  D0  002B5          MOVL    UAF_RECORD, R4                                  ; 1307
06                  20      04  A4          20  2D  002B8          CMPC5   #32, 4(R4), #32, #6, P.AAZ                      ;
                          0000'  CF             002BE                                                                      ;
                                            0C  12  002C1          BNEQ    19$                                             ;
06                  00      44  BB      40  AB  2D  002C3          CMPC5   PHY_TERM_NAME, @PHY_TERM_NAME+4, #0, #6, -      ; 1310
                          0000'  CF             002CA                  P.ABA                                               ;
                                            3F  13  002CD          BEQL    22$                                             ;
06                    01D5  C4              01  E0  002CF  18$:     BBS     #1, 469(R4), 19$                                ; 1316
0D                    01D5  C4              02  E1  002D5          BBC     #2, 469(R4), 20$                                ; 1318
                        00000000G           8F  DD  002DB  19$:     PUSHL   #LGI$_PWDEXPIR                                  ; 1320
                    00000000G  00           01  FB  002E1          CALLS   #1, LIB$STOP                                    ;
                               50           6B  D0  002E8  20$:     MOVL    UAF_RECORD, R0                                  ; 1322
0D                    01D4  C0              04  E1  002EB          BBC     #4, 468(R0), 21$                                ;
                        00000000G           8F  DD  002F1          PUSHL   #LGI$_NOTVALID                                  ; 1323
                    00000000G  00           01  FB  002F7          CALLS   #1, LIB$STOP                                    ;
                                     78     AE  9F  002FE  21$.     PUSHAB  TIME                                            ; 1324
                         0000V  CF          01  FB  00301          CALLS   #1, CHECK_ACCOUNT_EXPIRATION                    ;
                                     78     AE  9F  00306          PUSHAB  TIME                                            ; 1325
                         0000V  CF          01  FB  00309          CALLS   #1, CHECK_USER_HOURS                            ;
13 00000000G  00                            02  E1  0030E  22$:     BBC     #2, CTL$GL_CREPRC_FLAGS+1, 23$                  ; 1332
0B 00000000G  00                            05  E0  00316          BBS     #5, CTL$GL_CREPRC_FLAGS+1, 23$                  ; 1333
                                            6B  D5  0031E          TSTL    UAF_RECORD                                      ; 1334
                                            07  13  00320          BEQL    23$                                             ;
                    00000000G  00           00  FB  00322          CALLS   #0, ANNOUNCE                                    ; 1336
                               07     FF65  CB  E8  00329  23$:     BLBS    IMAGE_ACTIVATE, 24$                             ; 1342
                    00000000G  00           00  FB  0032E          CALLS   #0, INIT_CLI                                    ; 1344
                               50           6B  D0  00335  24$:     MOVL    UAF_RECORD, R0                                  ; 1350
                                            18  13  00338          BEQL    25$                                             ;
10 00000000G  00                            06  E0  0033A          BBS     #6, CTL$GL_CREPRC_FLAGS, 25$                    ; 1351
                               0B     FF64  CB  E8  00342          BLBS    SUBPROCESS, 25$                                 ; 1352
                               67       24  A0  D0  00347          MOVL    36(R0), (LGI)                                   ; 1355
```

```
                                00000000G  00                    00  FB 0034B          CALLS   #0, INIT_USER                                    ; 1356
                                                      00000000G 00  B5 00352 25$:     TSTW    INPUT_FAB+2                                     ; 1363
                                                                1E  12 00358          BNEQ    26$
                                                                7E  D4 0035A          CLRL    -(SP)                                           ; 1366
                                                      00000000G 00  9F 0035C          PUSHAB  OPEN_OUTPUT
                                00000000G  00                     02  FB 00362          CALLS   #2, SYS$CMEXEC
                                                                7E  D4 00369          CLRL    -(SP)                                           ; 1368
                                                      00000000G 00  9F 0036B          PUSHAB  OPEN_INPUT
                                00000000G  00                     02  FB 00371          CALLS   #2, SYS$CMEXEC
                                        21      FF64    CB  E9 00378 26$:     BLBC    SUBPROCESS, 27$                                 ; 1382
                                          6C    AB  D5 0037D          TSTL    SYS$ERROR                                               ; 1383
                                                      1C  13 00380          BEQL    27$
    60  AB              00        70    BB    6C    AB  2D 00382          CMPC5   SYS$ERROR, @SYS$ERROR+4, #0, SYS$OUTPUT, -              ; 1384
                                          64    BB       0038A                        @SYS$OUTPUT+4
                                                      10  13 0038C          BEQL    27$
                                                      02  DD 0038E          PUSHL   #2                                                      ; 1387
                                          6C    AB  9F 00390          PUSHAB  SYS$ERROR
                                        0000'   CF  9F 00393          PUSHAB  P.ABB
                                00000000G  00                     03  FB 00397          CALLS   #3, CREATE_LOGICAL
                                        56              6B  D0 0039E 27$:     MOVL    UAF_RECORD, R6                                  ; 1395
                                                      03  12 003A1          BNEQ    28$
                                                    00E3  31 003A3          BRW     31$
                                        57  00000000G  00  D0 003A6 28$:     MOVL    DISK_NAME, R7                                   ; 1403
                                                      3D  12 003AD          BNEQ    29$
                                        50        74    A6  9A 003AF          MOVZBL  116(R6), R0                                     ; 1407
                                        51      0094    C6  9A 003B3          MOVZBL  148(R6), R1
                                20  AE              50      C1 003B8          ADDL3   R1, R0, BUFDESC
                                    18  AE          74    A6  9A 003BD          MOVZBL  116(R6), DEVDESC                                ; 1409
                                        5A          74    A6  9A 003C2          MOVZBL  116(R6), R10                                    ; 1412
                                        59      0094    C6  9A 003C6          MOVZBL  148(R6), R9                                     ; 1414
                                        58        50    8F  9A 003CB          MOVZBL  #80, R8                                         ; 1415
                                        57        28    AE  9E 003CF          MOVAB   BUFFER, R7
        58              00        75    A6              5A  2C 003D3          MOVC5   R10, 117(R6), #0, R8, (R7)
                                                      67      003D9
                                                      48  18 003DA          BGEQ    30$
                                            57            5A  C0 003DC          ADDL2   R10, R7
                                            58            5A  C2 003DF          SUBL2   R10, R8
    58                  00      0095    C6              59  2C 003E2          MOVC5   R9, 149(R6), #0, R8, (R7)
                                                      67      003E9
                                                      38  11 003EA          BRB     30$                                             ; 1403
                                        50      0094    C6  9A 003EC 29$:     MOVZBL  148(R6), R0                                     ; 1421
                                20  AE  57              50  C1 003F1          ADDL3   R0, R7, BUFDESC
                                    18  AE              57  D0 003F6          MOVL    R7, DEVDESC                                     ; 1423
                                        50  00000000G  00  D0 003FA          MOVL    DISK_NAME+4, R0                                 ; 1427
                                        5A      0094    C6  9A 00401          MOVZBL  148(R6), R10                                    ; 1428
                                        59        50    8F  9A 00406          MOVZBL  #80, R9                                         ; 1429
                                        58        28    AE  9E 0040A          MOVAB   BUFFER, R8
    59              00                60              57  2C 0040E          MOVC5   R7, (R0), #0, R9, (R8)
                                                      68      00413
                                                      0E  18 00414          BGEQ    30$
                                            58            57  C0 00416          ADDL2   R7, R8
                                            59            57  C2 00419          SUBL2   R7, R9
    59              00      0095    C6              5A  2C 0041C          MOVC5   R10, 149(R6), #0, R9, (R8)
                                                      68      00423
                                    24  AE      28    AE  9E 00424 30$:     MOVAB   BUFFER, BUFDESC+4                               ; 1408
                                    1C  AE      28    AE  9E 00429          MOVAB   BUFFER, DEVDESC+4                               ; 1410
                                                6E          05  D0 0042E          MOVL    #5, CREATE_ARGLIST                              ; 1445
```

```
                    0C   AE              01   7D 00431          MOVQ    #1, CREATE_ARGLIST+12                   ; 1446
                    14   AE      0000'   CF   9E 00435          MOVAB   P.ABD, CREATE_ARGLIST+20                ; 1448
                    04   AE      0000'   CF   9E 0043B          MOVAB   P.ABF, CREATE_ARGLIST+4                 ; 1450
                    08   AE        20    AE   9E 00441          MOVAB   BUFDESC, CREATE_ARGLIST+8               ; 1451
                                           5E   DD 00446          PUSHL   SP                                      ; 1453
                         00000000G         00   9F 00448          PUSHAB  CREATE_LOGICAL                          ;
                00000000G   00             02   FB 0044E          CALLS   #2, SYS$CMEXEC                          ;
                    04   AE      0000'   CF   9E 00455          MOVAB   P.ABH, CREATE_ARGLIST+4                 ; 1455
                    08   AE        18    AE   9E 0045B          MOVAB   DEVDESC, CREATE_ARGLIST+8               ; 1456
                                           5E   DD 00460          PUSHL   SP                                      ; 1458
                         00000000G         00   9F 00462          PUSHAB  CREATE_LOGICAL                          ;
                00000000G   00             02   FB 00468          CALLS   #2, SYS$CMEXEC                          ;
                    04   AE      0000'   CF   9E 0046F          MOVAB   P.ABJ, CREATE_ARGLIST+4                 ; 1460
                    08   AE        20    AE   9E 00475          MOVAB   BUFDESC, CREATE_ARGLIST+8               ; 1461
                                           5E   DD 0047A          PUSHL   SP                                      ; 1463
                         00000000G         00   9F 0047C          PUSHAB  CREATE_LOGICAL                          ;
                00000000G   00             02   FB 00482          CALLS   #2, SYS$CMEXEC                          ;
                                           6B   D5 00489 31$:     TSTL    UAF_RECORD                              ; 1472
                                           08   13 0048B          BEQL    32$                                     ;
                                     78    AE   9F 0048D          PUSHAB  TIME                                    ; 1473
                        0000V   CF         01   FB 00490          CALLS   #1, UPDATE_UAF_RECORD                   ;
                                           05   DD 00495 32$:     PUSHL   #5                                      ; 1479
                00000000G   00             01   FB 00497          CALLS   #1, SECURITY_AUDIT                      ;
                            0A     FF65    CB   E9 0049E          BLBC    IMAGE_ACTIVATE, 33$                     ; 1485
                                           7E   D4 004A3          CLRL    -(SP)                                   ; 1487
                         00000000G         00   DD 004A5          PUSHL   CTL$AG_CLIMAGE                          ;
                                           08   11 004AB          BRB     34$                                     ;
                                           7E   D4 004AD 33$:     CLRL    -(SP)                                   ; 1489
                         00000000G         00   9F 004AF          PUSHAB  EXECUTE_CLI                             ;
                00000000G   00             02   FB 004B5 34$:     CALLS   #2, SYS$CMEXEC                          ;
                                           04   004BC             RET                                             ; 1493

; Routine Size:  1213 bytes,    Routine Base:  $CODE$ + 0180
```

```
;  1073          1494  1 ROUTINE logout: NOVALUE =
;  1074          1495  1
;  1075          1496  1 !---
;  1076          1497  1 !
;  1077          1498  1 !       If this is a batch job, an attempt is made to start the
;  1078          1499  1 !       next job step.  For all other jobs, the process is terminated.
;  1079          1500  1 !
;  1080          1501  1 ! Inputs:
;  1081          1502  1 !
;  1082          1503  1 !       None
;  1083          1504  1 !
;  1084          1505  1 ! Outputs:
;  1085          1506  1 !
;  1086          1507  1 !       None
;  1087          1508  1 !---
;  1088          1509  1
;  1089          1510  2 BEGIN
;  1090          1511  2
;  1091          1512  2 !
;  1092          1513  2 ! The routine below is a copy of the CLI$PRESENT routine in CLI$INTERFACE.
;  1093          1514  2 ! This is used so that we call the current CLI's result parse routines, not
;  1094          1515  2 ! the routines linked into us for LOGIN's sake.
;  1095          1516  2 !
;  1096          1517  2
;  1097          1518  2 ROUTINE real_cli$present(name) =
;  1098          1519  3 BEGIN
;  1099          1520  3 EXTERNAL ROUTINE
;  1100          1521  3     lib$get_vm,                       ! Allocate virtual memory
;  1101          1522  3     lib$free_vm,                      ! Deallocate virtual memory
;  1102          1523  3     sys$cli;                          ! Current CLI's callback address
;  1103          1524  3 MAP
;  1104          1525  3     name :      REF VECTOR;           ! (Static) Descriptor by reference
;  1105          1526  3 LOCAL
;  1106          1527  3     req_desc :  BBLOCK [cli$c_reqdesc], ! Request descriptor block
;  1107          1528  3     rpw :       BBLOCK [cli$c_workarea],! Result parse work area
;  1108          1529  3     req_flags : BITVECTOR [32];       ! Request flags array
;  1109          1530  3 CH$FILL(0, cli$c_reqdesc, req_desc);  ! Zero request descriptor block
;  1110          1531  3 req_desc [int_b_type] = cli$k_present; ! Set request type
;  1111          1532  3 req_desc [int_l_getvm] = lib$get_vm;  ! Set address of get vm routine
;  1112          1533  3 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
;  1113          1534  3 req_desc [int_w_entlen] = .name [0];  ! Set length of passed name
;  1114          1535  3 req_desc [int_l_entaddr] = .name [1]; ! Set address of passed name
;  1115          1536  3 RETURN (sys$cli(req_desc, rpw, req_flags)); ! Call callback utility
;  1116          1537  2 END;


                                                                .EXTRN  LIB$FREE_VM, SYS$CLI

                                                        003C 00000 REAL_CLI$PRESENT:
                                                                .WORD   Save R2,R3,R4,R5                        ; 1518
                                        5E      FF60    CE  9E 00002            MOVAB   -160(SP), SP            ;
                1C              00              6E              00  2C 00007    MOVC5   #0, (SP), #0, #28, REQ_DESC     ; 1530
                                                        E4      AD      0000C   ;
                                                E4      AD      50  8F  90 0000E MOVB    #80, REQ_DESC                  ; 1531
                                        E4      AD  00000000G   00  9E 00013    MOVAB   LIB$GET_VM, REQ_DESC+16         ; 1532
                                        F8      AD  00000000G   00  9E 0001B    MOVAB   LIB$FREE_VM, REQ_DESC+20        ; 1533
```

```
                          50      04  AC  D0 00023         MOVL    NAME, R0                                ; 1534
                       EC AD          60  B0 00027         MOVW    (R0), REQ_DESC+8                        ;
                       F0 AD      04  A0  D0 0002B         MOVL    4(R0), REQ_DESC+12                      ; 1535
                                  5E  DD 00030             PUSHL   SP                                      ; 1536
                                  08  AE  9F 00032         PUSHAB  RPW                                     ;
                                  E4  AD  9F 00035         PUSHAB  REQ_DESC                                ;
                  00000000G       00      03  FB 00038     CALLS   #3, SYS$CLI                             ;
                                          04  0003F         RET                                             ; 1537

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 063D


;  1117          1538  2
;  1118          1539  2 LOCAL status;
;  1119          1540  2
;  1120          1541  2 !
;  1121          1542  2 ! First cancel the CLI's exit handlers. Thus, if a forced exit occurs,
;  1122          1543  2 ! the process terminates, rather than having control return to the CLI
;  1123          1544  2 ! with the process in some state of disarray.
;  1124          1545  2 !
;  1125        P 1546  2 status = $CMEXEC (ROUTIN = lgi$cmsupr,
;  1126          1547  2                   ARGLST = rundown_list);
;  1127          1548  2 IF NOT .status THEN SIGNAL_STOP (.status);
;  1128          1549  2
;  1129          1550  2 !
;  1130          1551  2 ! Change the page protection on the PPD structure to allow user mode
;  1131          1552  2 ! write access, so that this program may run primarily in user mode.
;  1132          1553  2 !
;  1133        P 1554  2 status = $CMEXEC (ROUTIN = set_ppd_prot,
;  1134          1555  2                   ARGLST = prt$c_uw);
;  1135          1556  2 IF NOT .status THEN SIGNAL_STOP (.status);
;  1136          1557  2
;  1137          1558  2 input_fab [fab$w_ifi] = .ppd [ppd$w_inpifi];    ! Restore input IFI
;  1138          1559  2 CH$MOVE(ppd$c_dvifid, ppd [ppd$t_inpdvi], input_nam [nam$t_dvi]);
;  1139          1560  2 output_fab [fab$w_ifi] = .ppd [ppd$w_outifi];   ! Restore output IFI
;  1140          1561  2 output_rab [rab$w_isi] = .ppd [ppd$w_outisi];   ! Restore output ISI
;  1141          1562  2 CH$MOVE(ppd$c_dvifid, ppd [ppd$t_outdvi], output_nam [nam$t_dvi]);
;  1142          1563  2
;  1143          1564  2 !
;  1144          1565  2 ! Security audit all LOGOUTs
;  1145          1566  2 !
;  1146          1567  2 security_audit(nsa$k_rectyp_logo);      ! Security audit all LOGOUTs
;  1147          1568  2
;  1148          1569  2 full_logout = .ppd [ppd$v_mode];        ! Assume /FULL if batch job
;  1149          1570  2
;  1150          1571  2 IF real_cli$present(%ASCID 'FULL')      ! If /FULL present,
;  1151          1572  2 THEN
;  1152          1573  2     full_logout = true;
;  1153          1574  2
;  1154          1575  2 IF real_cli$present(%ASCID 'BRIEF')     ! If /BRIEF present,
;  1155          1576  2 THEN
;  1156          1577  2     full_logout = false;
;  1157          1578  2
;  1158          1579  2 !
;  1159          1580  2 ! If the logout contains an explicit reference to either /HANGUP or
;  1160          1581  2 ! /NOHANGUP then process that request. First disable the installed
;  1161          1582  2 ! LOG_IO privilege so that the terminal's MODHANGUP protection
```

```
;  1162     1583  2 ! functions correctly.
;  1163     1584  2 !
;  1164     1585  2
;  1165     1586  2 clear_log_io();
;  1166     1587  2 status = real_cli$present(%ASCID 'HANGUP');
;  1167     1588  2 IF .status EQL cli$_present
;  1168     1589  2 OR .status EQL cli$_negated
;  1169     1590  2 THEN
;  1170     1591  2     set_terminal_hangup(.status);
;  1171     1592  2
;  1172     1593  2 IF .ctl$gl_creprc_flags[prc$v_batch]            ! If batch job,
;  1173     1594  2 THEN
;  1174     1595  3     BEGIN
;  1175     1596  3     $CMEXEC(ROUTIN = close_input);       ! Close input file
;  1176     1597  3     init_batch();                        ! Get next input file; if none, exit
;  1177     1598  3     $CMEXEC(ROUTIN = open_input);        ! Open input file
;  1178     1599  3     $CMEXEC(ROUTIN = execute_cli);       ! Call already mapped CLI again
;  1179     1600  2     END;
;  1180     1601  2
;  1181     1602  2 logout_message();                        ! Write logout message
;  1182     1603  2
;  1183     1604  2 $CMEXEC(ROUTIN = close_input);          ! Close input file
;  1184     1605  2 $CMEXEC(ROUTIN = close_output);         ! Close output file
;  1185     1606  2
;  1186     1607  2 $CMEXEC(ROUTIN = exit_process);         ! Terminate process
;  1187     1608  2
;  1188     1609  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                        4C  4C  55  46  0016C P.ABM:   .ASCII  \FULL\                                  ;
                                    010E0004  00170 P.ABL:   .LONG   17694724                          ;
                                    00000000' 00174          .ADDRESS P.ABM                        ;
00  00  00  46  45  49  52  42         00178 P.ABO:   .ASCII  \BRIEF\<0><0><0>                       ;
                                    010E0005  00180 P.ABN:   .LONG   17694725                          ;
                                    00000000' 00184          .ADDRESS P.ABO                        ;
00  00  50  55  47  4E  41  48         00188 P.ABQ:   .ASCII  \HANGUP\<0><0>                         ;
                                    010E0006  00190 P.ABP:   .LONG   17694726                          ;
                                    00000000' 00194          .ADDRESS P.ABQ                        ;


