I 3

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

```
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    WW      WW  DDDDDDDD
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    WW      WW  DDDDDDDD
SS          EE              TT      PP      PP  WW      WW  DD      DD
SS          EE              TT      PP      PP  WW      WW  DD      DD
SS          EE              TT      PP      PP  WW      WW  DD      DD
SS          EE              TT      PP      PP  WW      WW  DD      DD
  SSSSSS    EEEEEEEE        TT      PPPPPPPP    WW      WW  DD      DD
  SSSSSS    EEEEEEEE        TT      PPPPPPPP    WW      WW  DD      DD
        SS  EE              TT      PP          WW  WW  WW  DD      DD
        SS  EE              TT      PP          WW  WW  WW  DD      DD
        SS  EE              TT      PP          WWWW  WWWW  DD      DD      ....
        SS  EE              TT      PP          WWWW  WWWW  DD      DD      ....
SSSSSSSS    EEEEEEEEEE      TT      PP          WW      WW  DDDDDDDD        ....
SSSSSSSS    EEEEEEEEEE      TT      PP          WW      WW  DDDDDDDD        ....


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 SETPASSWORD(
;     2          0002  0                 IDENT = 'V04-000',
;     3          0003  0                 ADDRESSING_MODE(EXTERNAL = GENERAL)
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     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 !++
;    32          0032  1 ! FACILITY:     Set Password
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:     Changes a users password in SYSUAF.DAT
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:  DCL Set Utility
;    37          0037  1 !
;    38          0038  1 ! AUTHOR:       Chris Hume, CREATION DATE: 14-Jun-1979
;    39          0039  1 !
;    40          0040  1 ! MODIFIED BY:
;    41          0041  1 !
;    42          0042  1 !       V03-009 JRL0017         John R. Lawson, Jr.     6-Jul-1984 10:18
;    43          0043  1 !               Place system password in SYSUAF.DAT in a special record
;    44          0044  1 !               unaccessible by AUTHORIZE.
;    45          0045  1 !
;    46          0046  1 !       V03-008 BLS0291         Benn Schreiber          24-MAR-1984
;    47          0047  1 !               Define SET$_BADVALUE here so we don't pull
;    48          0048  1 !               SETFILE into SETP0.
;    49          0049  1 !
;    50          0050  1 !       V03-007 SHZ0003         Stephen H. Zalewski,    02-Mar-1984
;    51          0051  1 !               No longer make special checks for null passwords.  LGI$HPWD
;    52          0052  1 !               does now return a null password.
;    53          0053  1 !
;    54          0054  1 !       V03-006 SHZ0002         Stephen H. Zalewski
;    55          0055  1 !               Add support for /SECONDARY.  All misc other security
;    56          0056  1 !               enhancements.
;    57          0057  1 !
```

```
;    58          0058  1 !       V03-005 SHZ0001         Stephen H. Zalewski     30-Dec-1983
;    59          0059  1 !               Add support for /GENERATE.
;    60          0060  1 !
;    61          0061  1 !       V03-004 CWH3004         CW Hobbs                9-Nov-1983
;    62          0062  1 !               Remove the test for CAPTIVE added in GAS0080.  This broke
;    63          0063  1 !               some existing applications and is not strictly necessary
;    64          0064  1 !               since the LOCKPWD bit can be used.
;    65          0065  1 !
;    66          0066  1 !       V03-003 GAS0139         Gerry Smith             17-Jun-1983
;    67          0067  1 !               Add the system password.
;    68          0068  1 !
;    69          0069  1 !       V03-002 GAS0112         Gerry Smith             29-Mar-1983
;    70          0070  1 !               Remove all references to the old command dispatcher.
;    71          0071  1 !
;    72          0072  1 !       V03-001 GAS0080         Gerry Smith              4-May-1982
;    73          0073  1 !               Disallow changing of password if the CAPTIVE bit
;    74          0074  1 !               is set in the authorization record.
;    75          0075  1 !