                                                .PSECT  $CODE$,NOWRT,2

                                OFFC 00000 LOGOUT:  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11   ; 1494
                5B      0000'   CF   9E 00002       MOVAB   FULL_LOGOUT, R11                       ;
                5A 00000000G    00   9E 00007       MOVAB   CLOSE_INPUT, R10                       ;
                59 00000000G    00   9E 0000E       MOVAB   LIB$STOP, R9                           ;
                58 00000000G    00   9E 00015       MOVAB   PPD+32, R8                             ;
                57 00000000G    00   9E 0001C       MOVAB   SYS$CMEXEC, R7                         ;
                     01C7       CB   9F 00023       PUSHAB  RUNDOWN_LIST                           ; 1547
                   00000000G    00   9F 00027       PUSHAB  LGI$CMSOPR                             ;
                67              02   FB 0002D       CALLS   #2, SYS$CMEXEC                         ;
                56              50   D0 00030       MOVL    R0, STATUS                             ;
                05              56   E8 00033       BLBS    STATUS, 1$                             ; 1548
```

```
                                                56  DD 00036          PUSHL   STATUS                                        ;
                                             69 01  FB 00038          CALLS   #1, LIB$STOP                                ;
                                                04  DD 0003B 1$:      PUSHL   #4                                          ; 1555
                                          0000V CF  9F 0003D          PUSHAB  SET_PPD_PROT
                                             67 02  FB 00041          CALLS   #2, SYS$CMEXEC                              ;
                                             56 50  D0 00044          MOVL    R0, STATUS                                  ;
                                             05 56  E8 00047          BLBS    STATUS, 2$                                  ; 1556
                                                56  DD 0004A          PUSHL   STATUS                                      ;
                                             69 01  FB 0004C          CALLS   #1, LIB$STOP                                ;
                                 00000000G   00 68  B0 0004F 2$:      MOVW    PPD+32, INPUT_FAB+2                         ; 1558
                              00000000G  00   08 A8  1C 28 00056      MOVC3   #28, PPD+40, INPUT_NAM+20                   ; 1559
                                 00000000G   00   04 A8  B0 0005F     MOVW    PPD+36, OUTPUT_FAB+2                        ; 1560
                                 00000000G   00   06 A8  B0 00067     MOVW    PPD+38, OUTPUT_RAB+2                        ; 1561
                              00000000G  00   28 A8  1C 28 0006F      MOVC3   #28, PPD+72, OUTPUT_NAM+20                  ; 1562
                                                07  DD 00078          PUSHL   #7                                          ; 1567
                                 00000000G   00 01  FB 0007A          CALLS   #1, SECURITY_AUDIT
                           50        E2  A8      01  EF 00081          EXTZV   #1, #1, PPD+2, R0                           ; 1569
                                             6B 50  90 00087          MOVB    R0, FULL_LOGOUT
                                          0000' CF  9F 0008A          PUSHAB  P.ABL                                       ; 1571
                                       FF2D  CF 01  FB 0008E          CALLS   #1, REAL_CLI$PRESENT
                                             03 50  E9 00093          BLBC    R0, 3$
                                             6B 01  90 00096          MOVB    #1, FULL_LOGOUT                             ; 1573
                                          0000' CF  9F 00099 3$:      PUSHAB  P.ABN                                       ; 1575
                                       FF1E  CF 01  FB 0009D          CALLS   #1, REAL_CLI$PRESENT
                                             02 50  E9 000A2          BLBC    R0, 4$
                                                6B 94 000A5          CLRB    FULL_LOGOUT                                 ; 1577
                                          0000V CF  00  FB 000A7 4$:  CALLS   #0, CLEAR_LOG_IO                            ; 1586
                                          0000' CF  9F 000AC          PUSHAB  P.ABP                                       ; 1587
                                       FF0B  CF 01  FB 000B0          CALLS   #1, REAL_CLI$PRESENT
                                             56 50  D0 000B5          MOVL    R0, STATUS
                                 00000000G   8F 56  D1 000B8          CMPL    STATUS, #CLI$_PRESENT                       ; 1588
                                             09 13  000BF             BEQL    5$
                                 00000000G   8F 56  D1 000C1          CMPL    STATUS, #CLI$_NEGATED                       ; 1589
                                             07 12  000C8             BNEQ    6$
                                                56  DD 000CA 5$:      PUSHL   STATUS                                      ; 1591
                                       0000V CF  01  FB 000CC          CALLS   #1, SET_TERMINAL_HANGUP
                           24 00000000G   00 04  E1 000D1 6$:         BBC     #4, CTL$GL_CREPRC_FLAGS, 7$                 ; 1593
                                                7E  D4 000D9          CLRL    -(SP)                                       ; 1596
                                                5A  DD 000DB          PUSHL   R10
                                             67 02  FB 000DD          CALLS   #2, SYS$CMEXEC
                                 00000000G   00 00  FB 000E0          CALLS   #0, INIT_BATCH                              ; 1597
                                                7E  D4 000E7          CLRL    -(SP)                                       ; 1598
                                          00000000G 00  9F 000E9      PUSHAB  OPEN_INPUT
                                             67 02  FB 000EF          CALLS   #2, SYS$CMEXEC
                                                7E  D4 000F2          CLRL    -(SP)                                       ; 1599
                                          00000000G 00  9F 000F4      PUSHAB  EXECUTE_CLI
                                             67 02  FB 000FA          CALLS   #2, SYS$CMEXEC
                                       0000V CF  00  FB 000FD 7$:     CALLS   #0, LOGOUT_MESSAGE                         ; 1602
                                                7E  D4 00102          CLRL    -(SP)                                       ; 1604
                                                5A  DD 00104          PUSHL   R10
                                             67 02  FB 00106          CALLS   #2, SYS$CMEXEC
                                                7E  D4 00109          CLRL    -(SP)                                       ; 1605
                                          00000000G 00  9F 0010B      PUSHAB  CLOSE_OUTPUT
                                             67 02  FB 00111          CALLS   #2, SYS$CMEXEC
                                                7E  D4 00114          CLRL    -(SP)                                       ; 1607
                                             0000V CF  9F 00116       PUSHAB  EXIT_PROCESS
                                             67 02  FB 0011A          CALLS   #2, SYS$CMEXEC                              ;
```

```
                                04 0011D        RET                                             ; 1609

; Routine Size:  286 bytes,    Routine Base:  $CODE$ + 067D
```

```
;  1190          1610  1 ROUTINE check_job_quotas: NOVALUE =
;  1191          1611  1
;  1192          1612  1 !---
;  1193          1613  1 !
;  1194          1614  1 !       Check if the interactive job quota has been exceeded, and if
;  1195          1615  1 !       so, issue a fatal message to the user.
;  1196          1616  1 !
;  1197          1617  1 ! Inputs:
;  1198          1618  1 !
;  1199          1619  1 !       uaf_record  = Address of user's UAF record, if any
;  1200          1620  1 !
;  1201          1621  1 ! Outputs:
;  1202          1622  1 !
;  1203          1623  1 !       None
;  1204          1624  1 !---
;  1205          1625  1
;  1206          1626  2 BEGIN
;  1207          1627  2
;  1208          1628  2 EXTERNAL
;  1209          1629  2     sys$gw_ijobcnt:     WORD,           ! Number of interactive jobs
;  1210          1630  2     sys$gw_ijoblim:     WORD;           ! Interactive job limit
;  1211          1631  2
;  1212          1632  2 LOCAL
;  1213          1633  2     privmask:           REF BBLOCK;     ! Address of user's privilege mask
;  1214          1634  2
;  1215          1635  2 IF .uaf_record NEQ 0                   ! If UAF record valid,
;  1216          1636  2 THEN
;  1217          1637  3     BEGIN
;  1218          1638  3     privmask = uaf_record [uaf$q_priv]; ! Get address of user's privmask
;  1219          1639  3     IF .privmask [prv$v_oper]           ! If operator,
;  1220          1640  3     THEN
;  1221          1641  3         RETURN;                         ! then bypass all quota checking
;  1222          1642  2     END;
;  1223          1643  2
;  1224          1644  2 IF .sys$gw_ijobcnt GTRU .sys$gw_ijoblim ! If job limit exceeded,
;  1225          1645  2 THEN
;  1226          1646  3     BEGIN
;  1227          1647  3     IF .sys$gw_ijoblim EQL 0            ! If logins disabled,
;  1228          1648  3     THEN
;  1229          1649  3         SIGNAL_STOP(lgi$_logdisabl)     ! then signal logins disabled
;  1230          1650  3     ELSE
;  1231          1651  3         SIGNAL_STOP(lgi$_exquota);      ! else signal login quota exceeded
;  1232          1652  2     END;
;  1233          1653  2
;  1234          1654  1 END;


                                                .EXTRN  SYS$GW_IJOBCNT, SYS$GW_IJOBLIM

                                0000 00000 CHECK_JOB_QUOTAS:
                                                .WORD   Save nothing                    ; 1610
                     50    0000'  CF   D0 00002        MOVL    UAF_RECORD, R0          ; 1635
                                  09   13 00007        BEQL    1$                      ;
                     50    019C   C0   9E 00009        MOVAB   412(R0), PRIVMASK       ; 1638
                  29 60            12   E0 0000E        BBS     #18, (PRIVMASK), 4$     ; 1639
                     50 00000000G  00   3C 00012 1$:     MOVZWL  SYS$GW_IJOBLIM, R0      ; 1644
```

```
                          50 00000000G   00  B1 00019          CMPW    SYS$GW_IJOBCNT, R0                      ;
                                         19  1B 00020          BLEQU   4$                                      ;
                                         50  D5 00022          TSTL    R0                                      ; 1647
                                         08  12 00024          BNEQ    2$                                      ;
                              00000000G  8F  DD 00026          PUSHL   #LGI$_LOGDISABL                         ; 1649
                                         06  11 0002C          BRB     3$                                      ;
                              00000000G  8F  DD 0002E 2$:      PUSHL   #LGI$_EXQUOTA                           ; 1651
                    00000000G   00       01  FB 00034 3$:      CALLS   #1, LIB$STOP                            ;
                                             04 0003B 4$:      RET                                             ; 1654

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 079B
```

```
;  1236          1655  1 FORWARD
;  1237          1656  1     synch_ipl;                          ! Cell to force locking of this page
;  1238          1657  1
;  1239          1658  1 ROUTINE check_user_quotas =
;  1240          1659  1
;  1241          1660  1 !---
;  1242          1661  1 !
;  1243          1662  1 !       Check if this process has reached it user job limit or account
;  1244          1663  1 !       job limit, if so, issue a fatal message to the user.
;  1245          1664  1 !       This routine runs in kernel mode and depends on all the local
;  1246          1665  1 !       variables to be either in registers or on the stack.
;  1247          1666  1 !
;  1248          1667  1 ! Inputs:
;  1249          1668  1 !
;  1250          1669  1 !       uaf_record  = Address of user's UAF record
;  1251          1670  1 !
;  1252          1671  1 ! Outputs:
;  1253          1672  1 !
;  1254          1673  1 !       None
;  1255          1674  1 !---
;  1256          1675  1
;  1257          1676  2 BEGIN
;  1258          1677  2
;  1259          1678  2 EXTERNAL
;  1260          1679  2     sch$gl_pcbvec:      REF VECTOR,     ! Address of PCB vector
;  1261          1680  2     sch$gl_maxpix:      LONG;           ! Last process slot index
;  1262          1681  2
;  1263          1682  2 LOCAL
;  1264          1683  2     pcb:                REF BBLOCK,     ! user's pcb
;  1265          1684  2     jib:                REF BBLOCK,     ! user's jib
;  1266          1685  2     usercnt,                            ! count of same user logged in
;  1267          1686  2     acntcnt,                            ! count of same accounts logged in
;  1268          1687  2     userstr:            BBLOCK [uaf$s_username],! local storage for user name
;  1269          1688  2     acntstr:            BBLOCK [uaf$s_account]; ! local storage for account
;  1270          1689  2
;  1271          1690  2 usercnt = 0;
;  1272          1691  2 acntcnt = 0;
;  1273          1692  2
;  1274          1693  2 CH$COPY (uaf$s_username, uaf_record [uaf$t_username],
;  1275          1694  2           0,
;  1276          1695  2           uaf$s_username, userstr );
;  1277          1696  2
;  1278          1697  2 CH$COPY (uaf$s_account, uaf_record [uaf$t_account],
;  1279          1698  2           0,
;  1280          1699  2           uaf$s_account, acntstr );
;  1281          1700  2
;  1282          1701  2 !
;  1283          1702  2 ! This page of code needs to be locked in the working set
;  1284          1703  2 !
;  1285          1704  2
;  1286          1705  2 SET_IPL (.synch_ipl);                  ! disable scheduling
;  1287          1706  2
;  1288          1707  2 !
;  1289          1708  2 ! For every process on the system, check the username and account name
;  1290          1709  2 ! in the JIB, and talley the number of users logged in under that username
;  1291          1710  2 ! and account name.
;  1292          1711  2 !
```

```
;  1293          1712  2
;  1294          1713  2 INCR j FROM 2 to .sch$gl_maxpix DO
;  1295          1714  3     BEGIN
;  1296          1715  3     pcb = .sch$gl_pcbvec [.j];          ! get the pcb address
;  1297          1716  3     IF .pcb NEQ .sch$gl_pcbvec [0]      ! if not NULL PROCESS pcb
;  1298          1717  3     THEN
;  1299          1718  3         IF .pcb [pcb$l_owner] EQL 0     ! and not sub-process
;  1300          1719  3         AND .pcb [pcb$v_netwrk] EQL 0   ! and not network job
;  1301          1720  3         THEN
;  1302          1721  4             BEGIN                                                          ;
;  1303          1722  4             jib = .pcb [pcb$l_jib];     ! get the associated jib address
;  1304          1723  4             IF .jib NEQ 0
;  1305          1724  4             THEN
;  1306          1725  5                 BEGIN
;  1307          1726  5                 usercnt = .usercnt + CH$EQL (uaf$s_username,
;  1308          1727  5                                             userstr,
;  1309          1728  5                                             jib$s_username,
;  1310          1729  5                                             jib [jib$t_username],
;  1311          1730  5                                             ' ' );
;  1312          1731  5                 acntcnt = .acntcnt + CH$EQL (uaf$s_account,
;  1313          1732  5                                             acntstr,
;  1314          1733  5                                             jib$s_account,
;  1315          1734  5                                             jib [jib$t_account],
;  1316          1735  5                                             ' ' );
;  1317          1736  4                 END;
;  1318          1737  3             END;
;  1319          1738  2     END;
;  1320          1739  2
;  1321          1740  2 SET_IPL (0);                            ! reenable scheduling
;  1322          1741  2
;  1323          1742  2 IF .uaf_record [uaf$w_maxjobs] NEQ 0    ! If max user jobs specified
;  1324          1743  2 THEN                                    ! check to make sure not exceeded
;  1325          1744  3     BEGIN
;  1326          1745  3     IF .usercnt - .uaf_record [uaf$w_maxjobs] GTR 0
;  1327          1746  3         THEN
;  1328          1747  3             RETURN lgi$_userexc;        ! then signal can't login now
;  1329          1748  2     END;
;  1330          1749  2
;  1331          1750  2 IF .uaf_record [uaf$w_maxacctjobs] NEQ 0       ! If max account jobs specified
;  1332          1751  2 THEN                                    ! check to make sure not exceeded
;  1333          1752  3     BEGIN
;  1334          1753  3     IF .acntcnt - .uaf_record [uaf$w_maxacctjobs] GTR 0
;  1335          1754  3         THEN
;  1336          1755  3             RETURN lgi$_acntexc;        ! then signal can't login now
;  1337          1756  2     END;
;  1338          1757  2
;  1339          1758  2 RETURN TRUE;
;  1340          1759  2
;  1341          1760  1 END;


                                                    .EXTRN  SCH$GL_PCBVEC, SCH$GL_MAXPIX

                                    0FFC 00000 CHECK_USER_QUOTAS:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1658
                           5E       C0   AE  9E 00002       MOVAB   -64(SP), SP                         ;
```

```
                                                5A  7C 00006          CLRQ    ACNTCNT                                 ; 1691
                                             58     0000'  CF  D0 00008          MOVL    UAF_RECORD, R8                          ; 1693
                                          20 AE     04  A8  20  28 0000D          MOVC3   #32, 4(R8), USERSTR
                                          6E     34  A8  20  28 00013          MOVC3   #32, 52(R8), ACNTSTR                    ; 1697
                                             12     0000V  CF  DA 00018          MTPR    SYNCH_IPL, #18                          ; 1705
                                             59 00000000G  00  D0 0001D          MOVL    SCH$GL_PCBVEC, R9                       ; 1715
                                             57            01  D0 00024          MOVL    #1, J
                                                           3B  11 00027          BRB     4$
                                             54          6947  D0 00029 1$:      MOVL    (R9)[J], PCB
                                             69            54  D1 0002D          CMPL    PCB, (R9)                               ; 1716
                                                           32  13 00030          BEQL    4$
                                                        1C  A4  D5 00032          TSTL    28(PCB)                                 ; 1718
                                                           2D  12 00035          BNEQ    4$
                                             28        26  A4  05  E0 00037          BBS     #5, 38(PCB), 4$                         ; 1719
                                                56     0080  C4  D0 0003C          MOVL    128(PCB), JIB                           ; 1722
                                                           21  13 00041          BEQL    4$                                      ; 1723
                                                           55  D4 00043          CLRL    R5                                      ; 1729
                                    0C        20        20  AE  20  2D 00045          CMPC5   #32, USERSTR, #32, #12, 12(JIB)
                                                        0C  A6      0004B
                                                           02  12 0004D          BNEQ    2$
                                                           55  D6 0004F          INCL    R5
                                             5B            55  C0 00051 2$:      ADDL2   R5, USERCNT
                                                           55  D4 00054          CLRL    R5                                      ; 1734
                                    08        20           6E  20  2D 00056          CMPC5   #32, ACNTSTR, #32, #8, 24(JIB)
                                                        18  A6      0005B
                                                           02  12 0005D          BNEQ    3$
                                                           55  D6 0005F          INCL    R5
                                             5A            55  C0 00061 3$:      ADDL2   R5, ACNTCNT
                                          BD               57 00000000G  00  F3 00064 4$:      AOBLEQ  SCH$GL_MAXPIX, J, 1$            ; 1713
                                                           12  00  DA 0006C          MTPR    #0, #18                                 ; 1740
                                                           50     0206  C8  3C 0006F          MOVZWL  518(R8), R0                     ; 1742
                                                                      0D  13 00074          BEQL    5$
                                             50                       5B  D1 00076          CMPL    USERCNT, R0                     ; 1745
                                                                      08  15 00079          BLEQ    5$
                                             50 00000000G             8F  D0 0007B          MOVL    #LGI$_USEREXC, R0               ; 1747
                                                                          04 00082          RET
                                             50                    0208  C8  3C 00083 5$:      MOVZWL  520(R8), R0             ; 1750
                                                                      0D  13 00088          BEQL    6$
                                             50                       5A  D1 0008A          CMPL    ACNTCNT, R0             ; 1753
                                                                      08  15 0008D          BLEQ    6$
                                             50 00000000G             8F  D0 0008F          MOVL    #LGI$_ACNTEXC, R0       ; 1755
                                                                          04 00096          RET
                                             50                       01  D0 00097 6$:      MOVL    #1, R0                  ; 1758
                                                                          04 0009A          RET                             ; 1760

; Routine Size:  155 bytes,    Routine Base:  $CODE$ + 07D7

;  1342          1761  1
;  1343          1762  1 PSECT OWN = $CODE$(NOWRITE,EXECUTE);     ! Cause this own to occur here
;  1344          1763  1
;  1345          1764  1 OWN synch_ipl:  INITIAL(ipl$_synch);     ! Used to lock previous code page
;  1346          1765  1
;  1347          1766  1 PSECT OWN = $OWN$(WRITE,NOEXECUTE);      ! Restore normal own psect
```

```
;1349     1767  1 ROUTINE check_user_hours (time) : NOVALUE =
;1350     1768  1
;1351     1769  1 !---
;1352     1770  1 !
;1353     1771  1 !       Check to see if this process is allowed to login now on the type of
;1354     1772  1 !       terminal line indicated. If not, issue a fatal message to the user.
;1355     1773  1 !
;1356     1774  1 ! Inputs:
;1357     1775  1 !
;1358     1776  1 !       time = quadword date and time
;1359     1777  1 !       uaf_record  = Address of user's UAF record, if any
;1360     1778  1 !
;1361     1779  1 ! Outputs:
;1362     1780  1 !
;1363     1781  1 !       None
;1364     1782  1 !---
;1365     1783  1
;1366     1784  2 BEGIN
;1367     1785  2
;1368     1786  2 STRUCTURE
;1369     1787  2     threebytevector [i; n, ext=0] =
;1370     1788  2         [n*3]
;1371     1789  2         (threebytevector+i*3)<0, 24, ext>;
;1372     1790  2
;1373     1791  2 MAP
;1374     1792  2     time:               REF VECTOR;     ! Quadword date and time
;1375     1793  2
;1376     1794  2 EXTERNAL
;1377     1795  2     exe$gl_flags:       BITVECTOR;      ! System wide flag bits
;1378     1796  2
;1379     1797  2 EXTERNAL LITERAL
;1380     1798  2     exe$v_explicitp:    UNSIGNED (6),   ! day type set by operator
;1381     1799  2     exe$v_explicits:    UNSIGNED (6);   ! flag as to whether operator set day
;1382     1800  2
;1383     1801  2 LOCAL
;1384     1802  2     day,                                ! Day of week for today
;1385     1803  2     hour,                               ! Hour of day for today
;1386     1804  2     secondary_day,                      ! Prime/secondary day flag
;1387     1805  2     flags;                              ! Longword for correct days flags
;1388     1806  2
;1389     1807  2 !
;1390     1808  2 ! Define a vector structure over the UAF hourly restriction fields
;1391     1809  2 ! for quick reference.
;1392     1810  2 !
;1393     1811  2 $ASSUME (jib$c_network, EQL, 1);
;1394     1812  2 $ASSUME (jib$c_batch, EQL, 2);
;1395     1813  2 $ASSUME (jib$c_local, EQL, 3);
;1396     1814  2 $ASSUME (jib$c_dialup, EQL, 4);
;1397     1815  2 $ASSUME (jib$c_remote, EQL, 5);
;1398     1816  2 $ASSUME ($BYTEOFFSET (uaf$b_network_access_s), EQL, $BYTEOFFSET (uaf$b_network_access_p)+3);
;1399     1817  2 $ASSUME ($BYTEOFFSET (uaf$b_batch_access_p), EQL, $BYTEOFFSET (uaf$b_network_access_s)+3);
;1400     1818  2 $ASSUME ($BYTEOFFSET (uaf$b_batch_access_s), EQL, $BYTEOFFSET (uaf$b_batch_access_p)+3);
;1401     1819  2 $ASSUME ($BYTEOFFSET (uaf$b_local_access_p), EQL, $BYTEOFFSET (uaf$b_batch_access_s)+3);
;1402     1820  2 $ASSUME ($BYTEOFFSET (uaf$b_local_access_s), EQL, $BYTEOFFSET (uaf$b_local_access_p)+3);
;1403     1821  2 $ASSUME ($BYTEOFFSET (uaf$b_dialup_access_p), EQL, $BYTEOFFSET (uaf$b_local_access_s)+3);
;1404     1822  2 $ASSUME ($BYTEOFFSET (uaf$b_dialup_access_s), EQL, $BYTEOFFSET (uaf$b_dialup_access_p)+3);
;1405     1823  2 $ASSUME ($BYTEOFFSET (uaf$b_remote_access_p), EQL, $BYTEOFFSET (uaf$b_dialup_access_s)+3);
```

```
;  1406          1824  2 $ASSUME ($BYTEOFFSET (uaf$b_remote_access_s), EQL, $BYTEOFFSET (uaf$b_remote_access_p)+3);
;  1407          1825  2
;  1408          1826  2 BIND
;  1409          1827  2     restrict_vector = uaf_record[uaf$b_network_access_p]
;  1410          1828  2                     : threebytevector;
;  1411          1829  2
;  1412          1830  2 !
;  1413          1831  2 ! Hourly restrictions are not processed on detached jobs - they inherit
;  1414          1832  2 ! their creator's restrictions and will be checked up on by the job
;  1415          1833  2 ! controller.
;  1416          1834  2 !
;  1417          1835  2 IF .job_type EQL jib$c_detached THEN RETURN;
;  1418          1836  2
;  1419          1837  2 !
;  1420          1838  2 ! Get the necessary components of the time of day.
;  1421          1839  2 !
;  1422          1840  2 lib$day_of_week (.time, day);
;  1423          1841  2 lib$hour_of_day (.time, hour);
;  1424          1842  2
;  1425          1843  2 !
;  1426          1844  2 ! Find out whether today is a primary or secondary day, either from
;  1427          1845  2 ! the UAF record or from the operator override.
;  1428          1846  2 !
;  1429          1847  2 secondary_day = .BITVECTOR [uaf_record [uaf$b_primedays], .day-1];
;  1430          1848  2 IF .exe$gl_flags [exe$v_explicits]
;  1431          1849  2 THEN secondary_day = .exe$gl_flags [exe$v_explicitp];
;  1432          1850  2
;  1433          1851  2 !
;  1434          1852  2 ! Check the appropriate restriction vector, depending on login and day type,
;  1435          1853  2 ! and give an error message indicating whether the user is bagged for
;  1436          1854  2 ! right now, today, or always.
;  1437          1855  2 !
;  1438          1856  2 flags = .restrict_vector [(.job_type-1)*2+.secondary_day];
;  1439          1857  2 IF .BITVECTOR [flags, .hour]
;  1440          1858  2 THEN
;  1441          1859  3     BEGIN
;  1442          1860  3     IF .flags NEQ %X'FFFFFF'
;  1443          1861  3     THEN SIGNAL_STOP (lgi$_badhour)
;  1444          1862  3     ELSE IF .restrict_vector [(.job_type-1)*2+1-.secondary_day] NEQ %X'FFFFFF'
;  1445          1863  3     THEN SIGNAL_STOP (lgi$_badday)
;  1446          1864  3     ELSE SIGNAL_STOP (lgi$_restrict);
;  1447          1865  2     END;
;  1448          1866  2
;  1449          1867  1 END;


                                        00872          .BLKB   2
                              00000008  00874 SYNCH_IPL:
                                                       .LONG   8                                                       ;

                                                       .EXTRN  EXE$GL_FLAGS, EXE$V_EXPLICITP
                                                       .EXTRN  EXE$V_EXPLICITS

                                        003C 00000 CHECK_USER_HOURS:
                                                       .WORD   Save R2,R3,R4,R5                                        ; 1767
                                  55 00000000G   00  9E 00002          MOVAB   EXE$GL_FLAGS, R5                        ;
```

```
                                5E  08  C2 00009          SUBL2   #8, SP                                          ;
           53       0000'   CF 000001D8  8F  C1 0000C          ADDL3   #472, UAF_RECORD, R3                            ; 1827
                                   0000'  CF  D5 00016          TSTL    JOB_TYPE                                        ; 1835
                                          01  12 0001A          BNEQ    1$                                              ;
                                          04 0001C              RET                                                     ;
                                          5E  DD 0001D 1$:      PUSHL   SP                                              ; 1840
                                    04    AC  DD 0001F          PUSHL   TIME                                            ;
                      00000000G  00       02  FB 00022          CALLS   #2, LIB$DAY_OF_WEEK                             ;
                                    04    AE  9F 00029          PUSHAB  HOUR                                            ; 1841
                                    04    AC  DD 0002C          PUSHL   TIME                                            ;
                      00000000G  00       02  FB 0002F          CALLS   #2, LIB$HOUR_OF_DAY                             ;
                                 50 0000'  CF  D0 00036          MOVL    UAF_RECORD, R0                                  ; 1847
                           51    6E       01  C3 0003B          SUBL3   #1, DAY, R1                                     ;
         54      0202      C0    01       51  EF 0003F          EXTZV   R1, #1, 514(R0), SECONDARY_DAY                  ;
                           05    65      00G  E1 00046          BBC     S^EXE$V_EXPLICITS, EXE$GL_FLAGS, 2$             ; 1848
         54                65    01      00G  EF 0004A          EXTZV   S^EXE$V_EXPLICITP, #1, EXE$GL_FLAGS, -          ; 1849
                                                                        SECONDARY_DAY                                   ;
                                 52 0000'  CF  D0 0004F 2$:      MOVL    JOB_TYPE, R2                                    ; 1856
                                 50   6442    3E 00054          MOVAW   (SECONDARY_DAY)[R2], R0                         ;
                                 50       03  C4 00058          MULL2   #3, R0                                          ;
         50           FA A043    18       00  EF 0005B          EXTZV   #0, #24, -6(R0)[R3], FLAGS                      ;
                           3D    50  04   AE  E1 00062          BBC     HOUR, FLAGS, 6$                                 ; 1857
                      00FFFFFF   8F       50  D1 00067          CMPL    FLAGS, #16777215                                ; 1860
                                          08  13 0006E          BEQL    3$                                              ;
                                 00000000G 8F DD 00070          PUSHL   #LGI$_BADHOUR                                   ; 1861
                                          25  11 00076          BRB     5$                                              ;
                           50    52       01  78 00078 3$:      ASHL    #1, R2, R0                                      ; 1862
                                 50       54  C2 0007C          SUBL2   SECONDARY_DAY, R0                               ;
                                 50       03  C4 0007F          MULL2   #3, R0                                          ;
00FFFFFF  8F          FD A043    18       00  ED 00082          CMPZV   #0, #24, -3(R0)[R3], #16777215                  ;
                                          08  13 0008D          BEQL    4$                                              ;
                                 00000000G 8F DD 0008F          PUSHL   #LGI$_BADDAY                                    ; 1863
                                          06  11 00095          BRB     5$                                              ;
                                 00000000G 8F DD 00097 4$:      PUSHL   #LGI$_RESTRICT                                  ; 1864
                      00000000G  00       01  FB 0009D 5$:      CALLS   #1, LIB$STOP                                    ;
                                          04 000A4 6$:          RET                                                     ; 1867

; Routine Size:  165 bytes,    Routine Base:  $CODE$ + 0878
```

```
;  1451          1868  1 ROUTINE check_account_expiration (time) : NOVALUE =
;  1452          1869  1
;  1453          1870  1 !---
;  1454          1871  1 !
;  1455          1872  1 !       Check to see if the account that this process belongs to has expired.
;  1456          1873  1 !       If so, issue a fatal message to the user.
;  1457          1874  1 !
;  1458          1875  1 ! Inputs:
;  1459          1876  1 !
;  1460          1877  1 !       time = quadword date and time
;  1461          1878  1 !       uaf_record  = Address of user's UAF record, if any
;  1462          1879  1 !
;  1463          1880  1 ! Outputs:
;  1464          1881  1 !
;  1465          1882  1 !       None
;  1466          1883  1 !---
;  1467          1864  1
;  1468          1885  2 BEGIN
;  1469          1886  2
;  1470          1887  2 MAP
;  1471          1888  2     time:                REF VECTOR;     ! Quadword date and time
;  1472          1889  2
;  1473          1890  2 BIND
;  1474          1891  2     expiration = uaf_record [uaf$q_expiration] : VECTOR; ! Use as 2 longwords
;  1475          1892  2
;  1476          1893  2 !
;  1477          1894  2 ! Return if no UAF record or no expiration date.
;  1478          1895  2 !
;  1479          1896  2 IF .uaf_record EQL 0
;  1480          1897  2 THEN RETURN;
;  1481          1898  2
;  1482          1899  3 IF (.expiration [0] EQL 0) AND (.expiration [1] EQL 0)
;  1483          1900  2 THEN RETURN;
;  1484          1901  2
;  1485          1902  2 !
;  1486          1903  2 ! Compare it to the account expiration date and return if the current time
;  1487          1904  2 ! is less.  Otherwise signal a fatal error.
;  1488          1905  2 !
;  1489          1906  2 IF .time [1] LSSU .expiration [1]
;  1490          1907  2 THEN RETURN;
;  1491          1908  2
;  1492          1909  2 IF .time [1] EQL .expiration [1]
;  1493          1910  2 AND .time [0] LSSU .expiration [0]
;  1494          1911  2 THEN RETURN;
;  1495          1912  2
;  1496          1913  2 SIGNAL_STOP (lgi$_acntexpir);
;  1497          1914  1 END;


                                        0000 00000 CHECK_ACCOUNT_EXPIRATION:
                                                   .WORD   Save nothing                          ; 1868
     51     0000'  CF 0000016C   8F  C1 00002             ADDL3   #364, UAF_RECORD, R1          ; 1891
                          0000'  CF  D5 0000C             TSTL    UAF_RECORD                    ; 1896
                                 28  13 00010             BEQL    3$                            ;
```

```
                                    61  D5 00012          TSTL    (R1)                                    ; 1899
                                    05  12 00014          BNEQ    1$                                      ;
                                04  A1  D5 00016          TSTL    4(R1)                                   ;
                                    1F  13 00019          BEQL    3$                                      ;
                            50      04  AC  D0 0001B 1$:   MOVL    TIME, R0                                ; 1906
                        04  A1      04  A0  D1 0001F       CMPL    4(R0), 4(R1)                            ;
                                    14  1F 00024          BLSSU   3$                                      ;
                                    05  12 00026          BNEQ    2$                                      ;
                            61                  60  D1 00028       CMPL    (R0), (R1)                      ; 1909
                                    0D  1F 0002B          BLSSU   3$                                      ; 1910
                                00000000G   8F  DD 0002D 2$:   PUSHL   #LGI$_ACNTEXPIR                     ;
                    00000000G   00          01  FB 00033       CALLS   #1, LIB$STOP                        ; 1913
                                    04 0003A 3$:       RET                                                 ;
                                                                                                           ; 1914

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 091D
```

```
;  1499          1915  1 GLOBAL ROUTINE update_uaf_record (time) : NOVALUE =
;  1500          1916  1
;  1501          1917  1 !---
;  1502          1918  1 !
;  1503          1919  1 !       Refetch and lock the UAF record.
;  1504          1920  1 !       Check for password expiration and set the UAF flag.
;  1505          1921  1 !       Update the last login time.
;  1506          1922  1 !       Zero the login failure count.
;  1507          1923  1 !
;  1508          1924  1 !       If time is missing, then only increment the login failure count.
;  1509          1925  1 !
;  1510          1926  1 ! Inputs:
;  1511          1927  1 !
;  1512          1928  1 !       time = quadword date and time
;  1513          1929  1 !       uaf_rab = RAB used to read the UAF
;  1514          1930  1 !       uaf_fab = FAB used to read the UAF
;  1515          1931  1 !
;  1516          1932  1 ! Outputs:
;  1517          1933  1 !
;  1518          1934  1 !       None
;  1519          1935  1 !---
;  1520          1936  1
;  1521          1937  2 BEGIN
;  1522          1938  2
;  1523          1939  2 BUILTIN
;  1524          1940  2     EMUL,
;  1525          1941  2     SUBM,
;  1526          1942  2     NULLPARAMETER;
;  1527          1943  2
;  1528          1944  2 MAP
;  1529          1945  2     time:                   REF VECTOR;     ! Quadword date and time
;  1530          1946  2
;  1531          1947  2 BIND
;  1532          1948  2     lifetime = uaf_record [uaf$q_pwd_lifetime] : VECTOR,
;  1533          1949  2     pwd1 = uaf_record [uaf$q_pwd] : VECTOR,
;  1534          1950  2     pwd2 = uaf_record [uaf$q_pwd2] : VECTOR;
;  1535          1951  2
;  1536          1952  2 LOCAL
;  1537          1953  2     status,
;  1538          1954  2     expiration:             VECTOR [2],     ! Quadword date and time
;  1539          1955  2     expir_5days:            VECTOR [2],     !  and another
;  1540          1956  2     delta_5days:            VECTOR [2];     ! Positive delta of 5 days
;  1541          1957  2
;  1542          1958  2 !
;  1543          1959  2 ! Refetch the UAF record.
;  1544          1960  2 !
;  1545          1961  2 uaf_rab [rab$b_rac] = rab$c_rfa;
;  1546          1962  2 uaf_rab [rab$l_rop] = rab$m_rlk OR rab$m_wat;
;  1547          1963  3 IF NOT (status = $GET (RAB = uaf_rab))
;  1548          1964  2 THEN
;  1549          1965  3     BEGIN
;  1550          1966  3     $DISCONNECT (RAB = uaf_rab);
;  1551          1967  3     $CLOSE (FAB = uaf_fab);
;  1552          1968  3     RETURN;
;  1553          1969  2     END;
;  1554          1970  2
;  1555          1971  2 IF NULLPARAMETER (1)
```