;    76          0076  1 !--
;    77          0077  1
;    78          0078  1 LIBRARY
;    79          0079  1     'SYS$LIBRARY:LIB';
;    80          0080  1
;    81          0081  1 !
;    82          0082  1 ! Macro to make counted ASCII strings
;    83          0083  1 !
;    84          0084  1 MACRO
;    85        M 0085  1     CSTRING[] = (UPLIT BYTE(%CHARCOUNT(%STRING(%REMAINING)),
;    86          0086  1                 %STRING(%REMAINING)))%;
;    87          0087  1
```

```
;    89          0088  1 !
;    90          0089  1 ! Table of contents
;    91          0090  1 !
;    92          0091  1 FORWARD ROUTINE
;    93          0092  1     set$password : NOVALUE,             ! The module entry point
;    94          0093  1     get_pwd,                            ! Terminal dialogue
;    95          0094  1     get_record,                         ! User Input
;    96          0095  1     update_uaf : NOVALUE,               ! Update the UAF record
;    97          0096  1     update_sys : NOVALUE,               ! Update the system password
;    98          0097  1     open_uaf,                           ! $OPEN and $CONNECT to UAF
;    99          0098  1     get_uaf_record,                     ! Get a record from UAF.
;   100          0099  1     check_qualifiers;                   ! Get and check command qualifiers.
;   101          0100  1
;   102          0101  1
;   103          0102  1 !
;   104          0103  1 ! External routines
;   105          0104  1 !
;   106          0105  1 EXTERNAL ROUTINE
;   107          0106  1     str$upcase,
;   108          0107  1     set_password_generate,
;   109          0108  1     cli$present,
;   110          0109  1     cli$get_value,
;   111          0110  1     lib$cvt_dtb,
;   112          0111  1     lgi$hpwd;
;   113          0112  1
;   114          0113  1 !
;   115          0114  1 ! External storage
;   116          0115  1 !
;   117          0116  1 EXTERNAL
;   118          0117  1     ctl$gq_procpriv : $BBLOCK[8];       ! Process privilege mask
;   119          0118  1
;   120          0119  1 !
;   121          0120  1 ! Define messages
;   122          0121  1 !
;   123          0122  1 EXTERNAL LITERAL
;   124          0123  1     set$_pwdnotval,                     ! Error validating old password
;   125          0124  1     set$_pwdnotver,                     ! Error verifying new password
;   126          0125  1     set$_pwdlocked,                     ! Password is locked
;   127          0126  1     set$_pwdsyntax,                     ! Syntax error on new password
;   128          0127  1     set$_syspwderr,                     ! Error modifying system password
;   129          0128  1     set$_uaferr,                        ! Error accessing authorization file
;   130          0129  1     set$_invpwdlen,                     ! Invalid password length
;   131          0130  1     set$_pwdnotdif;                     ! New password must be different from curretn one
;   132          0131  1
;   133        P 0132  1 $SHR_MSGDEF(set,119,local,
;   134          0133  1                 (badvalue,error));
;   135          0134  1
;   136          0135  1 !
;   137          0136  1 ! Define TRUE and FALSE
;   138          0137  1 !
;   139          0138  1 LITERAL
;   140          0139  1     false = 0,
;   141          0140  1     true = 1;
;   142          0141  1
;   143          0142  1 LITERAL
;   144        P 0143  1     $EQULST (QUAL_,,0,1,
;   145        P 0144  1         (system,);              ! /SYSTEM
```

```
;  146        P 0145  1          (generate,),              ! /GENERATE
;  147          0146  1          (secondary,));            ! /SECONDARY
;  148          0147  1
;  149          0148  1 !
;  150          0149  1 ! OWN storage
;  151          0150  1 !