```
;  1556          1972  2 THEN
;  1557          1973  3     BEGIN
;  1558          1974  3     !
;  1559          1975  3     ! Update the number of login failures.
;  1560          1976  3     !
;  1561          1977  3     uaf_record [uaf$w_logfails] = .uaf_record [uaf$w_logfails] + 1;
;  1562          1978  3     !
;  1563          1979  3     ! If this was tagged as a breakin attempt and we are disabling users
;  1564          1980  3     ! on attempted breakin, set the DISUSER flag.
;  1565          1981  3     !
;  1566          1982  3     IF .break_attempt
;  1567          1983  3     AND .exe$gl_dynamic_flags[exe$v_brk_disuser]
;  1568          1984  3     THEN uaf_record[uaf$v_disacnt] = true;
;  1569          1985  3     END
;  1570          1986  2 ELSE
;  1571          1987  3     BEGIN
;  1572          1988  3     !
;  1573          1989  3     ! Update the last login time.
;  1574          1990  3     !
;  1575          1991  3     IF .ppd [ppd$v_mode]
;  1576          1992  3     THEN
;  1577          1993  4         move_quad(time [0], uaf_record [uaf$q_lastlogin_n])
;  1578          1994  3     ELSE
;  1579          1995  4         BEGIN
;  1580          1996  4         uaf_record [uaf$w_logfails] = 0;
;  1581          1997  4         move_quad(time [0], uaf_record [uaf$q_lastlogin_i]);
;  1582          1998  4         !
;  1583          1999  4         ! Check for expired passwords.  Skip if no expiration period.
;  1584          2000  4         ! Note the signed compare; it is used to catch the pre-expired
;  1585          2001  4         ! encoding of -1,-1.
;  1586          2002  4         !
;  1587          2003  5         IF (.lifetime [0] NEQ 0) OR (.lifetime [1] NEQ 0)
;  1588          2004  4         THEN
;  1589          2005  5             BEGIN
;  1590          2006  5             !
;  1591          2007  5             ! Check first password.
;  1592          2008  5             !
;  1593          2009  5             EMUL (%REF(10*1000*1000), %REF(60*60*24*5), %REF(0), delta_5days);
;  1594          2010  6             IF (.pwd1 [0] NEQ 0) OR (.pwd1 [1] NEQ 0)
;  1595          2011  5             THEN
;  1596          2012  6                 BEGIN
;  1597          2013  6                 SUBM (2, lifetime, uaf_record [uaf$q_pwd_date], expiration);
;  1598          2014  6                 IF .time [1] GTR .expiration [1]
;  1599          2015  7                 OR (.time [1] EQL .expiration [1]
;  1600          2016  7                 AND .time [0] GEQU .expiration [0])
;  1601          2017  6                 THEN
;  1602          2018  7                     BEGIN
;  1603          2019  7                     uaf_record [uaf$v_pwd_expired] = true;
;  1604          2020  7                     IF .(uaf_record [uaf$q_pwd2_date]) eql 0
;  1605          2021  7                     THEN
;  1606        L 2022  7                         write_output(%ASCID %STRING(
;  1607        L 2023  7     %CHAR(bell),%CHAR(bell),%CHAR(bell),
;  1608          2024  7     'WARNING - Your password has expired; update immediately with SET PASSWORD!'))
;  1609          2025  7                     ELSE
;  1610        L 2026  7                         write_output(%ASCID %STRING(
;  1611        L 2027  7     %CHAR(bell),%CHAR(bell),%CHAR(bell),
;  1612          2028  7     'WARNING - Primary password has expired; update immediately with SET PASSWORD!'));
```

```
;  1613          2029  7                    END
;  1614          2030  6                ELSE
;  1615          2031  7                    BEGIN
;  1616          2032  7                    SUBM (2, delta_5days, expiration, expir_5days);
;  1617          2033  7                    IF .time [1] GTRU .expir_5days [1]
;  1618          2034  8                    OR (.time [1] EQL .expir_5days [1]
;  1619          2035  8                    AND .time [0] GEQU .expir_5days [0])
;  1620          2036  7                    THEN
;  1621          2037  8                        BEGIN
;  1622          2038  8                        IF .(uaf_record [uaf$q_pwd2_date]) eql 0
;  1623          2039  8                        THEN
;  1624        L 2040  8                            write_fao(UPLIT BYTE(%ASCIC %STRING(
;  1625        L 2041  8                %CHAR(bell),%CHAR(bell),
;  1626          2042  8                'WARNING - Your password expires on !AC, !17%D')),
;  1627          2043  8                            ascic_day_of_week(expiration), expiration)
;  1628          2044  8                        ELSE
;  1629        L 2045  8                            write_fao(UPLIT BYTE(%ASCIC %STRING(
;  1630        L 2046  8                %CHAR(bell),%CHAR(bell),
;  1631          2047  8                'WARNING - Primary password expires on !AC, !17%D')),
;  1632          2048  8                            ascic_day_of_week(expiration), expiration);
;  1633          2049  7                        END;
;  1634          2050  6                    END;
;  1635          2051  5                END;
;  1636          2052  5            !
;  1637          2053  5            ! Check second password.
;  1638          2054  5            !
;  1639          2055  6            IF (.pwd2 [0] NEQ 0) OR (.pwd2 [1] NEQ 0)
;  1640          2056  5            THEN
;  1641          2057  6                BEGIN
;  1642          2058  6                SUBM (2, lifetime, uaf_record [uaf$q_pwd2_date], expiration);
;  1643          2059  6                IF .time [1] GTR .expiration [1]
;  1644          2060  7                OR (.time [1] EQL .expiration [1]
;  1645          2061  7                AND .time [0] GEQU .expiration [0])
;  1646          2062  6                THEN
;  1647          2063  7                    BEGIN
;  1648          2064  7                    uaf_record [uaf$v_pwd2_expired] = true;
;  1649        L 2065  7                    write_output(%ASCID %STRING(
;  1650        L 2066  7                %CHAR(bell),%CHAR(bell),%CHAR(bell),
;  1651          2067  7                'WARNING - Secondary password has expired; update immediately with SET PASSWORD!'));
;  1652          2068  7                    END
;  1653          2069  6                ELSE
;  1654          2070  7                    BEGIN
;  1655          2071  7                    SUBM (2, delta_5days, expiration, expir_5days);
;  1656          2072  7                    IF .time [1] GTRU .expir_5days [1]
;  1657          2073  8                    OR (.time [1] EQL .expir_5days [1]
;  1658          2074  8                    AND .time [0] GEQU .expir_5days [0])
;  1659          2075  7                    THEN
;  1660        L 2076  7                        write_fao(UPLIT BYTE(%ASCIC %STRING(
;  1661        L 2077  7                %CHAR(bell),%CHAR(bell),
;  1662          2078  7                'WARNING - Secondary password expires on !AC, !17%D')),
;  1663          2079  7                            ascic_day_of_week(expiration), expiration);
;  1664          2080  6                    END;
;  1665          2081  5                END;
;  1666          2082  4            END;
;  1667          2083  3        END;
;  1668          2084  2    END;
;  1669          2085  2
```

```
;  1670          2086  2 !
;  1671          2087  2 ! Update the UAF record and close the file.
;  1672          2088  2 !
;  1673          2089  2 $UPDATE (rab = uaf_rab);
;  1674          2090  2 $DISCONNECT (RAB = uaf_rab)
;  1675          2091  2 $CLOSE (FAB = uaf_fab);
;  1676          2092  2
;  1677          2093  1 END;


                                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

6F 59 20 2D 20 47 4E 49 4E 52 41 57 07 07 07  00198 P.ABS:   .ASCII  <7><7><7>\WARNING - Your passwor\
                  72 6F 77 73 73 61 70 20 72 75  001A7
20 3B 64 65 72 69 70 78 65 20 7?? 61 68 20 64  001B1          .ASCII  \d has expired; update immediately with S\
74 61 69 64 65 6D 6D 69 20 65 .. 61 64 70 75  001C0
                  53 20 68 74 69 77 20 79 6C 65  001CF
00 00 00 21 44 52 4F 57 53 53 41 50 20 54 45  001D9          .ASCII  \ET PASSWORD!\<0><0><0>
                                    010E004D  001E8 P.ABR:   .LONG   17694797
                                    00000000' 001EC          .ADDRESS P.ABS
72 50 20 2D 20 47 4E 49 4E 52 41 57 07 07 07  001F0 P.ABU:   .ASCII  <7><7><7>\WARNING - Primary pass\
                  73 73 61 70 20 79 72 61 6D 69  001FF
65 72 69 70 78 65 20 73 61 68 20 64 72 6F 77  00209          .ASCII  \word has expired; update immediately wit\
64 65 6D 6D 69 20 65 74 61 64 70 75 20 3B 64  00218
                  74 69 77 20 79 6C 65 74 61 69  00227
21 44 52 4F 57 53 53 41 50 20 54 45 53 20 68  00231          .ASCII  \h SET PASSWORD!\
                                    010E0050  00240 P.ABT:   .LONG   17694800
                                    00000000' 00244          .ADDRESS P.ABU
6F 59 20 2D 20 47 4E 49 4E 52 41 57 07 07 2F  00248 P.ABV:   .ASCII  \/\<7><7>\WARNING - Your password ex\
   78 65 20 64 72 6F 77 73 73 61 70 20 72 75  00257
21 20 2C 43 41 21 20 6E 6F 20 73 65 72 69 70  00265          .ASCII  \pires on !AC, !17%D\
                                 44 25 37 31  00274
72 50 20 2D 20 47 4E 49 4E 52 41 57 07 07 32  00278 P.ABW:   .ASCII  \2\<7><7>\WARNING - Primary password\
   64 72 6F 77 73 73 61 70 20 79 72 61 6D 69  00287
43 41 21 20 6E 6F 20 73 65 72 69 70 78 65 20  00295          .ASCII  \ expires on !AC, !17%D\
                        44 25 37 31 21 20 2C  002A4
                                              002AB          .BLKB   1
65 53 20 2D 20 47 4E 49 4E 52 41 57 07 07 07  002AC P.ABY:   .ASCII  <7><7><7>\WARNING - Secondary pa\
                  61 70 20 79 72 61 64 6E 6F 63  002BB
69 70 78 65 20 73 61 68 20 64 72 6F 77 73 73  002C5          .ASCII  \ssword has expired; update immediately w\
6D 6D 69 20 65 74 61 64 70 75 20 3B 64 65 72  002D4
                  77 20 79 6C 65 74 61 69 64 65  002E3
52 4F 57 53 53 41 50 20 54 45 53 20 68 74 69  002ED          .ASCII  \ith SET PASSWORD!\<0><0>
                                 00 00 21 44  002FC
                                    010E0052  00300 P.ABX:   .LONG   17694802
                                    00000000' 00304          .ADDRESS P.ABY
65 53 20 2D 20 47 4E 49 4E 52 41 57 07 07 34  00308 P.ABZ:   .ASCII  \4\<7><7>\WARNING - Secondary passwo\
   6F 77 73 73 61 70 20 79 72 61 64 6E 6F 63  00317
21 20 6E 6F 20 73 65 72 69 70 78 65 20 64 72  00325          .ASCII  \rd expires on !AC, !17%D\
                  44 25 37 31 21 20 2C 43 41  00334

                                                               .EXTRN  SYS$GET, SYS$DISCONNECT
                                                               .EXTRN  SYS$CLOSE, SYS$UPDATE