;  152          0151  1 OWN
;  153          0152  1     setpwd$flags : BITVECTOR[32] INITIAL(0),
;  154          0153  1     min_pwd_length : INITIAL(0),
;  155          0154  1     old_pwd_buf : VECTOR[32,BYTE],
;  156          0155  1     new_pwd_buf : VECTOR[32,BYTE],
;  157          0156  1     vfy_pwd_buf : VECTOR[32,BYTE],
;  158          0157  1     old_pwd_dsc: VECTOR[2] INITIAL(0, old_pwd_buf),
;  159          0158  1     new_pwd_dsc: VECTOR[2] INITIAL(0, new_pwd_buf),
;  160          0159  1     uafbuf : BLOCK[uaf$c_length,BYTE],
;  161          0160  1     user_desc : VECTOR[2] INITIAL(0, uafbuf[uaf$t_username]),
;  162        P 0161  1     uaffab : $FAB(
;  163        P 0162  1         FAC = (get,put,upd),          ! access types
;  164        P 0163  1         FNM = 'SYSUAF',               ! user authorization file name
;  165        P 0164  1         DNM = 'SYS$SYSTEM:.DAT',
;  166          0165  1         SHR = (get,put,del,upd)),
;  167        P 0166  1     uafrab : $RAB(
;  168        P 0167  1         RAC = key,
;  169        P 0168  1         ROP = (rlk,uif),
;  170        P 0169  1         KRF = 0,
;  171        P 0170  1         KBF = uafbuf[uaf$t_username],
;  172        P 0171  1         KSZ = uaf$s_username,
;  173        P 0172  1         UBF = uafbuf,
;  174        P 0173  1         USZ = uaf$c_length,
;  175          0174  1         FAB = uaffab);
;  176          0175  1
```

```
;  178          0176  1 GLOBAL ROUTINE set$password : NOVALUE =
;  179          0177  2 BEGIN
;  180          0178  2 !+++
;  181          0179  2 !
;  182          0180  2 ! This is the entry for the SET PASSWORD command.  It is called by the SET
;  183          0181  2 ! command dispatcher
;  184          0182  2 !
;  185          0183  2 !---
;  186          0184  2
;  187          0185  2 LOCAL
;  188          0186  2     status;
;  189          0187  2
;  190          0188  2 !
;  191          0189  2 ! If not setting SYSTEM password, then open and connect to SYSUAF.DAT,
;  192          0190  2 ! then get uaf record of user.
;  193          0191  2 !
;  194          0192  2 IF NOT open_uaf()
;  195          0193  2     THEN RETURN;
;  196          0194  2 IF NOT get_uaf_record()
;  197          0195  2     THEN RETURN;
;  198          0196  2 IF .uafbuf[uaf$v_lockpwd] AND NOT CLI$PRESENT(%ASCID'SYSTEM') ! Check to see if the user has the right
;  199          0197  2     THEN                                                     ! to change the password.
;  200          0198  3         BEGIN                                                ! If the password is locked he can't.
;  201          0199  3         SIGNAL_STOP(set$_pwdlocked);
;  202          0200  3         RETURN;
;  203          0201  2         END;
;  204          0202  2
;  205          0203  2 !
;  206          0204  2 ! Check for command qualifiers, and determine minimum password length
;  207          0205  2 ! if /GENERATE was specified.
;  208          0206  2 !
;  209          0207  2 check_qualifiers();
;  210          0208  2
;  211          0209  2 !
;  212          0210  2 ! Get the old and new passwords.  If an error, then simply exit.
;  213          0211  2 !
;  214          0212  3 IF NOT (status = get_pwd())
;  215          0213  2 THEN
;  216          0214  3     BEGIN
;  217          0215  3     SIGNAL(.status);
;  218          0216  3     RETURN;
;  219          0217  2     END;
;  220          0218  2
;  221          0219  2 !
;  222          0220  2 ! If /SYSTEM, then modify the system password.  Otherwise change the password
;  223          0221  2 ! for this user.
;  224          0222  2 !
;  225          0223  2 IF .setpwd$flags[qual_system]
;  226          0224  2 THEN update_sys();
;  227          0225  2
;  228          0226  2 update_uaf();
;  229          0227  2
;  230          0228  2 RETURN;
;  231          0229  1 END;

                                                                .TITLE  SETPASSWORD
```

```
                                                                        .IDENT  \V04-000\

                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                    46  41  55  53  59  53  00000 P.AAA:  .ASCII  \SYSUAF\
54  41  44  2E  3A  4D  45  54  53  59  53  24  53  59  53  00006 P.AAB:  .ASCII  \SYS$SYSTEM:.DAT\
                                                                00015       .BLKB   3
                                        00  00  4D  45  54  53  59  53  00018 P.AAD:  .ASCII  \SYSTEM\<0><0>
                                                  010E0006      00020 P.AAC:  .LONG   17694726
                                                  00000000'     00024         .ADDRESS P.AAD

                                                                        .PSECT  $OWN$,NOEXE,2

                                                  00000000      00000 SETPWD$FLAGS:
                                                                        .LONG   0
                                                  00000000      00004 MIN_PWD_LENGTH:
                                                                        .LONG   0
                                                                00008 OLD_PWD_BUF:
                                                                        .BLKB   32
                                                                00028 NEW_PWD_BUF:
                                                                        .BLKB   32
                                                                00048 VFY_PWD_BUF:
                                                                        .BLKB   32
                                                  00000000      00068 OLD_PWD_DSC:
                                                                        .LONG   0
                                                  00000000'     0006C         .ADDRESS OLD_PWD_BUF
                                                  00000000      00070 NEW_PWD_DSC:
                                                                        .LONG   0
                                                  00000000'     00074         .ADDRESS NEW_PWD_BUF
                                                                00078 UAFBUF:   .BLKB   1412
                                                  00000000      005FC USER_DESC:
                                                                        .LONG   0
                                                  00000000'     00600         .ADDRESS UAFBUF+4
                                                        03      00604 UAFFAB:   .BYTE   3
                                                        50      00605         .BYTE   80
                                                      0000      00606         .WORD   0
                                                  0000000C      00608         .LONG   0
                                                  00000000      0060C         .LONG   0
                                                  00000000      00610         .LONG   0
                                                  00000000      00614         .LONG   0
                                                      0000      00618         .WORD   0
                                                        0B      0061A         .BYTE   11
                                                        0F      0061B         .BYTE   15
                                                  00000000      0061C         .LONG   0
                                                        00      00620         .BYTE   0
                                                        00      00621         .BYTE   0
                                                        00      00622         .BYTE   0
                                                        02      00623         .BYTE   2
                                                  00000000      00624         .LONG   0
                                                  00000000      00628         .LONG   0
                                                  00000000      0062C         .LONG   0
                                                  00000000'     00630         .ADDRESS P.AAA
                                                  00000000'     00634         .ADDRESS P.AAB
                                                        06      00638         .BYTE   6
                                                        0F      00639         .BYTE   15
                                                      0000      0063A         .WORD   0
                                                  00000000      0063C         .LONG   0
```

```
                0000  00640          .WORD   0                                       ;
                  00  00642          .BYTE   0                                       ;
                  00  00643          .BYTE   0                                       ;
            00000000  00644          .LONG   0                                       ;
            00000000  00648          .LONG   0                                       ;