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

```
                                 0FFC 00000          .ENTRY  UPDATE_UAF_RECORD, Save R2,R3,R4,R5,R6,R7,-  ; 1915
                                                             R8,R9,R10,R11
                      5B 00000000G   00  9E 00002    MOVAB   WRITE_FAO, R11                               ;
                      5A 00000000G   00  9E 00009    MOVAB   WRITE_OUTPUT, R10                            ;
                      59 00000000G   00  9E 00010    MOVAB   ASCIC_DAY_OF_WEEK, R9                        ;
                      58      0000'  CF  9E 00017    MOVAB   UAF_RECORD, R8                               ;
                      5E             18  C2 0001C    SUBL2   #24, SP                                      ;
                      50             68  D0 0001F    MOVL    UAF_RECORD, R0                               ; 1948
                      54      0174   C0  9E 00022    MOVAB   372(R0), R4                                  ;
                      57      0154   C0  9E 00027    MOVAB   340(R0), R7                                  ; 1949
                      56      015C   C0  9E 0002C    MOVAB   348(R0), R6                                  ; 1950
                   DA    A8        02  90 00031    MOVB    #2, UAF_RAB+30                               ; 1961
                   C0    A8 000A0000 8F  D0 00035    MOVL    #655360, UAF_RAB+4                           ; 1962
                         BC    A8    9F  0003D      PUSHAB  UAF_RAB                                      ; 1963
            00000000G    00          01  FB 00040    CALLS   #1, SYS$GET                                  ;
                         03          50  E8 00047    BLBS    STATUS, 1$                                   ;
                                016A 31  0004A      BRW     22$                                          ;
                         52          68  D0 0004D 1$: MOVL    UAF_RECORD, R2                               ; 1977
                                     6C  95 00050    TSTB    (AP)                                         ; 1971
                                     05  13 00052    BEQL    2$                                           ;
                               04    AC  D5 00054    TSTL    4(AP)                                        ;
                                     18  12 00057    BNEQ    3$                                           ;
                               0164  C2  B6 00059 2$: INCW    356(R2)                                      ; 1977
                         27    FF66  C8  E9 0005D    BLBC    BREAK_ATTEMPT, 4$                            ; 1982
                1F 00000000G    00        00G E1 00062    BBC     S^EXE$V_BRK_DISUSER, EXE$GL_DYNAMIC_FLAGS, - ; 1983
                                                             4$
                      01D4      C2        10  88 0006A    BISB2   #16, 468(R2)                             ; 1984
                                          18  11 0006F    BRB     4$                                       ; 1971
                         53       04      AC  D0 00071 3$: MOVL    TIME, R3                                 ; 1993
                   0F 00000000G   00        01  E1 00075    BBC     #1, PPD+2, 5$                            ; 1991
                         0194     C2       04  BC D0 0007D    MOVL    @TIME, 404(R2)                           ; 1993
                         0198     C2       04  A3 D0 00083    MOVL    4(R3), 408(R2)                           ;
                                        0121 31  00089 4$:   BRW     21$                                      ; 1991
                                     0164  C2  B4 0008C 5$:   CLRW    356(R2)                                  ; 1996
                         55       04      BC  D0 00090    MOVL    @TIME, R5                                ; 1997
                         018C     C2          55  D0 00094    MOVL    R5, 396(R2)                              ;
                         0190     C2       04  A3 D0 00099    MOVL    4(R3), 400(R2)                           ;
                                          64  D5 0009F    TSTL    (R4)                                     ; 2003
                                          05  12 000A1    BNEQ    6$                                       ;
                                     04   A4  D5 000A3    TSTL    4(R4)                                    ;
                                          E1  13 000A6    BEQL    4$                                       ;
            6E          00 00069780   8F 00989680   8F  7A 000A8 6$: EMUL    #10000000, #432000, #0, DELTA_5DAYS      ; 2009
                                          67  D5 000B5    TSTL    (R7)                                     ; 2010
                                          05  1^ 000B7    BNEQ    7$                                       ;
                                     04   A7  D5 000B9    TSTL    4(R7)                                    ;
                                          39  13 000BC    BEQL    11$                                      ;
                   10    AE       017C   C2        64  C3 000BE 7$: SUBL3   (R4), 380(R2), EXPIRATION        ; 2013
                                   14   AE  0180   C2  D0 000C5    MOVL    384(R2), EXPIRATION                      ;
                                   14   AE  04     A4  D9 000CB    SBWC    4(R4), EXPIRATION                        ;
                                   14   AE  04     A3  D1 000D0    CMPL    4(R3), EXPIRATION+4                      ; 2014
                                               08  14  000D5    BGTR    8$                                       ;
                                               20  12  000D7    BNEQ    12$                                      ; 2015
                                   10   AE          55  D1 000D9    CMPL    R5, EXPIRATION                           ; 2016
                                               1A  1F  000DD    BLSSU   12$                                      ;
                                   01D5 C2         02  88 000DF 8$: BISB2   #2, 469(R2)                          ; 2019
                                                0184 C2  D5 000E4    TSTL    388(R2)                              ; 2020
                                               06  12  000E8    BNEQ    9$                                       ;
```

```
                        0000'  CF   9F 000EA          PUSHAB  P.ABR                                                   ; 2024
                               04   11 000EE          BRB     10$
                        0000'  CF   9F 000F0 9$:      PUSHAB  P.ABT                                                   ; 2028
                    6A         01   FB 000F4 10$:     CALLS   #1, WRITE_OUTPUT
                               48   11 000F7 11$:     BRB     16$                                                     ; 2014
            08  AE      10    AE     6E   C3 000F9 12$:     SUBL3   DELTA_5DAYS, EXPIRATION, EXPIR_5DAYS            ; 2032
                     0C    AE   14    AE   D0 000FF          MOVL    EXPIRATION, EXPIR_5DAYS
                     0C    AE   04    AE   D9 00104          SBWC    DELTA_5DAYS, EXPIR_5DAYS
                     0C    AE   04    A3   D1 00109          CMPL    4(R3), EXPIR_5DAYS+4                            ; 2033
                               08   1A 0010E          BGTRU   13$
                               2F   12 00110          BNEQ    16$                                                     ; 2034
                     08    AE         55   D1 00112          CMPL    R5, EXPIR_5DAYS                                 ; 2035
                               29   1F 00116          BLSSU   16$
                        0184   C2   D5 00118 13$:     TSTL    388(R2)                                                 ; 2038
                               11   12 0011C          BNEQ    14$
                          10   AE   9F 0011E          PUSHAB  EXPIRATION                                              ; 2040
                          14   AE   9F 00121          PUSHAB  EXPIRATION                                              ; 2043
                    69         01   FB 00124          CALLS   #1, ASCIC_DAY_OF_WEEK
                               50   DD 00127          PUSHL   R0
                        0000'  CF   9F 00129          PUSHAB  P.ABV                                                   ; 2040
                               0F   11 0012D          BRB     15$
                          10   AE   9F 0012F 14$:     PUSHAB  EXPIRATION                                              ; 2045
                          14   AE   9F 00132          PUSHAB  EXPIRATION                                              ; 2048
                    69         01   FB 00135          CALLS   #1, ASCIC_DAY_OF_WEEK
                               50   DD 00138          PUSHL   R0
                        0000'  CF   9F 0013A          PUSHAB  P.ABW                                                   ; 2045
                    6B         03   FB 0013E 15$:     CALLS   #3, WRITE_FAO
                               66   D5 00141 16$:     TSTL    (R6)                                                    ; 2055
                               05   12 00143          BNEQ    17$
                           04  A6   D5 00145          TSTL    4(R6)
                               63   13 00148          BEQL    21$
                    50         68   D0 0014A 17$:     MOVL    UAF_RECORD, R0                                          ; 2058
            10  AE      0184   C0   64   C3 0014D          SUBL3   (R4), 388(R0), EXPIRATION
                     14    AE   0188  C0   D0 00154          MOVL    392(R0), EXPIRATION
                     14    AE   04    A4   D9 0015A          SBWC    4(R4), EXPIRATION
                     14    AE   04    A3   D1 0015F          CMPL    4(R3), EXPIRATION+4                             ; 2059
                               08   14 00164          BGTR    18$
                               14   12 00166          BNEQ    19$                                                     ; 2060
                     10    AE         55   D1 00168          CMPL    R5, EXPIRATION                                  ; 2061
                               0E   1F 0016C          BLSSU   19$
                     01D5  C0         04   88 0016E 18$:     BISB2   #4, 469(R0)                                     ; 2064
                        0000'  CF   9F 00173          PUSHAB  P.ABX                                                   ; 2067
                    6A         01   FB 00177          CALLS   #1, WRITE_OUTPUT
                               31   11 0017A          BRB     21$                                                     ; 2059
            08  AE      10    AE     6E   C3 0017C 19$:     SUBL3   DELTA_5DAYS, EXPIRATION, EXPIR_5DAYS            ; 2071
                     0C    AE   14    AE   D0 00182          MOVL    EXPIRATION, EXPIR_5DAYS
                     0C    AE   04    AE   D9 00187          SBWC    DELTA_5DAYS, EXPIR_5DAYS
                     0C    AE   04    A3   D1 0018C          CMPL    4(R3), EXPIR_5DAYS+4                            ; 2072
                               08   1A 00191          BGTRU   20$
                               18   12 00193          BNEQ    21$                                                     ; 2073
                     08    AE         55   D1 00195          CMPL    R5, EXPIR_5DAYS                                 ; 2074
                               12   1F 00199          BLSSU   21$
                          10   AE   9F 0019B 20$:     PUSHAB  EXPIRATION                                              ; 2076
                          14   AE   9F 0019E          PUSHAB  EXPIRATION                                              ; 2079
                    69         01   FB 001A1          CALLS   #1, ASCIC_DAY_OF_WEEK
                               50   DD 001A4          PUSHL   R0
                        0000'  CF   9F 001A6          PUSHAB  P.ABZ                                                   ; 2076
```

```
                        6B          03 FB 001AA          CALLS   #3, WRITE_FAO                  ;
                             BC     A8 9F 001AD 21$:     PUSHAB  UAF_RAB                        ; 2089
              00000000G  00         01 FB 001B0          CALLS   #1, SYS$UPDATE                 ;
                             BC     A8 9F 001B7 22$:     PUSHAB  UAF_RAB                        ; 2090
              00000000G  00         01 FB 001BA          CALLS   #1, SYS$DISCONNECT             ;
                           FF6C     C8 9F 001C1          PUSHAB  UAF_FAB                        ; 2091
              00000000G  00         01 FB 001C5          CALLS   #1, SYS$CLOSE                  ;
                                       04 001CC          RET                                    ; 2093

; Routine Size:  461 bytes,    Routine Base:  $CODE$ + 0958
```

```
;  1679          2094  1 GLOBAL ROUTINE set_ppd_prot =
;  1680          2095  1
;  1681          2096  1 !---
;  1682          2097  1 !
;  1683          2098  1 !       Change the page protection on the process permanent data area.
;  1684          2099  1 !
;  1685          2100  1 ! Inputs:
;  1686          2101  1 !
;  1687          2102  1 !       Access mode is executive.
;  1688          2103  1 !
;  1689          2104  1 !       AP = New page protection
;  1690          2105  1 !
;  1691          2106  1 ! Outputs:
;  1692          2107  1 !
;  1693          2108  1 !       routine = status (not signaled)
;  1694          2109  1 !---
;  1695          2110  1
;  1696          2111  2 BEGIN
;  1697          2112  2
;  1698          2113  2 BUILTIN AP;
;  1699          2114  2
;  1700          2115  2 LOCAL
;  1701          2116  2     range:      VECTOR [2];             ! Address range
;  1702          2117  2
;  1703          2118  2 range [0] = ppd;
;  1704          2119  2 range [1] = ppd + npd$c_length + lgi$c_length - 1;
;  1705          2120  2
;  1706        P 2121  2 RETURN $SETPRT(INADR = range,           ! Set page protection
;  1707          2122  2                  PROT  = .AP);          !  as specified
;  1708          2123  2
;  1709          2124  1 END;


                                                        .EXTRN  SYS$SETPRT

                                                        .ENTRY  SET_PPD_PROT, Save nothing      ; 2094
                                                        SUBL2   #4, SP                          ;
                                                        PUSHAB  PPD                             ; 2118
                                                        MOVAB   PPD+383, RANGE+4                ; 2119
                                                        CLRL    -(SP)                           ; 2122
                                                        PUSHL   AP                              ;
                                                        CLRQ    -(SP)                           ;
                                                        PUSHAB  RANGE                           ;
                                                        CALLS   #5, SYS$SETPRT                  ;
                                                        RET                                     ; 2124

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0B25
```

```
;  1711          2125  1 GLOBAL ROUTINE set_sysprv: NOVALUE =
;  1712          2126  1
;  1713          2127  1 !---
;  1714          2128  1 !
;  1715          2129  1 !       Enable SYSPRV privilege in the current privilege mask in
;  1716          2130  1 !       in order to obtain acess to system-wide files such as
;  1717          2131  1 !       SYSUAF and SYSALF.
;  1718          2132  1 !
;  1719          2133  1 ! Inputs:
;  1720          2134  1 !
;  1721          2135  1 !       None
;  1722          2136  1 !
;  1723          2137  1 ! Outputs:
;  1724          2138  1 !
;  1725          2139  1 !       None
;  1726          2140  1 !---
;  1727          2141  1
;  1728          2142  2 BEGIN
;  1729          2143  2
;  1730          2144  2 LOCAL
;  1731          2145  2     privmask:   BBLOCK [8];             ! Privilege mask
;  1732          2146  2
;  1733          2147  2 BIND
;  1734          2148  2     quadword = privmask: VECTOR;        ! Access as 2 longwords
;  1735          2149  2
;  1736          2150  2 quadword [0] = 0;                       ! Initialize mask
;  1737          2151  2 quadword [1] = 0;
;  1738          2152  2 privmask [prv$v_sysprv] = true;        ! Set SYSPRV bit in mask
;  1739          2153  2
;  1740        P 2154  2 $SETPRV(PRVADR = privmask,              ! Disable SYSPRV privilege
;  1741          2155  2         ENBFLG = 1);
;  1742          2156  2
;  1743          2157  1 END


                                                        .EXTRN  SYS$SETPRV

                                              0000 00000        .ENTRY  SET_SYSPRV, Save nothing                ; 2125
                                         5E     04   C2 00002  SUBL2   #4, SP                                  ;
                                                7E   D4 00005  CLRL    QUADWORD                                ; 2150
                                           04   AE   D4 00007  CLRL    QUADWORD+4                              ; 2151
                                     03    AE      10   88 0000A  BISB2   #16, PRIVMASK+3                      ; 2152
                                                7E   7C 0000E  CLRQ    -(SP)                                   ; 2155
                                           08   AE   9F 00010  PUSHAB  PRIVMASK                               ;
                                                01   DD 00013  PUSHL   #1                                     ;
                                00000000G  00        04   FB 00015  CALLS   #4, SYS$SETPRV                    ;
                                                     04 0001C  RET                                           ; 2157

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 0B49
```

```
;  1745          2158  1 GLOBAL ROUTINE clear_sysprv: NOVALUE =
;  1746          2159  1
;  1747          2160  1 !---
;  1748          2161  1 !
;  1749          2162  1 !       Disable SYSPRV privilege which is given because the program
;  1750          2163  1 !       is normally installed with it in order to obtain acess to
;  1751          2164  1 !       system-wide files such as SYSUAF and SYSALF.  The privilege
;  1752          2165  1 !       is not disabled if the process is authorized to have it.
;  1753          2166  1 !
;  1754          2167  1 ! Inputs:
;  1755          2168  1 !
;  1756          2169  1 !       None
;  1757          2170  1 !
;  1758          2171  1 ! Outputs:
;  1759          2172  1 !
;  1760          2173  1 !       None
;  1761          2174  1 !---
;  1762          2175  1
;  1763          2176  2 BEGIN
;  1764          2177  2
;  1765          2178  2 LOCAL
;  1766          2179  2     privmask:   BBLOCK [8];             ! Privilege mask
;  1767          2180  2
;  1768          2181  2 BIND
;  1769          2182  2     quadword = privmask: VECTOR;        ! Access as 2 longwords
;  1770          2183  2
;  1771        P 2184  2 $SETPRV(PRVPRV = privmask,              ! Get authorized privilege mask
;  1772          2185  2         PRMFLG = 1);
;  1773          2186  2
;  1774          2187  2 IF .privmask [prv$v_sysprv]           ! If authorized to have SYSPRV,
;  1775          2188  2 THEN
;  1776          2189  2     RETURN;                           ! then do nothing
;  1777          2190  2
;  1778          2191  2 quadword [0] = 0;                     ! Initialize mask
;  1779          2192  2 quadword [1] = 0;
;  1780          2193  2 privmask [prv$v_sysprv] = true;       ! Set SYSPRV bit in mask
;  1781          2194  2
;  1782        P 2195  2 $SETPRV(PRVADR = privmask,            ! Disable SYSPRV privilege
;  1783          2196  2         ENBFLG = 0);
;  1784          2197  2
;  1785          2198  1 END;


                                                0004 00000          .ENTRY  CLEAR_SYSPRV, Save R2                   ; 2158
                          52 00000000G      00   9E 00002          MOVAB   SYS$SETPRV, R2                          ;
                          5E                08   C2 00009          SUBL2   #8, SP                                  ;
                                            5E   DD 0000C          PUSHL   SP                                      ; 2185
                                            01   DD 0000E          PUSHL   #1                                      ;
                                            7E   7C 00010          CLRQ    -(SP)                                   ;
                          62                04   FB 00012          CALLS   #4, SYS$SETPRV                          ;
                10        03      AE        04   E0 00015          BBS     #4, PRIVMASK+3, 1$                      ; 2187
                                            6E   7C 0001A          CLRQ    QUADWORD                                ; 2191
                          03      AE        10   88 0001C          BISB2   #16, PRIVMASK+3                         ; 2193
                                            7E   7C 00020          CLRQ    -(SP)                                   ; 2196
```

```
                    08  AE  9F 00022          PUSHAB  PRIVMASK                                        ;
                        7E  D4 00025          CLRL    -(SP)                                           ;
                62      04  FB 00027          CALLS   #4, SYS$SETPRV                                  ;
                        04 0002A 1$:          RET                                                     ; 2198

; Routine Size:  43 bytes,    Routine Base:  $CODE$ + 0B66
```

```
;  1787          2199  1 GLOBAL ROUTINE clear_log_io: NOVALUE =
;  1788          2200  1
;  1789          2201  1 !---
;  1790          2202  1 !
;  1791          2203  1 !       Disable LOG_IO privilege which is given because the program
;  1792          2204  1 !       is normally installed with it in order to force terminal
;  1793          2205  1 !       hangups in case of error.  The privilege is not disabled if
;  1794          2206  1 !       the process is authorized to have it.
;  1795          2207  1 !
;  1796          2208  1 ! Inputs:
;  1797          2209  1 !
;  1798          2210  1 !       None
;  1799          2211  1 !
;  1800          2212  1 ! Outputs:
;  1801          2213  1 !
;  1802          2214  1 !       None
;  1803          2215  1 !---
;  1804          2216  1
;  1805          2217  2 BEGIN
;  1806          2218  2
;  1807          2219  2 LOCAL
;  1808          2220  2     privmask:   BBLOCK [8];             ! Privilege mask
;  1809          2221  2
;  1810          2222  2 BIND
;  1811          2223  2     quadword = privmask: VECTOR;        ! Access as 2 longwords
;  1812          2224  2
;  1813        P 2225  2 $SETPRV(FRVPRV = privmask,              ! Get authorized privilege mask
;  1814          2226  2         PRMFLG = 1);
;  1815          2227  2
;  1816          2228  2 IF .privmask [prv$v_log_io]            ! If authorized to have LOG_IO,
;  1817          2229  2 THEN
;  1818          2230  2     RETURN;                           ! then do nothing
;  1819          2231  2
;  1820          2232  2 quadword [0] = 0;                     ! Initialize mask
;  1821          2233  2 quadword [1] = 0;
;  1822          2234  2 privmask [prv$v_log_io] = true;       ! Set LOG_IO bit in mask
;  1823          2235  2
;  1824        P 2236  2 $SETPRV(PRVADR = privmask,            ! Disable LOG_IO privilege
;  1825          2237  2         ENBFLG = 0);
;  1826          2238  2
;  1827          2239  1 END;


                                  0004 00000         .ENTRY  CLEAR_LOG_IO, Save R2                   ; 2199
                52 00000000G   00  9E 00002         MOVAB   SYS$SETPRV, R2
                5E             08  C2 00009         SUBL2   #8, SP
                               5E  DD 0000C         PUSHL   SP                                      ; 2226
                               01  DD 0000E         PUSHL   #1
                               7E  7C 00010         CLRQ    -(SP)
                62             04  FB 00012         CALLS   #4, SYS$SETPRV
                               6E  95 00015         TSTB    PRIVMASK                                ; 2228
                               10  19 00017         BLSS    1$
                               6E  7C 00019         CLRQ    QUADWORD                                ; 2232
                6E        80   8F  88 0001B         BISB2   #128, PRIVMASK                          ; 2234
```

```
                         7E  7C 0001F          CLRQ    -(SP)                                   ; 2237
                      08 AE  9F 00021          PUSHAB  PRIVMASK                                ;
                         7E  D4 00024          CLRL    -(SP)                                   ;
                   62    04  FB 00026          CALLS   #4, SYS$SETPRV                          ;
                         04  00029 1$:         RET                                             ; 2239

; Routine Size:  42 bytes,    Routine Base:  $CODE$ + 0B91
```

```
;  1829     2240  1 GLOBAL ROUTINE validate_uafrec (username, password1, password2): NOVALUE =
;  183C     2241  1
;  1851     2242  1 !---
;  1852     2243  1 !
;  1853     2244  1 !       Lookup the specified username in the authorization file
;  1854     2245  1 !       and validate the passwords. Apply breakin detection / evasion
;  1855     2246  1 !       as appropriate.
;  1856     2247  1 !
;  1857     2248  1 ! Inputs:
;  1858     2249  1 !
;  1859     2250  1 !       username = Address of descriptor of username
;  1860     2251  1 !       password1 = Address of descriptor of primary password
;  1861     2252  1 !       password2 = Address of descriptor of secondary password
;  1862     2253  1 !
;  1863     2254  1 ! Outputs:
;  1864     2255  1 !
;  1865     2256  1 !       uaf_record = Address of UAF record for user
;  1866     2257  1 !---
;  1867     2258  1
;  1868     2259  2 BEGIN
;  1869     2260  2
;  1870     2261  2 MAP
;  1871     2262  2     username:   REF VECTOR,              ! Address of username descriptor
;  1872     2263  2     password1:  REF VECTOR,              ! Address of password #1 descriptor
;  1873     2264  2     password2:  REF VECTOR;              ! Address of password #2 descriptor
;  1874     2265  2
;  1875     2266  2 LOCAL
;  1876     2267  2     status,
;  1877     2268  2     arglist:    VECTOR [2];             ! Argument list for CIA_SCAN
;  1878     2269  2
;  1879     2270  2 status = get_uafrec(.username);          ! Read the UAF record
;  1880     2271  2
;  1881     2272  2 IF .uaf_record NEQ 0                    ! Validate both passwords
;  1882     2273  2 THEN
;  1883     2274  3     BEGIN
;  1884     2275  4     IF NOT (status = lgi$check_pass(.password1, .uaf_record, 0))
;  1885     2276  3     THEN
;  1886     2277  4         CH$COPY((fail_password [0] = MINU(.password1 [0],       ! Save password
;  1887     2278  3                                          nsa$s_pkt_password)),
;  1888     2279  3                 .password1 [1],                                 !  for auditing
;  1889     2280  3                 0,
;  1890     2281  3                 nsa$s_pkt_password,
;  1891     2282  3                 .fail_password [1])
;  1892     2283  3     ELSE
;  1893     2284  4         BEGIN
;  1894     2285  4         IF .bblock [uaf_record [uaf$q_pwd2],0,0,32,0] NEQ 0
;  1895     2286  4         OR .bblock [uaf_record [uaf$q_pwd2],4,0,32,0] NEQ 0
;  1896     2287  4         THEN
;  1897     2288  5             IF NOT (status = lgi$check_pass(.password2, .uaf_record, 1))
;  1898     2289  4             THEN
;  1899     2290  5                 CH$COPY((fail_password [0] = MINU(.password2 [0],
;  1900     2291  4                                                  nsa$s_pkt_password)),
;  1901     2292  4                         .password2 [1],
;  1902     2293  4                         0,
;  1903     2294  4                         nsa$s_pkt_password,
;  1904     2295  4                         .fail_password [1]);
;  1905     2296  3         END;
```

```
;  1886      2297  3     IF .status EQL -4                       . Special invalid password status?
;  1887      2298  3     THEN status = lgi$_invpwd;
;  1888      2299  3     IF NOT .status
;  1889      2300  3     THEN update_uaf_record();               ! Increment login failure count
;  1890      2301  2     END;
;  1891      2302  2
;  1892      2303  2 arglist[0] = 1;
;  1893      2304  2 IF NOT .status                               ! If error detected,
;  1894      2305  2 THEN
;  1895      2306  3     BEGIN
;  1896      2307  3     arglist[1] = 0;                         ! Call CIA_SCAN, with 0,
;  1897    P 2308  3     IF NOT $CMKRNL(ROUTIN = cia_scan,       ! which means "check this
;  1898      2309  4                    ARGLST = arglist)       ! for a suspect".
;  1899      2310  3     THEN
;  1900      2311  3         security_audit(nsa$k_rectyp_logb); ! Audit the breakin
;  1901      2312  3     END
;  1902      2313  2 ELSE
;  1903      2314  3     BEGIN
;  1904      2315  3     arglist[1] = 1;                         ! Call CIA_SCAN, to see if
;  1905    P 2316  3     IF NOT $CMKRNL(ROUTIN = cia_scan,       ! this might be an intruder.
;  1906      2317  4                    ARGLST = arglist)
;  1907      2318  3     THEN status = lgi$_evade;
;  1908      2319  2     END;
;  1909      2320  2
;  1910      2321  2 ppd [ppd$l_ststatus] = .status;            ! set final job status
;  1911      2322  2
;  1912      2323  2 IF NOT .status
;  1913      2324  2 THEN SIGNAL_STOP(lgi$_notvalid);           ! say it was a password problem
;  1914      2325  2
;  1915      2326  1 END;
```

```
                                07FC 00000          .ENTRY  VALIDATE_UAFREC, Save R2,R3,R4,R5,R6,R7,R8,-  ; 2240
                                                            R9,R10                                        ;
                    5A 00000000G   00   9E 00002    MOVAB   SYS$CMKRNL, R10                               ;
                    59 00000000G   00   9E 00009    MOVAB   CIA_SCAN, R9                                  ;
                    58 00000000G   00   9E 00010    MOVAB   LGI$CHECK_PASS, R8                            ;
                    57      0000'  CF   9E 00017    MOVAB   UAF_RECORD, R7                                ;
                    5E             08   C2 0001C    SUBL2   #8, SP                                        ;
                                04 AC   DD 0001F    PUSHL   USERNAME                                      ; 2270
              0000V CF             01   FB 00022    CALLS   #1, GET_UAFREC                                ;
                    56             50   D0 00027    MOVL    R0, STATUS                                    ;
                    50             67   D0 0002A    MOVL    UAF_RECORD, R0                                ; 2272
                                   70   13 0002D    BEQL    7$                                            ;
                                   7E   D4 0002F    CLRL    -(SP)                                         ; 2275
                                   50   DD 00031    PUSHL   R0                                            ;
                    52         08 AC   D0 00033    MOVL    PASSWORD1, R2                                 ;
                                   52   DD 00037    PUSHL   R2                                            ;
                    68             03   FB 00039    CALLS   #3, LGI$CHECK_PASS                            ;
                    56             50   D0 0003C    MOVL    R0, STATUS                                    ;
                    0A             56   E8 0003F    BLBS    STATUS, 1$                                    ;
                    50             62   D0 00042    MOVL    (R2), R0                                      ; 2277
                    1F             50   D1 00045    CMPL    R0, #31                                       ;
                                   2E   1A 00048    BGTRU   3$                                            ;
```

```
                                 2F  11 0004A          BRB     4$                                                      ;
                           51    67  D0 0004C 1$:      MOVL    UAF_RECORD, R1                                          ; 2285
                           50 015C   C1  9E 0004F      MOVAB   348(R1), R0                                             ;
                                 60  D5 00054          TSTL    (R0)                                                    ;
                                 05  12 00056          BNEQ    2$                                                      ;
                              04 A0  D5 00058          TSTL    4(R0)                                                   ; 2286
                                 2A  13 0005B          BEQL    5$                                                      ;
                                 01  DD 0005D 2$:      PUSHL   #1                                                      ; 2288
                                 51  DD 0005F          PUSHL   R1                                                      ;
                           52    0C  AC  D0 00061      MOVL    PASSWORD2, R2                                           ;
                                 52  DD 00065          PUSHL   R2                                                      ;
                           68        03  FB 00067      CALLS   #3, LGI$CHECK_PASS                                      ;
                           56        50  D0 0006A      MOVL    R0, STATUS                                              ;
                           17        56  E8 0006D      BLBS    STATUS, 5$                                              ;
                           50        62  D0 00070      MOVL    (R2), R0                                                ; 2290
                           1F        50  D1 00073      CMPL    R0, #31                                                 ;
                                     03  1B 00076      BLEQU   4$                                                      ;
                           50        1F  D0 00078 3$:   MOVL    #31, R0                                                 ;
                        30 A7        50  D0 0007B 4$:   MOVL    R0, FAIL_PASSWORD                                       ;
1F                00    04 B2        50  2C 00080      MOVC5   R0, @4(R2), #0, #31, @FAIL_PASSWORD+4                   ; 2295
                              34 B7      00085                                                                         ;
                FFFFFFFC      8F        56  D1 00087 5$:   CMPL    STATUS, #-4                                             ; 2297
                                        07  12 0008E      BNEQ    6$                                                      ;
                           56 00000000G   8F  D0 00090      MOVL    #LGI$_INVPWD, STATUS                                    ; 2298
                           05        56  E8 00097 6$:   BLBS    STATUS, 7$                                              ; 2299
                     FCFE  CF              00  FB 0009A      CALLS   #0, UPDATE_UAF_RECORD                           ; 2300
                           6E        01  D0 0009F 7$:   MOVL    #1, ARGLIST                                             ; 2303
                           18        56  E8 000A2      BLBS    STATUS, 8$                                              ; 2304
                                 04  AE  D4 000A5      CLRL    ARGLIST+4                                               ; 2307
                              4200   8F  BB 000A8      PUSHR   #^M<R9,SP>                                              ; 2309
                           6A        02  FB 000AC      CALLS   #2, SYS$CMKRNL                                          ;
                           20        50  E8 000AF      BLBS    R0, 9$                                                  ;
                                     04  DD 000B2      PUSHL   #4                                                      ; 2311
                00000000G  00        01  FB 000B4      CALLS   #1, SECURITY_AUDIT                                      ;
                                     15  11 000BB      BRB     9$                                                      ; 2304
                     04    AE        01  D0 000BD 8$:   MOVL    #1, ARGLIST+4                                           ; 2315
                                 4200   8F  BB 000C1      PUSHR   #^M<R9,SP>                                              ; 2317
                           6A        02  FB 000C5      CALLS   #2, SYS$CMKRNL                                          ;
                           07        50  E8 000C8      BLBS    R0, 9$                                                  ;
                           56 00000000G   8F  D0 000CB      MOVL    #LGI$_EVADE, STATUS                                     ; 2318
                00000000G  00        56  D0 000D2 9$:   MOVL    STATUS, PPD+24                                          ; 2321
                           0D        56  E8 000D9      BLBS    STATUS, 10$                                             ; 2323
                              00000000G   8F  DD 000DC      PUSHL   #LGI$_NOTVALID                                          ; 2324
                00000000G  00        01  FB 000E2      CALLS   #1, LIB$STOP                                            ;
                                     04  000E9 10$:    RET                                                             ; 2326

; Routine Size:  234 bytes,    Routine Base:  $CODE$ + 0BBB
```

```
;  1917          2327  1 GLOBAL ROUTINE get_uafrec (username) =
;  1918          2328  1
;  1919          2329  1 !---
;  1920          2330  1 !
;  1921          2331  1 !       Look up the specified username in the authorization file
;  1922          2332  1 !       and use it without checking any password.
;  1923          2333  1 !
;  1924          2334  1 ! Inputs:
;  1925          2335  1 !
;  1926          2336  1 !       username = Optional address of descriptor of username to look for.
;  1927          2337  1 !
;  1928          2338  1 ! Outputs:
;  1929          2339  1 !
;  1930          2340  1 !       uaf_record = Address of UAF record for user
;  1931          2341  1 !
;  1932          2342  1 ! Return Value:
;  1933          2343  1 !
;  1934          2344  1 !       true if UAF record successfully read OR if UAF is unavailable
;  1935          2345  1 !               and the terminal is OPA0:
;  1936          2346  1 !       false otherwise
;  1937          2347  1 !---
;  1938          2348  1
;  1939          2349  2 BEGIN
;  1940          2350  2
;  1941          2351  2 BUILTIN
;  1942          2352  2     NULLPARAMETER;                    ! True if no parameter specified
;  1943          2353  2
;  1944          2354  2 MAP
;  1945          2355  2     username:   REF VECTOR;           ! Address of username descriptor
;  1946          2356  2
;  1947          2357  2 EXTERNAL
;  1948          2358  2     ctl$t_username:     BBLOCK;       ! Current username
;  1949          2359  2
;  1950          2360  2 LOCAL
;  1951          2361  2     status,
;  1952          2362  2     uaf_desc:   VECTOR [2],          ! Descriptor of UAF record buffer
;  1953          2363  2     desc:       VECTOR [2];          ! Username descriptor
;  1954          2364  2
;  1955          2365  2 IF NOT NULLPARAMETER(1)              ! If parameter specified,
;  1956          2366  2 THEN
;  1957          2367  3     BEGIN
;  1958          2368  3     desc [0] = .username [0];        ! then use given username
;  1959          2369  3     desc [1] = .username [1];
;  1960          2370  3     END
;  1961          2371  2 ELSE
;  1962          2372  3     BEGIN
;  1963          2373  3     desc [0] = jib$s_username;       ! Setup descriptor of current username
;  1964          2374  3     desc [1] = ctl$t_username;
;  1965          2375  2     END;
;  1966          2376  2
;  1967          2377  2 uaf_desc [0] = uaf$k_length;        ! Setup descriptor of buffer
;  1968          2378  2 status = LIB$GET_VM(uaf_desc [0], uaf_desc [1]); ! Allocate buffer
;  1969          2379  2 IF NOT .status                       ! If error detected,
;  1970          2380  2 THEN
;  1971          2381  2     SIGNAL_STOP(.status);           ! then signal fatal error
;  1972          2382  2
;  1973          2383  2 status = lgi$searchuser(desc,       ! Search UAF for authorization record
```

```
;  1974     2384  2                          0,                  ! Skip password checking
;  1975     2385  2                          uaf_desc,           ! Address of descriptor of buffer
;  1976     2386  2                          uaf_fab,            ! FAB for UAF access
;  1977     2387  2                          uaf_rab);           ! RAB for UAF access
;  1978     2388  2 IF .status EQL -2                            ! Special invalid username status?
;  1979     2389  2 THEN status = lgi$_nosuchuser;
;  1980     2390  2
;  1981     2391  2 IF NOT .status                               ! If error detected,
;  1982     2392  2 THEN
;  1983     2393  3     BEGIN
;  1984     2394  3     uaf_record = 0;                          ! Show no user record
;  1985     2395  3     pcb_sts [$BITPOSITION(pcb$v_secaudit)] = 0; !  and no mandatory auditing
;  1986     2396  3     ppd [ppd$l_loststatus] = .status;        ! Initialize loststatus
;  1987     2397  3
;  1988     2398  3     IF .status EQL lgi$_nosuchuser           ! If invalid username
;  1989     2399  3     THEN RETURN .status;                     ! return the error
;  1990     2400  3
;  1991     2401  3     !
;  1992     2402  3     ! If user is at the console, allow login.
;  1993     2403  3     ! Otherwise, signal a fatal file access error.
;  1994     2404  3     !
;  1995     2405  3     IF CH$NEQ (.phy_term_name[0], .phy_term_name[1], 6, UPLIT BYTE('_OPA0:'))
;  1996     2406  3     THEN
;  1997     2407  3         SIGNAL_STOP(lgi$_fileacc,0,.status);
;  1998     2408  3     END
;  1999     2409  3
;  2000     2410  2 ELSE
;  2001     2411  3     BEGIN
;  2002     2412  3     uaf_record = .uaf_desc [1];              ! Mark UAF record present
;  2003     2413  3     pcb_sts [$BITPOSITION(pcb$v_secaudit)] = 0; ! Guess at no mandatory audit
;  2004     2414  3     IF .uaf_record [uaf$v_audit]             ! If UAF says mandatory auditing
;  2005     2415  3     THEN pcb_sts [$BITPOSITION(pcb$v_secaudit)] = 1; !  then do auditing
;  2006     2416  2     END;
;  2007     2417  2
;  2008     2418  2 RETURN 1;
;  2009     2419  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                    3A  30  41  50  4F  5F  0033D P.ACA:  .ASCII  \_OPA0:\                                  ;


                                                .PSECT  $CODE$,NOWRT,2                                      ;

                                      00FC 00000        .ENTRY  GET_UAFREC, Save R2,R3,R4,R5,R6,R7          ; 2327
                57 00000000G      8F  D0 00002        MOVL    #LGI$_NOSUCHUSER, R7                        ;
                56 00000000G      00  9E 00009        MOVAB   LIB$STOP, R6                                ;
                55      0000'     CF  9E 00010        MOVAB   UAF_RECORD, R5                              ;
                5E                10  C2 00015        SUBL2   #16, SP                                     ;
                                  6C  95 00018        TSTB    (AP)                                        ; 2365
                                  0E  13 0001A        BEQL    1$                                          ;
                          04      AC  D5 0001C        TSTL    4(AP)                                       ;
                                  09  13 0001F        BEQL    1$                                          ;
                50        04      AC  D0 00021        MOVL    USERNAME, R0                                ; 2368
```

```
                        6E                  60  7D 00025          MOVQ    (R0), DESC                                          ;
                                            0B  11 00028          BRB     2$                                                  ; 2365
                        6E                  0C  D0 0002A 1$:      MOVL    #12, DESC                                           ; 2373
                     04 AE 00000000G        00  9E 0002D          MOVAB   CTL$T_USERNAME, DESC+4                              ; 2374
                     08 AE      0584        8F  3C 00035 2$:      MOVZWL  #1412, UAF_DESC                                     ; 2377
                           AE      0C       9F  0003B             PUSHAB  UAF_DESC+4                                          ; 2378
                           AE      0C       9F  0003E             PUSHAB  UAF_DESC
                00000000G  00               02  FB 00041          CALLS   #2, LIB$GET_VM
                           54               50  D0 00048          MOVL    R0, STATUS
                           05               54  E8 0004B          BLBS    STATUS, 3$                                          ; 2379
                                            54  DD 0004E          PUSHL   STATUS                                              ; 2381
                           66               01  FB 00050          CALLS   #1, LIB$STOP
                              BC        A5  9F  00053 3$:      PUSHAB  UAF_RAB                                             ; 2383
                            FF6C        C5  9F  00056          PUSHAB  UAF_FAB
                              10        AE  9F  0005A          PUSHAB  UAF_DESC
                                        7E  D4  0005D          CLRL    -(SP)
                              10        AE  9F  0005F          PUSHAB  DESC
                00000000G  00               05  FB 00062          CALLS   #5, LGI$SEARCHUSER
                           54               50  D0 00069          MOVL    R0, STATUS
                FFFFFFFE   8F               54  D1 0006C          CMPL    STATUS, #-2                                         ; 2388
                                            03  12 00073          BNEQ    4$
                           54               57  D0 00075          MOVL    R7, STATUS                                          ; 2389
                     FF53  C5               08  8A 00078 4$:      BICB2   #8, PCB_STS+3                                      ; 2395
                           2D               54  E8 0007D          BLBS    STATUS, 6$                                          ; 2391
                                            65  D4 00080          CLRL    UAF_RECORD                                          ; 2394