                0000  0064C          .WORD   0                                       ;
                  00  0064E          .BYTE   0                                       ;
                  00  0064F          .BYTE   0                                       ;
            00000000  00650          .LONG   0                                       ;
                  0'  00654 UAFRAB:  .BYTE   1                                       ;
                  44  00655          .BYTE   68                                      ;
                0000  00656          .WORD   0                                       ;
            00080010  00658          .LONG   524304                                  ;
            00000000  0065C          .LONG   0                                       ;
            00000000  00660          .LONG   0                                       ;
                0000# 00664          .WORD   0[3]                                    ;
                0000  0066A          .WORD   0                                       ;
            00000000  0066C          .LONG   0                                       ;
                0000  00670          .WORD   0                                       ;
                  01  00672          .BYTE   1                                       ;
                  00  00673          .BYTE   0                                       ;
                0584  00674          .WORD   1412                                    ;
                0000  00676          .WORD   0                                       ;
            00000000' 00678          .ADDRESS UAFBUF                                 ;
            00000000  0067C          .LONG   0                                       ;
            00000000  0068C          .LONG   0                                       ;
            00000000' 00684          .ADDRESS UAFBUF+4                               ;
                  20  00688          .BYTE   32                                      ;
                  00  00689          .BYTE   0                                       ;
                  00  0068A          .BYTE   0                                       ;
                  00  0068B          .BYTE   0                                       ;
            00000000  0068C          .LONG   0                                       ;
            00000000' 00690          .ADDRESS UAFFAB                                 ;
            00000000  00694          .LONG   0                                       ;

                                     .EXTRN  STR$UPCASE, SET_PASSWORD_GENERATE
                                     .EXTRN  CLI$PRESENT, CLI$GET_VALUE
                                     .EXTRN  LIB$CVT_DTB, LGI$HPWD
                                     .EXTRN  CTL$GQ_PROCPRIV
                                     .EXTRN  SET$_PWDNOTVAL, SET$_PWDNOTVER
                                     .EXTRN  SET$_PWDLOCKED, SET$_PWDSYNTAX
                                     .EXTRN  SET$_SYSPWDERR, SET$_UAFERR
                                     .EXTRN  SET$_INVPWDLEN, SET$_PWDNOTDIF

                                     .PSECT  $CODE$,NOWRT,2

                          0000 00000 .ENTRY  SET$PASSWORD, Save nothing              ; 0176
          0000V  CF         00  FB 00002 CALLS   #0, OPEN_UAF                      ; 0192
                 50         50  E9 00007 BLBC    R0, 4$                           ;
          0000V  CF         00  FB 0000A CALLS   #0, GET_UAF_RECORD                ; 0194
                 48         50  E9 0000F BLBC    R0, 4$                           ;
1C        0000'  CF         02  E1 00012 BBC     #2, UAFBUF+468, 1$               ; 0196
                       0000' CF  9F 00018 PUSHAB  P.AAC                            ;
    00000000G    00         01  FB 0001C CALLS   #1, CLI$PRESENT                   ;
                 0E         50  E8 00023 BLBS    R0, 1$                           ;
                       00000000G 8F  DD 00026 PUSHL   #SET$_PWDLOCKED              ; 0199
    00000000G    00         01  FB 0002C CALLS   #1, LIB$STOP                      ;
```

```
                                04 00033        RET                                    ; 0198
      0000V  CF         00  FB 00034 1$:     CALLS   #0, CHECK_QUALIFIERS              ; 0207
      0000V  CF         00  FB 00039         CALLS   #0, GET_PWD                       ; 0212
             0A         50  E8 0003E         BLBS    STATUS, 2$                        ;
                        50  DD 00041         PUSHL   STATUS                            ; 0215
  00000000G  00         01  FB 00043         CALLS   #1, LIB$SIGNAL                    ;
                            04 0004A         RET                                       ; 0214
             05  0000'  CF  E9 0004B 2$:     BLBC    SETPWD$FLAGS, 3$                  ; 0223
      0000V  CF         00  FB 00050         CALLS   #0, UPDATE_SYS                    ; 0224
      0000V  CF         00  FB 00055 3$:     CALLS   #0, UPDATE_UAF                    ; 0226
                            04 0005A 4$:     RET                                       ; 0229
```

; Routine Size: 91 bytes, Routine Base: $CODE$ + 0000

; 232 0230 1

```
;  234          0231  1 ROUTINE get_pwd =
;  235          0232  2 BEGIN
;  236          0233  2 !+++
;  237          0234  2 !
;  238          0235  2 ! Ask the user for the new and old passwords, and verify that the user can
;  239          0236  2 ! type the new password twice in a row.
;  240          0237  2 !
;  241          0238  2 ! Inputs:
;  242          0239  2 !       None.
;  243          0240  2 !
;  244          0241  2 ! Outputs:
;  245          0242  2 !       None.
;  246          0243  2 !
;  247          0244  2 !---
;  248          0245  2
;  249          0246  2 LOCAL
;  250          0247  2     status,
;  251        P 0248  2     fab : $FAB(FNM = 'SYS$INPUT',
;  252          0249  2                FAC = get),
;  253        P 0250  2     rab : $RAB(ROP = <pmt,cvt,rne>,
;  254        P 0251  2                USZ = %ALLOCATION(old_pwd_buf),
;  255          0252  2                FAB = fab);
;  256          0253  2
;  257          0254  2 !
;  258          0255  2 ! Open sys$input.
;  259          0256  2 !
;  260          0257  3 IF NOT $OPEN(FAB = fab)
;  261          0258  2 THEN RETURN set$_pwdnotval;
;  262          0259  2
;  263          0260  3 IF NOT $CONNECT(RAB = rab)
;  264          0261  2 THEN RETURN set$_pwdnotval;
;  265          0262  2
;  266          0263  2 !
;  267          0264  2 ! Get the old password.
;  268          0265  2 !
;  269          0266  2 rab[rab$l_ubf] = old_pwd_buf;
;  270          0267  2 IF NOT get_record(CSTRING('Old password: '), rab)
;  271          0268  2 THEN RETURN set$_pwdnotval
;  272          0269  2 ELSE old_pwd_dsc[0] = .rab[rab$w_rsz];
;  273          0270  2
;  274          0271  2 !
;  275          0272  2 ! Get the new password.  If user specified /GENERATE, then generate passwords
;  276          0273  2 ! for s/he to choose from. (Grammar?)
;  277          0274  2 !
;  278          0275  2 IF .setpwd$flags[qual_generate]
;  279          0276  2 THEN
;  280          0277  3     BEGIN
;  281          0278  3     BIND new_password = new_pwd_buf : VECTOR[,WORD];
;  282          0279  3     set_password_generate(new_pwd_buf,min_pwd_length);  ! Returns ASCIC string.
;  283          0280  3     new_pwd_dsc[0] = .new_password[0];          ! Move count into descriptor.
;  284          0281  3     new_pwd_dsc[1] = .new_pwd_dsc[1] + 2;       ! Move buffer address past count.
;  285          0282  3     str$upcase (new_pwd_dsc, new_pwd_dsc);      ! Upcase the password
;  286          0283  3     END
;  287          0284  2 ELSE
;  288          0285  3     BEGIN
;  289          0286  3     rab[rab$l_ubf] = new_pwd_buf;
;  290          0287  3     IF NOT get_record(CSTRING(%CHAR(13),%CHAR(10),'New password: '), rab)
```

```
;  291          0288  3       THEN RETURN set$_pwdnotver;
;  292          0289  3     new_pwd_dsc[0] = .rab[rab$w_rsz];
;  293          0290  3     IF .rab[rab$w_rsz] GTRU 31                          ! Make sure password not to long.
;  294          0291  3         THEN RETURN set$_invpwdlen;
;  295          0292  3     IF NOT .setpwd$flags[qual_system] AND               ! If not SYSTEM password,
;  296          0293  3        .rab[rab$w_rsz] LSSU .uafbuf[uaf$b_pwd_length]   !  then check to make sure not to shord.
;  297          0294  3     THEN
;  298          0295  4         BEGIN
;  299          0296  5         IF NOT (.setpwd$flags[qual_secondary] AND       ! Null password allowed
;  300          0297  5                 .rab[rab$w_rsz] EQL 0)                  ! on SECONDARY password.