                00000000G  00               54  D0 00082          MOVL    STATUS, PPD+24                                      ; 2396
                           57               54  D1 00089          CMPL    STATUS, R7                                          ; 2398
                                            04  12 0008C          BNEQ    5$
                           50               54  D0 0008E          MOVL    STATUS, R0                                          ; 2399
                                            04  00091             RET
        06        00        44  B5        40  A5  2D 00092 5$:      CMPC5   PHY_TERM_NAME, @PHY_TERM_NAME+4, #0, #6, -          ; 2405
                                         0000' CF      00099             P.ACA
                                            21  13 0009C          BEQL    7$
                                            54  DD 0009E          PUSHL   STATUS                                              ; 2407
                                            7E  D4 000A0          CLRL    -(SP)
                              00000000G     8F  DD 000A2          PUSHL   #LGI$_FILEACC
                                        66      03  FB 000A8          CALLS   #3, LIB$STOP
                                            12  11 000AB          BRB     7$                                                  ; 2391
                                        65  0C  AE  D0 000AD 6$:      MOVL    UAF_DESC+4, UAF_RECORD                          ; 2412
                                        50      65  D0 000B1          MOVL    UAF_RECORD, R0                                  ; 2414
                                 05  01D5  C0      03  E1 000B4          BBC     #3, 469(R0), 7$
                                     FF53  C5      08  88 000BA          BISB2   #8, PCB_STS+3                               ; 2415
                                        50      01  D0 000BF 7$:      MOVL    #1, R0                                          ; 2418
                                                04  000C2             RET                                                 ; 2419

; Routine Size:  195 bytes,    Routine Base:  $CODE$ + 0CA5
```

```
;  2011     2420  1 GLOBAL ROUTINE logout_message: NOVALUE =
;  2012     2421  1
;  2013     2422  1 !---
;  2014     2423  1 !
;  2015     2424  1 !       Write the logout message to the output stream.
;  2016     2425  1 !
;  2017     2426  1 ! Inputs:
;  2018     2427  1 !
;  2019     2428  1 !       full_logout = True if should output full display, else brief
;  2020     2429  1 !
;  2021     2430  1 ! Outputs:
;  2022     2431  1 !
;  2023     2432  1 !       None
;  2024     2433  1 !---
;  2025     2434  1
;  2026     2435  2 BEGIN
;  2027     2436  2
;  2028     2437  2 BUILTIN EMUL;
;  2029     2438  2
;  2030     2439  2 EXTERNAL ROUTINE
;  2031     2440  2     lib$subx;                          ! Subtract 2 quadword numbers
;  2032     2441  2
;  2033     2442  2 OWN
;  2034     2443  2     bufio,                             ! Number of buffered I/O's
;  2035     2444  2     wspeak,                            ! Peak working set size
;  2036     2445  2     dirio,                             ! Number of direct I/O's
;  2037     2446  2     virtpeak,                          ! Peak page file size
;  2038     2447  2     pageflts,                          ! Number of page faults
;  2039     2448  2     volumes;                           ! Number of volumes mounted
;  2040     2449  2
;  2041     2450  2 LOCAL
;  2042     2451  2     cputim:     VECTOR [2],            ! Total CPU time
;  2043     2452  2     logintim:   VECTOR [2],            ! Login date/time
;  2044     2453  2     curtime:    VECTOR [2],            ! Current date/time
;  2045     2454  2     username:   VECTOR [2],            ! Descriptor of username
;  2046     2455  2     procname:   VECTOR [2],            ! Descriptor of process name
;  2047     2456  2     username_buffer: VECTOR [16,BYTE],
;  2048     2457  2     procname_buffer: VECTOR [16,BYTE],
;  2049     2458  2     jpi_list:   BBLOCK [128];          ! GETJPI item list
;  2050     2459  2
;  2051     2460  2 username [1] = username_buffer;
;  2052     2461  2 procname [1] = procname_buffer;
;  2053     2462  2
;  2054     2463  2 jpi_list [0,0,16,0] = %ALLOCATION(username_buffer);
;  2055     2464  2 jpi_list [2,0,16,0] = jpi$_username;
;  2056     2465  2 jpi_list [4,0,32,0] = username_buffer;
;  2057     2466  2 jpi_list [8,0,32,0] = username [0];
;  2058     2467  2 jpi_list [12,0,16,0] = %ALLOCATION(procname_buffer);
;  2059     2468  2 jpi_list [14,0,16,0] = jpi$_prcnam;
;  2060     2469  2 jpi_list [16,0,32,0] = procname_buffer;
;  2061     2470  2 jpi_list [20,0,32,0] = procname [0];
;  2062     2471  2 jpi_list [24,0,32,0] = 0;
;  2063     2472  2
;  2064     2473  2 $GETJPI(ITMLST = jpi_list);          ! Get username and process name
;  2065     2474  2
;  2066     2475  2 IF .ppd [ppd$v_mode]                 ! If batch job,
;  2067     2476  2 THEN
```

```
;2068      2477  2     write_fao(UPLIT BYTE(%ASCIC '  .AS jcb terminated at !%D'),
;2069      2478  2         username,
;2070      2479  2         0)
;2071      2480  2 ELSE IF NOT .subprocess                ! else if main interactive job,
;2072      2481  2 THEN
;2073      2482  2     write_fao(UPLIT BYTE(%ASCIC '  !AS logged out at !%D'),
;2074      2483  2         username,
;2075      2484  2         0)
;2076      2485  2 ELSE                                   ! Else, if subprocess
;2077      2486  2     write_fao(UPLIT BYTE(%ASCIC '  Process !AS logged out at !%D'),
;2078      2487  2         procname,
;2079      2488  2         0);
;2080      2489  2
;2081      2490  2 IF NOT .full_logout                    ! If brief message desired,
;2082      2491  2 THEN
;2083      2492  2     RETURN;                            ! then skip rest
;2084      2493  2
;2085    P 2494  2 setup_jpidvi_list(jpi_list,
;2086    P 2495  2         jpi$_bufio,4,bufio,
;2087    P 2496  2         jpi$_wspeak,4,wspeak,
;2088    P 2497  2         jpi$_dirio,4,dirio,
;2089    P 2498  2         jpi$_virtpeak,4,virtpeak,
;2090    P 2499  2         jpi$_pagefIts,4,pagefIts,
;2091    P 2500  2         jpi$_volumes,4,volumes,
;2092    P 2501  2         jpi$_cputim,8,cputim,
;2093    P 2502  2         jpi$_logintim,8,logintim,
;2094      2503  2         0,0,0);
;2095      2504  2
;2096      2505  2 $GETJPI(ITMLST = jpi_list);             ! Get statistics
;2097      2506  2
;2098      2507  2 EMUL(%REF(-100000),cputim,%REF(0),cputim); ! Convert MS to 100 NS units
;2099      2508  2                                        ! and negate to convert to delta time
;2100      2509  2
;2101      2510  2 $GETTIM(TIMADR = curtime);             ! Get current time
;2102      2511  2
;2103      2512  2 LIB$SUBX(logintim,curtime,logintim);   ! Subtract giving negative elapsed time
;2104      2513  2
;2105      2514  2 write_fao(UPLIT BYTE(%ASCIC '!/  Accounting information:'));
;2106      2515  2 write_fao(UPLIT BYTE(%ASCIC
;2107      2516  2         '  Buffered I/O count:!13UL      Peak working set size:!6UL'),
;2108      2517  2         .bufio, .wspeak);
;2109      2518  2 write_fao(UPLIT BYTE(%ASCIC
;2110      2519  2         '  Direct I/O count:  !13UL      Peak page file size:  !6UL'),
;2111      2520  2         .dirio, .virtpeak);
;2112      2521  2 write_fao(UPLIT BYTE(%ASCIC
;2113      2522  2         '  Page faults:       !13UL      Mounted volumes:      !6UL'),
;2114      2523  2         .pagefIts, .volumes);
;2115      2524  2 write_fao(UPLIT BYTE(%ASCIC
;2116      2525  2         '  Charged CPU time:  !%D   Elapsed time:  !%D'),
;2117      2526  2         cputim, logintim);
;2118      2527  2
;2119      2528  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
6D 72 65 74 20 62 6F 6A 20 53 41 21 20 20 1B 00343 P.ACB:  .ASCII  <27>\  !AS job terminated at !%D\
      44 25 21 20 74 61 20 64 65 74 61 6E 69 00352
6F 20 64 65 67 67 6F 6C 20 53 41 21 20 20 17 0035F P.ACC:  .ASCII  <23>\  !AS logged out at !%D\
               44 25 21 20 74 61 20 74 75 0036E
20 53 41 21 20 73 73 65 63 6F 72 50 20 20 1F 00377 P.ACD:  .ASCII  <31>\  Process !AS logged out at !%D\
21 20 74 61 20 74 75 6F 20 54 65 67 67 6F 6C 00386
                                  44 25 00395
67 6E 69 74 6E 75 6F 63 63 41 20 20 2F 21 1B 00397 P.ACE:  .ASCII  <27>\!/  Accounting information:\
      3A 6E 6F 69 74 61 6D 72 6F 66 6E 69 20 003A6
4F 2F 49 20 64 65 72 65 66 66 75 42 20 20 3A 003B3 P.ACF:  .ASCII  \:  Buffered I/O count:!13UL      Peak wo\
20 20 20 4C 55 33 31 21 3A 74 6E 75 6F 63 20 003C2
               6F 77 20 6B 61 65 50 20 20 20 003D1
3A 65 7A 69 73 20 74 65 73 20 67 6E 69 6B 72 003DB          .ASCII  \rking set size:!6UL\
                              4C 55 36 21 003EA
63 20 4F 2F 49 20 74 63 65 72 69 44 20 20 3A 003EE P.ACG:  .ASCII  \:  Direct I/O count:  !13UL      Peak pa\
20 20 20 4C 55 33 31 21 20 20 3A 74 6E 75 6F 003FD
               61 70 20 6B 61 65 50 20 20 20 0040C
20 20 3A 65 7A 69 73 20 65 6C 69 66 20 65 67 00416          .ASCII  \ge file size:  !6UL\
                              4C 55 36 21 00425
3A 73 74 6C 75 61 66 20 65 67 61 50 20 20 3A 00429 P.ACH:  .ASCII  \:  Page faults:       !13UL      Mounted\
20 20 20 4C 55 33 31 21 20 20 20 20 20 20 20 00438
               64 65 74 6E 75 6F 4D 20 20 20 00447
20 20 20 20 20 20 3A 73 65 6D 75 6C 6F 76 20 00451          .ASCII  \ volumes:      !6UL\
                              4C 55 36 21 00460
20 55 50 43 20 64 65 67 72 61 68 43 20 20 2D 00464 P.ACI:  .ASCII  \-  Charged CPU time:  !%D   Elapsed time\
6C 45 20 20 20 44 25 21 20 20 3A 65 6D 69 74 00473
               65 6D 69 74 20 64 65 73 70 61 00482
                        44 25 21 20 20 3A 0048C          .ASCII  \:  !%D\

                                                         .PSECT  $OWN$,NOEXE,2

                                                001D4 BUFIO:   .BLKB   4
                                                001D8 WSPEAK:  .BLKB   4
                                                001DC DIRIO:   .BLKB   4
                                                001E0 VIRTPEAK:
                                                               .BLKB   4
                                                001E4 PAGEFLTS:
                                                               .BLKB   4
                                                001E8 VOLUMES: .BLKB   4

                                                         .EXTRN  LIB$SUBX, SYS$GETJPI

                                                         .PSECT  $CODE$,NOWRT,2

                                    003C 00000           .ENTRY  LOGOUT_MESSAGE, Save R2,R3,R4,R5        ; 2420
      55 00000000G   00   9E 00002          MOVAB   SYS$GETJPI, R5                                  ;
      54      0000'  CF   9E 00009          MOVAB   P.ACB, R4                                       ;
      53      0000'  CF   9E 0000E          MOVAB   BUFIO, R3                                       ;
      52 00000000G   00   9E 00013          MOVAB   WRITE_FAO, R2                                   ;
      5E      FF3C   CE   9E 0001A          MOVAB   -196(SP), SP                                    ;
E4    AD       C8   AD   9E 0001F          MOVAB   USERNAME_BUFFER, USERNAME+4                     ; 2460
DC    AD       B8   AD   9E 00024          MOVAB   PROCNAME_BUFFER, PROCNAME+4                     ; 2461
         02020010   8F   DD 00029          PUSHL   #33685520                                       ; 2463
04    AE       C8   AD   9E 0002F          MOVAB   USERNAME_BUFFER, JPI_LIST+4                     ; 2465
08    AE       E0   AD   9E 00034          MOVAB   USERNAME, JPI_LIST+8                            ; 2466
0C    AE 031C0010   8F   D0 00039          MOVL    #52166672, JPI_LIST+12                          ; 2467
10    AE       B8   AD   9E 00041          MOVAB   PROCNAME_BUFFER, JPI_LIST+16                    ; 2469
```

```
                14  AE      D8  AD  9E 00046          MOVAB   PROCNAME, JPI_LIST+20                  ; 2470
                    18      AE  D4  0004B            CLRL    JPI_LIST+24                           ; 2471
                            7E  7C  0004E            CLRQ    -(SP)                                 ; 2473
                            7E  D4  00050            CLRL    -(SP)
                        0C  AE  9F  00052            PUSHAB  JPI_LIST
                            7E  7C  00055            CLRQ    -(SP)
                            7E  D4  00057            CLRL    -(SP)
                    65          07  FB  00059        CALLS   #7, SYS$GETJPI
    09 00000000G    00          01  E1  0005C        BBC     #1, PPD+2, 1$                         ; 2475
                            7E  D4  00064            CLRL    -(SP)                                 ; 2477
                        E0  AD  9F  00066            PUSHAB  USERNAME
                            54  DD  00069            PUSHL   R4
                            17  11  0006B            BRB     3$
                    0A  0000'   CF  E8  0006D 1$:    BLBS    SUBPROCESS, 2$                        ; 2480
                            7E  D4  00072            CLRL    -(SP)                                 ; 2482
                        E0  AD  9F  00074            PUSHAB  USERNAME
                        1C  A4  9F  00077            PUSHAB  P.ACC
                            08  11  0007A            BRB     3$
                            7E  D4  0007C 2$:        CLRL    -(SP)                                 ; 2486
                        D8  AD  9F  0007E            PUSHAB  PROCNAME
                        34  A4  9F  00081            PUSHAB  P.ACD
                    62          03  FB  00084 3$:    CALLS   #3, WRITE_FAO
                    01  FE2D   C3  E8  00087        BLBS    FULL_LOGOUT, 4$                       ; 2490
                                04  0008C            RET
                    50          6E  9E  0008D 4$:    MOVAB   JPI_LIST, PTR                         ; 2503
                    80 040C0004 8F  D0  00090        MOVL    #67895300, (PTR)+
                    80          63  9E  00097        MOVAB   BUFIO, (PTR)+
                                80  D4  0009A        CLRL    (PTR)+
                    80 02010004 8F  D0  0009C        MOVL    #33619972, (PTR)+
                    80      04  A3  9E  000A3        MOVAB   WSPEAK, (PTR)+
                                80  D4  000A7        CLRL    (PTR)+
                    80 040B0004 8F  D0  000A9        MOVL    #67829764, (PTR)+
                    80      08  A3  9E  000B0        MOVAB   DIRIO, (PTR)+
                                80  D4  000B4        CLRL    (PTR)+
                    80 02000004 8F  D0  000B6        MOVL    #33554436, (PTR)+
                    80      0C  A3  9E  000BD        MOVAB   VIRTPEAK, (PTR)+
                                80  D4  000C1        CLRL    (PTR)+
                    80 040A0004 8F  D0  000C3        MOVL    #67764228, (PTR)+
                    80      10  A3  9E  000CA        MOVAB   PAGEFLTS, (PTR)+
                                80  D4  000CE        CLRL    (PTR)+
                    80 02050004 8F  D0  000D0        MOVL    #33882116, (PTR)+
                    80      14  A3  9E  000D7        MOVAB   VOLUMES, (PTR)+
                                80  D4  000DB        CLRL    (PTR)+
                    80 04070008 8F  D0  000DD        MOVL    #67567624, (PTR)+
                    80      F8  AD  9E  000E4        MOVAB   CPUTIM, (PTR)+
                                80  D4  000E8        CLRL    (PTR)+
                    80 02060008 8F  D0  000EA        MOVL    #33947656, (PTR)+
                    80      F0  AD  9E  000F1        MOVAB   LOGINTIM, (PTR)+
                                80  D4  000F5        CLRL    (PTR)+
                                80  7C  000F7        CLRQ    (PTR)+
                                80  D4  000F9        CLRL    (PTR)+
                                7E  7C  000FB        CLRQ    -(SP)                                 ; 2505
                                7E  D4  000FD        CLRL    -(SP)
                            0C  AE  9F  000FF        PUSHAB  JPI_LIST
                                7E  7C  00102        CLRQ    -(SP)
                                7E  D4  00104        CLRL    -(SP)
                    65          07  FB  00106        CALLS   #7, SYS$GETJPI
```

```
      F8   AD            00       F8   AD FFFE7960   8F   7A 00109          EMUL    #-100000, CPUTIM, #0, CPUTIM                    ; 2507
                                          E8   AD   9F 00114          PUSHAB  CURTIME                                         ; 2510
                     00000000G  00             01   FB 00117          CALLS   #1, SYS$GETTIM
                                          F0   AD   9F 0011E          PUSHAB  LOGINTIM                                        ; 2512
                                          E8   AD   9F 00121          PUSHAB  CURTIME
                                          F0   AD   9F 00124          PUSHAB  LOGINTIM
                     00000000G  00             03   FB 00127          CALLS   #3, LIB$SUBX
                                          54   A4   9F 0012E          PUSHAB  P.ACE                                           ; 2514
                                62             01   FB 00131          CALLS   #1, WRITE_FAO
                                7E             63   7D 00134          MOVQ    BUFIO, -(SP)                                    ; 2517
                                          70   A4   9F 00137          PUSHAB  P.ACF                                           ; 2515
                                62             03   FB 0013A          CALLS   #3, WRITE_FAO
                                7E       08    A3   7D 0013D          MOVQ    DIRIO, -(SP)                                    ; 2520
                                       00AB    C4   9F 00141          PUSHAB  P.ACG                                           ; 2518
                                62             03   FB 00145          CALLS   #3, WRITE_FAO
                                7E       10    A3   7D 00148          MOVQ    PAGEFLTS, -(SP)                                 ; 2523
                                       00E6    C4   9F 0014C          PUSHAB  P.ACH                                           ; 2521
                                62             03   FB 00150          CALLS   #3, WRITE_FAO
                                          F0   AD   9F 00153          PUSHAB  LOGINTIM                                        ; 2524
                                          F8   AD   9F 00156          PUSHAB  CPUTIM
                                       0121    C4   9F 00159          PUSHAB  P.ACI
                                62             03   FB 0015D          CALLS   #3, WRITE_FAO
                                               04   00160          RET                                                     ; 2528

; Routine Size:  353 bytes,    Routine Base:  $CODE$ + 0D68
```

```
;  2121          2529  1 ROUTINE rundown_cli =
;  2122          2530  1
;  2123          2531  1 !---
;  2124          2532  1 !
;  2125          2533  1 !       This routine executes in supervisor mode to cancel the CLI's exit
;  2126          2534  1 !       handlers. This is necessary to prevent external attacks on a LOGOUT in
;  2127          2535  1 !       progress with the $FORCEX system service. With the exit handlers
;  2128          2536  1 !       canceled, if LOGOUT is forcibly exited, the process goes away.
;  2129          2537  1 !       In addition, a $CANCEL is done on the PPD$W_INPCHAN channel.
;  2130          2538  1 !
;  2131          2539  1 ! Calling Sequence:
;  2132          2540  1 !
;  2133          2541  1 !       This procedure is called in SUPERVISOR mode with the LGI$CMSUPR
;  2134          2542  1 !       procedure.
;  2135          2543  1 !
;  2136          2544  1 ! Inputs:
;  2137          2545  1 !
;  2138          2546  1 !       None
;  2139          2547  1 !
;  2140          2548  1 ! Outputs:
;  2141          2549  1 !
;  2142          2550  1 !       Status of the rundown.
;  2143          2551  1 !
;  2144          2552  1 ! Side Effects:
;  2145          2553  1 !
;  2146          2554  1 !       Supervisor mode exit handlers canceled.
;  2147          2555  1 !       Channel PPD$W_INPCHAN is canceled.
;  2148          2556  1 !
;  2149          2557  1 !---
;  2150          2558  1
;  2151          2559  2 BEGIN
;  2152          2560  2
;  2153          2561  2 LOCAL
;  2154          2562  2     status;
;  2155          2563  2
;  2156          2564  3 IF (status = $CANEXH())                              ! Cancel all exit handlers
;  2157          2565  2 AND .terminal_device
;  2158          2566  2 THEN
;  2159          2567  2     status = $CANCEL(CHAN = .ppd [ppd$w_inpchan]); ! Cancel AST request(s)
;  2160          2568  2 RETURN .status;
;  2161          2569  2
;  2162          2570  1 END;


                                                        .EXTRN  SYS$CANEXH, SYS$CANCEL

                                        0000 00000 RUNDOWN_CLI:
                                                        .WORD   Save nothing                    ;  2529
                                        7E  D4 00002            CLRL    -(SP)                   ;  2564
                00000000G   00          01  FB 00004            CALLS   #1, SYS$CANEXH          ;
                            13          50  E9 0000B            BLBC    STATUS, 1$              ;
                            0E      0000' CF  E9 0000E            BLBC    TERMINAL_DEVICE, 1$     ;  2565
                            7E 00000000G  00  3C 00013            MOVZWL  PPD+30, -(SP)           ;  2567
                00000000G   00          01  FB 0001A            CALLS   #1, SYS$CANCEL          ;
                                            04 00021 1$:        RET                             ;  2570
```

```
; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 0EC9
```

```
;  2164          2571  1 GLOBAL ROUTINE handler (signal_args, mechanism_args) =
;  2165          2572  1
;  2166          2573  1 !---
;  2167          2574  1 !
;  2168          2575  1 !       This is the primary condition handler for the loginout
;  2169          2576  1 !       image.  Issue the error message, and for fatal errors,
;  2170          2577  1 !       force the termination of the process.
;  2171          2578  1 !
;  2172          2579  1 ! Inputs:
;  2173          2580  1 !
;  2174          2581  1 !       Access mode may be either user or executive depending on
;  2175          2582  1 !       the current mode when the condition is signaled.
;  2176          2583  1 !
;  2177          2584  1 !       signal_args = Address of signal argument vector
;  2178          2585  1 !       mechanism_args = Address of mechanism argument vector
;  2179          2586  1 !
;  2180          2587  1 ! Outputs:
;  2181          2588  1 !
;  2182          2589  1 !       None
;  2183          2590  1 !---
;  2184          2591  1
;  2185          2592  2 BEGIN
;  2186          2593  2
;  2187          2594  2 MAP
;  2188          2595  2     signal_args:        REF BBLOCK,     ! Address of signal vector
;  2189          2596  2     mechanism_args:     REF BBLOCK;     ! Address of mechanism vector
;  2190          2597  2
;  2191          2598  2 BIND
;  2192          2599  2     status = signal_args [chf$l_sig_name]: BBLOCK; ! Get at status fields
;  2193          2600  2
;  2194          2601  2 IF .signal_args [chf$l_sig_name] EQL ss$_unwind ! If unwinding,