;  301          0298  4           THEN RETURN set$_invpwdlen;
;  302          0299  3         END;
;  303          0300  3
;  304          0301  3     INCR i FROM 0 TO .new_pwd_dsc[0] - 1        ! Check new password for illegal characters.  The only allow
;  305          0302  3     DO                                          ! characters are A to Z, 0 to 9, the $ and _.
;  306          0303  4         BEGIN
;  307          0304  4         BIND
;  308          0305  4             char = new_pwd_buf[.i] : BYTE;
;  309          0306  4
;  310          0307  5         IF (.char GEQU %C'0' AND .char LEQU %C'9')
;  311          0308  5         OR (.char GEQU %C'A' AND .char LEQU %C'Z')
;  312          0309  4         OR .char EQLU %C'$'
;  313          0310  4         OR .char EQLU %C'_'
;  314          0311  4         THEN 1
;  315          0312  4         ELSE RETURN set$_pwdsyntax;
;  316          0313  3         END;
;  317          0314  2     END;
;  318          0315  2
;  319          0316  2 !
;  320          0317  2 ! Make sure user knows what he typed in.  Do this by asking for the
;  321          0318  2 ! new password again, and then checking that it's the same as the
;  322          0319  2 ! last one typed in.
;  323          0320  2 !
;  324          0321  2 rab[rab$l_ubf] = vfy_pwd_buf;
;  325          0322  2 IF NOT get_record(CSTRING(%CHAR(13),%CHAR(10),'Verification: '), rab)
;  326          0323  2 THEN RETURN set$_pwdnotver;
;  327          0324  2
;  328          0325  2 IF .rab[rab$w_rsz] NEQU .new_pwd_dsc[0]
;  329          0326  2 THEN RETURN set$_pwdnotver;
;  330          0327  2
;  331          0328  2 IF CH$NEQ(.rab[rab$w_rsz],
;  332          0329  2           .new_pwd_dsc[1],
;  333          0330  2           .rab[rab$w_rsz],
;  334          0331  2           vfy_pwd_buf)
;  335          0332  2 THEN RETURN set$_pwdnotver;
;  336          0333  2
;  337          0334  2 RETURN true;
;  338          0335  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                54  55  50  4E  49  24  53  59  53  00028 P.AAE:  .ASCII  \SYS$INPUT\                    ;
                                                    00031        .BLKB   3
                                                03  00034 P.AAF:  .BYTE   3                           ;
                                                50  00035        .BYTE   80                           ;
```

```
                0000  00036          .WORD   0
            00000000  00038          .LONG   0
            00000000  0003C          .LONG   0
            00000000  00040          .LONG   0
            00000000  00044          .LONG   0
                0000  00048          .WORD   0
                  02  0004A          .BYTE   2
                  00  0004B          .BYTE   0
            00000000  0004C          .LONG   0
                  00  00050          .BYTE   0
                  00  00051          .BYTE   0
                  00  00052          .BYTE   0
                  02  00053          .BYTE   2
            00000000  00054          .LONG   0
            00000000  00058          .LONG   0
            00000000  0005C          .LONG   0
            00000000' 00060          .ADDRESS P.AAE
            00000000  00064          .LONG   0
                  09  00068          .BYTE   9
                  00  00069          .BYTE   0
                0000  0006A          .WORD   0
            00000000  0006C          .LONG   0
                0000  00070          .WORD   0
                  00  00072          .BYTE   0
                  00  00073          .BYTE   0
            00000000  00074          .LONG   0
            00000000  00078          .LONG   0
                0000  0007C          .WORD   0
                  00  0007E          .BYTE   0
                  00  0007F          .BYTE   0
            00000000  00080          .LONG   0
                  01  00084 P.AAG:   .BYTE   1
                  44  00085          .BYTE   68
                0000  00086          .WORD   0
            45000000  00088          .LONG   1157627904
            00000000  0008C          .LONG   0
            00000000  00090          .LONG   0
                0000# 00094          .WORD   0[3]
                0000  0009A          .WORD   0
            00000000  0009C          .LONG   0
                0000  000A0          .WORD   0
                  00  000A2          .BYTE   0