;  2195          2602  2 THEN
;  2196          2603  2     RETURN false;                          ! then pass on to next handler
;  2197          2604  2
;  2198          2605  2 signal_args [chf$l_sig_args] = .signal_args [chf$l_sig_args] - 2;
;  2199          2606  2 signal_args [2,0,16,0] = 1;                ! Output only text portion
;  2200          2607  2                                           ! (inhibit message prefixes)
;  2201          2608  2
;  2202          2609  2 write_output_status = 0;                  ! Preset no message output
;  2203        P 2610  2 $PUTMSG(MSGVEC = .signal_args,            ! Output messages
;  2204          2611  2         ACTRTN = write_output);           ! using write_output to put messages
;  2205          2612  2
;  2206          2613  2 IF .status NEQ lgi$_notvalid             ! If user validation error,
;  2207          2614  2     AND .status NEQ lgi$_userauth        ! return with validation code
;  2208          2615  2 THEN                                     ! already set by signaller
;  2209          2616  2     ppd [ppd$l_lststatus] = .status;     ! Else, set final job status
;  2210          2617  2
;  2211          2618  2 IF .status EQL lgi$_inputerr            ! If error opening input file,
;  2212          2619  2 THEN
;  2213          2620  2     ppd [ppd$l_lststatus] = .(.signal_args+12); ! Exit with RMS status code
;  2214          2621  2                                           ! for NETACP interpretation of netjob
;  2215          2622  2
;  2216          2623  2 IF .doing_login                          ! If this is a login
;  2217          2624  2 AND .status [sts$v_severity] NEQ sts$k_warning !  and not only a warning
;  2218          2625  2 THEN security_audit(nsa$k_rectyp_logf,  !   then security audit the failure
;  2219          2626  2                     .ppd [ppd$l_lststatus]);
;  2220          2627  2
```

```
;  2221          2628  2 IF .status [sts$v_severity] EQL sts$k_severe     . If fatal error,
;  2222          2629  2 THEN
;  2223          2630  3     BEGIN
;  2224          2631  3     IF .ctl$gl_creprc_flags[prc$v_batch]         ! then, if batch job
;  2225          2632  3     THEN terminate_batch(.signal_args)           ! terminate job stream
;  2226          2633  3     ELSE
;  2227          2634  4         BEGIN
;  2228          2635  4         IF .doing_login                          ! If this is a login,
;  2229          2636  4         THEN set_terminal_hangup(true);          ! Ensure terminal hangs up
;  2230          2637  4         IF .write_output_status                  ! If exit status output,
;  2231          2638  4         THEN ppd [ppd$l_[ststatus] =             ! Inhibit further output
;  2232          2639  4             .ppd [ppd$l_[ststatus] OR sts$m_inhib_msg;
;  2233          2640  4         $CMEXEC(ROUTIN = exit_process);          ! Terminate process
;  2234          2641  3         END;
;  2235          2642  2     END;
;  2236          2643  2
;  2237          2644  2 RETURN ss$_continue;                  ! otherwise, continue execution
;  2238          2645  2
;  2239          2646  1 END;


                                                        .EXTRN  SYS$PUTMSG

                                                        .ENTRY  HANDLER, Save R2,R3,R4                          ; 2571
                    54 00000000G   00   9E 00002        MOVAB   WRITE_OUTPUT_STATUS, R4                         ;
                    53 00000000G   00   9E 00009        MOVAB   PPD+24, R3                                      ;
                    52       04    AC   D0 00010        MOVL    SIGNAL_ARGS, R2                                 ; 2599
         00000920   8F       04    A2   D1 00014        CMPL    4(R2), #2336                                    ; 2601
                                03   12 0001C            BNEQ    1$                                              ;
                             0095   31 0001E            BRW     9$                                              ;
                    62           02   C2 00021 1$:      SUBL2   #2, (R2)                                        ; 2605
                02  A2           01   B0 00024        MOVW    #1, 2(R2)                                       ; 2606
                                64   D4 00028            CLRL    WRITE_OUTPUT_STATUS                             ; 2609
                                7E   7C 0002A            CLRQ    -(SP)                                           ; 2611
                         00000000G   00   9F 0002C        PUSHAB  WRITE_OUTPUT                                    ;
                                52   DD 00032            PUSHL   R2                                              ;
                00000000G       00       04   FB 00034        CALLS   #4, SYS$PUTMSG                          ;
                00000000G       8F       04   A2   D1 0003B        CMPL    4(R2), #LGI$_NOTVALID                   ; 2613
                                                0E   13 00043        BEQL    2$                                      ;
                00000000G       8F       04   A2   D1 00045        CMPL    4(R2), #LGI$_USERAUTH                   ; 2614
                                                04   13 0004D        BEQL    2$                                      ;
                                63       04   A2   D0 0004F        MOVL    4(R2), PPD+24                           ; 2616
                00000000G       8F       04   A2   D1 00053 2$:     CMPL    4(R2), #LGI$_INPUTERR                   ; 2618
                                                04   12 0005B        BNEQ    3$                                      ;
                                63       0C   A2   D0 0005D        MOVL    12(R2), PPD+24                          ; 2620
                                11   0000'   CF   E9 00061 3$:      BLBC    DOING_LOGIN, 4$                         ; 2623
                                07       04   A2   93 00066        BITB    4(R2), #7                               ; 2624
                                                0B   13 0006A        BEQL    4$                                      ;
                                                63   DD 0006C        PUSHL   PPD+24                                  ; 2626
                                                06   DD 0006E        PUSHL   #6                                      ; 2625
                00000000G       00               02   FB 00070        CALLS   #2, SECURITY_AUDIT                      ;
04       04  A2                   03               00   ED 00077 4$:     CMPZV   #0, #3, 4(R2), #4                   ; 2628
                                                33   12 0007D        BNEQ    8$                                      ;
             0B 0000C000G       00               04   E1 0007F        BBC     #4, CTL$GL_CREPRC_FLAGS, 5$             ; 2631
                                                52   DD 00087        PUSHL   R2                                      ; 2632
                00000000G       00               01   FB 00089        CALLS   #1, TERMINATE_BATCH                     ;
```

```
                                 20  11 00090          BRB     8$                                                      ;
                         07      CF  E9 00092 5$:      BLBC    DOING_LOGIN, 6$                                         ; 2635
                                 01  DD 00097          PUSHL   #1                                                      ; 2636
                    0000V        CF  01 FB 00099       CALLS   #1, SET_TERMINAL_HANGUP                                 ;
                         04      64  E9 0009E 6$:      BLBC    WRITE_OUTPUT_STATUS, 7$                                 ; 2637
                      03 A3      10  88 000A1          BISB2   #16, PPD+27                                             ; 2639
                                 7E  D4 000A5 7$:      CLRL    -(SP)                                                   ; 2640
                              0000V  CF 9F 000A7       PUSHAB  EXIT_PROCESS                                            ;
                00000000G        00  02 FB 000AB       CALLS   #2, SYS$CMEXEC                                          ;
                         50      01  D0 000B2 8$:      MOVL    #1, R0                                                  ; 2644
                                     04 000B5          RET                                                             ;
                                 50  D4 000B6 9$:      CLRL    R0                                                      ; 2646
                                     04 000B8          RET                                                             ;

; Routine Size:  185 bytes,    Routine Base:  $CODE$ + 0EEB
```

```
;  2241          2647  1 GLOBAL ROUTINE exit_process: NOVALUE =
;  2242          2648  1
;  2243          2649  1 !---
;  2244          2650  1 !
;  2245          2651  1 !       Terminate the process execution.  This is done in executive
;  2246          2652  1 !       mode in order to bypass any supervisor mode exit handlers.
;  2247          2653  1 !
;  2248          2654  1 ! Inputs:
;  2249          2655  1 !
;  2250          2656  1 !       Access mode is executive.
;  2251          2657  1 !
;  2252          2658  1 !       None
;  2253          2659  1 !
;  2254          2660  1 ! Outputs:
;  2255          2661  1 !
;  2256          2662  1 !       There is no return - the image is exited.
;  2257          2663  1 !---
;  2258          2664  1
;  2259          2665  2 BEGIN
;  2260          2666  2
;  2261          2667  2 $EXIT(CODE = .ppd [ppd$l_iststatus]);   ! Exit with final job status
;  2262          2668  2
;  2263          2669  1 END;


                                                        .EXTRN  SYS$EXIT

                                        0000 00000      .ENTRY  EXIT_PROCESS, Save nothing                      ; 2647
                              00000000G 00  DD 00002    PUSHL   PPD+24                                          ; 2667
                    00000000G   00        01  FB 00008 CALLS   #1, SYS$EXIT                                    ;
                                          04  0000F     RET                                                     ; 2669

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 0FA4
```

```
;  2265          2670  1 ROUTINE clear_creator_cli: NOVALUE =
;  2266          2671  1
;  2267          2672  1 !---
;  2268          2673  1 !
;  2269          2674  1 !       Clear the creator's CLI and CLI table name strings.
;  2270          2675  1 !       Also clear inherited mandatory auditing status.
;  2271          2676  1 !
;  2272          2677  1 ! Inputs:
;  2273          2678  1 !
;  2274          2679  1 !       Access mode is kernel.
;  2275          2680  1 !
;  2276          2681  1 ! Outputs:
;  2277          2682  1 !
;  2278          2683  1 !       None.
;  2279          2684  1 !
;  2280          2685  1 !---
;  2281          2686  1
;  2282          2687  2 BEGIN
;  2283          2688  2
;  2284          2689  2 EXTERNAL
;  2285          2690  2     sch$gl_curpcb:      REF BBLOCK,     ! Current PCB pointer
;  2286          2691  2     ctl$gt_cliname:     VECTOR[,BYTE],  ! Creator's CLI name (ASCIC)
;  2287          2692  2     ctl$gt_tablename:   VECTOR[,BYTE];  ! Creator's CLI table name (ASCIC)
;  2288          2693  2
;  2289          2694  2 sch$gl_curpcb [pcb$v_secaudit] = false;
;  2290          2695  2 pcb_sts [$BITPOSITION(pcb$v_secaudit)] = false;
;  2291          2696  2 ctl$gt_cliname [0] = 0;
;  2292          2697  2 ctl$gt_tablename [0] = 0;
;  2293          2698  2
;  2294          2699  1 END;


                                                        .EXTRN  SCH$GL_CURPCB, CTL$GT_CLINAME
                                                        .EXTRN  CTL$GT_TABLENAME

                                        0000 00000 CLEAR_CREATOR_CLI:
                                                        .WORD   Save nothing                    ; 2670
                     50 00000000G   00  D0 00002        MOVL    SCH$GL_CURPCB, R0               ; 2694
                  27 A0             08  8A 00009        BICB2   #8, 39(R0)                      ;
            0000' CF                08  8A 0000D        BICB2   #8, PCB_STS+3                   ; 2695
                        00000000G   00  94 00012        CLRB    CTL$GT_CLINAME                  ; 2696
                        00000000G   00  94 00018        CLRB    CTL$GT_TABLENAME                ; 2697
                                    04  0001E        RET                                     ; 2699

; Routine Size:  31 bytes,    Routine Base:  $CODE$ + 0FB4
```

```
;  2296          2700  1 GLOBAL ROUTINE set_terminal_hangup (hangup): NOVALUE =
;  2297          2701  1
;  2298          2702  1 !---
;  2299          2703  1 !
;  2300          2704  1 !       Set or clear terminal's hangup state.
;  2301          2705  1 !
;  2302          2706  1 ! Inputs:
;  2303          2707  1 !
;  2304          2708  1 !       hangup = Hangup state to set.
;  2305          2709  1 !
;  2306          2710  1 ! Outputs:
;  2307          2711  1 !
;  2308          2712  1 !       None.
;  2309          2713  1 !
;  2310          2714  1 !---
;  2311          2715  1
;  2312          2716  2 BEGIN
;  2313          2717  2
;  2314          2718  2 OWN
;  2315          2719  2     hang_devchar2: BLOCK [4,BYTE];      ! Holds terminal's disconnected state
;  2316          2720  2
;  2317          2721  2 BIND
;  2318          2722  2     hang_item = UPLIT (4 OR (dvi$_devchar2 ^ 16), hang_devchar2, 0, 0);
;  2319          2723  2
;  2320          2724  2 LOCAL
;  2321          2725  2     channel:    WORD,                   ! Place to store assigned channel
;  2322          2726  2     iosb:       VECTOR [4,WORD],        ! I/O status block
;  2323          2727  2     buffer:     VECTOR [3];             ! Characteristics buffer
;  2324          2728  2
;  2325          2729  2 IF NOT .terminal_device                 ! If not a real terminal,
;  2326          2730  2 OR .dev_char_2 [dev$v_rtt]             !  or terminal is a remote terminal,
;  2327          2731  2 OR .ppd [ppd$v_mode]                   !  or non-interactive,
;  2328          2732  2 OR .subprocess                         !  or a subprocess,
;  2329        P 2733  2 OR NOT $GETDVIW(DEVNAM = term_name,     !  or can't fetch terminal's state,
;  2330          2734  3                   ITMLST = hang_item)
;  2331          2735  2 OR .hang_devchar2 [dev$v_det]          !  or terminal now disconnected,
;  2332          2736  2 THEN RETURN;                           !   then forget any hangup
;  2333          2737  2
;  2334        P 2738  3 IF ($ASSIGN(DEVNAM = term_name,         ! Assign a channel to the terminal
;  2335          2739  3             CHAN   = channel))
;  2336          2740  2 THEN
;  2337          2741  3     BEGIN
;  2338        P 2742  4     IF ($QIOW(CHAN = .channel,          ! Get current characteristics
;  2339        P 2743  4               FUNC = io$_sensemode,
;  2340        P 2744  4               IOSB = iosb,
;  2341        P 2745  4               P1   = buffer,
;  2342          2746  4               P2   = 12))
;  2343          2747  3     AND .iosb[0]
;  2344          2748  3     THEN
;  2345          2749  4         BEGIN
;  2346          2750  5         (
;  2347          2751  5          BIND devdepend2 = buffer [2]: BBLOCK; ! Where HANGUP bit is stored
;  2348          2752  5          devdepend2 [tt2$v_hangup] = .hangup; ! Set the correct hangup state
;  2349          2753  4         );
;  2350        P 2754  4         $QIOW(CHAN = .channel,          ! Write characteristics back
;  2351        P 2755  4               FUNC = io$_setmode,
;  2352        P 2756  4               P1   = buffer,
```

```
;  2353        2757  4                 P2   = 12);
;  2354        2758  3             END;
;  2355        2759  3         $DASSGN(CHAN = .channel);          ! Free up the channel
;  2356        2760  2         END;
;  2357        2761  2
;  2358        2762  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                        00492           .BLKB   2
                                00E60004 00494 P.ACJ:   .LONG   15073284
                                00000000' 00498         .ADDRESS HANG_DEVCHAR2                          ;
                          00000000 00000000 0049C       .LONG   0, 0                                    ;

                                                .PSECT  $OWN$,NOEXE,2

                                        001EC HANG_DEVCHAR2:
                                                .BLKB   4

                                     HANG_ITEM=         P.ACJ
                                                .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                .EXTRN  SYS$DASSGN

                                                .PSECT  $CODE$,NOWRT,2

                                          0004 00000    .ENTRY  SET_TERMINAL_HANGUP, Save R2            ; 2700
                    52 00000000G   00   9E 00002        MOVAB   SYS$QIOW, R2                            ;
                    5E             18   C2 00009        SUBL2   #24, SP                                 ;
                    01      0000'  CF   E8 0000C        BLBS    TERMINAL_DEVICE, 1$                     ; 2729
                                   04      00011        RET                                             ;
                01      0000'  CF          02   E1 00012 1$:     BBC     #2, DEV_CHAR_2, 2$              ; 2730
                                   04      00018        RET                                             ;
                7E 00000000G   00          01   E0 00019 2$:     BBS     #1, PPD+2, 4$                   ; 2731
                               79      0000' CF   E8 00021        BLBS    SUBPROCESS, 4$                  ; 2732
                                      7E   7C 00026        CLRQ    -(SP)                                   ; 2734
                                      7E   7C 00028        CLRQ    -(SP)                                   ;
                                    0000'  CF   9F 0002A        PUSHAB  HANG_ITEM                       ;
                                    0000'  CF   9F 0002E        PUSHAB  TERM_NAME                       ;
                                      7E   7C 00032        CLRQ    -(SP)                               ;
                   00000000G   00          08   FB 00034        CALLS   #8, SYS$GETDVIW                 ;
                               61          50   E9 0003B        BLBC    R0, 4$                          ;
                5B      0000'  CF          01   E0 0003E        BBS     #1, HANG_DEVCHAR2, 4$           ; 2735
                                      7E   7C 00044        CLRQ    -(SP)                               ; 2739
                                      08   AE   9F 00046        PUSHAB  CHANNEL                         ;
                                    0000'  CF   9F 00049        PUSHAB  TERM_NAME                       ;
                   00000000G   00          04   FB 0004D        CALLS   #4, SYS$ASSIGN                  ;
                               48          50   E9 00054        BLBC    R0, 4$                          ;
                                      7E   7C 00057        CLRQ    -(SP)                               ; 2746
                                      7E   7C 00059        CLRQ    -(SP)                               ;
                                      0C   DD 0005B        PUSHL   #12                             ;
                                      18   AE   9F 0005D        PUSHAB  BUFFER                          ;
                                      7E   7C 00060        CLRQ    -(SP)                               ;
                                      30   AE   9F 00062        PUSHAB  IOSB                            ;
                                      27   DD 00065        PUSHL   #39                             ;
                               7E     28   AE   3C 00067        MOVZWL  CHANNEL, -(SP)                  ;
```

```
                                        7E  D4 0006B          CLRL    -(SP)                                                   ;
                                    62  0C  FB 0006D          CALLS   #12, SYS$QIOW                                           ;
                                    22  50  E9 00070          BLBC    R0, 3$                                                  ;
                                    1E      AE E9 00073          BLBC    IOSB, 3$                                                ; 2747
      0C    AE              01        02  04  AC F0 00077          INSV    HANGUP, #2, #1, DEVDEPEND2                              ; 2752
                                        7E  7C C007E          CLRQ    -(SP)                                                   ; 2757
                                        7E  7C 00080          CLRQ    -(SP)                                                   ;
                                        0C  DD 00082          PUSHL   #12                                                     ;
                                    18      AE 9F 00084          PUSHAB  BUFFER                                                  ;
                                        7E  7C 00087          CLRQ    -(SP)                                                   ;
                                    7E  23  7D 00089          MOVQ    #35, -(SP)                                              ;
                                    7E  28  AE 3C 0008C          MOVZWL  CHANNEL, -(SP)                                          ;
                                        7E  D4 00090          CLRL    -(SP)                                                   ;
                                    62  0C  FB 00092          CALLS   #12, SYS$QIOW                                           ;
                                    7E  6E  3C 00095 3$:      MOVZWL  CHANNEL, -(SP)                                          ; 2759
                  00000000G         00  01  FB 00098          CALLS   #1, SYS$DASSGN                                          ;
                                            04 0009F 4$:      RET                                                             ; 2762

; Routine Size:  160 bytes,    Routine Base:  $CODE$ + 0FD3
```

```
; 2360          2763  1 END
; 2361          2764  0 ELUDOM


                                                        .EXTRN  LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  $GLOBAL$                      292  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                         496  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                       1188  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       4211  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       157         0        1000        00:01.4



;                                       COMMAND QUALIFIERS

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

; Size:         4205 code + 1982 data bytes
; Run Time:        00:54.9
; Elapsed Time:    03:50.2
; Lines/CPU Min:    3019
; Lexemes/CPU-Min: 35826
; Memory Used:  354 pages
; Compilation Complete
```

LIBHOUR
LIS

LIBDAYWK
LIS

INITUSER
LIS

LOGIN
LIS

INTERACT
LIS

DEFINE
MAR

LOGINCMD
LIS

ACTONE
LIS

MESSAGES
LIS

ACTCHR
LIS

VALIDATE
LIS

ACTIF
LIS

PROTCLI
LIS

CRFMACRO5
MAR

MACRO

MACRO32
MAP