                  00  000A3          .BYTE   0
                0020  000A4          .WORD   32
                0000  000A6          .WORD   0
            00000000  000A8          .LONG   0
            00000000  000AC          .LONG   0
            00000000  000B0          .LONG   0
            00000000  000B4          .LONG   0
                  00  000B8          .BYTE   0
                  00  000B9          .BYTE   0
                  00  000BA          .BYTE   0
                  00  000BB          .BYTE   0
            00000000  000BC          .LONG   0
            00000000  000C0          .LONG   0
            00000000  000C4          .LONG   0
                  0E  000C8 P.AAH:   .BYTE   14
```

```
   20 3A 64 72 6F 77 73 73 61 70 20 64 6C 4F 000C9            .ASCII  \Old password: \                    ;
                                          10 000D7 P.AAI:    .BYTE   16                                  ;
3A 64 72 6F 77 73 73 61 70 20 77 65 4E 0A 0D 000D8            .ASCII  <13><10>\New passwora: \            ;
                                          20 000E7                                                       ;
                                          10 000E8 P.AAJ:    .BYTE   16                                  ;
3A 6E 6F 69 74 61 63 69 66 69 72 65 56 0A 0D 000E9            .ASCII  <13><10>\Verification: \            ;
                                          20 000F8                                                       ;

                                                   NEW_PASSWORD=        NEW_PWD_BUF
                                                             .EXTRN  SYS$OPEN, SYS$CONNECT

                                                             .PSECT  $CODE$,NOWRT,2

                                        00FC 00000 GET_PWD:.WORD     Save R2,R3,R4,R5,R6,R7              ; 0231
                   57        0000V   CF   9E 00002           MOVAB   GET_RECORD, R7                      ;
                   56        0000'   CF   9E 00007           MOVAB   NEW_PWD_DSC, R6                     ;
                   5E        FF6C    CE   9E 0000C           MOVAB   -148(SP), SP                        ;
      44    AE     0000'   CF        0050 8F   28 00011      MOVC3   #80, P.AAF, FAB                     ; 0249
            6E     0000'   CF        0044 8F   28 0001A      MOVC3   #68, P.AAG, RAB                     ; 0252
                   3C      AE          44 AE   9E 00022      MOVAB   FAB, RAB+60                         ; 0249
                                       44 AE   9F 00027      PUSHAB  FAB                                 ; 0257
           00000000G       00             01   FB 0002A      CALLS   #1, SYS$OPEN                        ;
                           1D             50   E9 00031      BLBC    R0, 1$                              ;
                                          5E   DD 00034      PUSHL   SP                                  ; 0260
           00000000G       00             01   FB 00036      CALLS   #1, SYS$CONNECT                     ;
                           11             50   E9 0003D      BLBC    R0, 1$                              ;
                   24      AE          98 A6   9E 00040      MOVAB   OLD_PWD_BUF, RAB+36                 ; 0266
                                          5E   DD 00045      PUSHL   SP                                  ; 0267
                                 0000'    CF   9F 00047      PUSHAB  P.AAH                               ;
                           67             02   FB 0004B      CALLS   #2, GET_RECORD                      ;
                           08             50   E8 0004E      BLBS    R0, 2$                              ;
                           50 00000000G   8F   D0 00051 1$:  MOVL    #SET$_PWDNOTVAL, R0                 ; 0268
                                               04 00058      RET                                         ;
                   F8      A6          22 AE   3C 00059 2$:  MOVZWL  RAB+34, OLD_PWD_DSC                 ; 0269
            22     90      A6             01   E1 0005E      BBC     #1, SETPWD$FLAGS, 3$                ; 0275
                                       94 A6   9F 00063      PUSHAB  MIN_PWD_LENGTH                      ; 0279
                                       B8 A6   9F 00066      PUSHAB  NEW_PWD_BUF                         ;
           00000000G       00             02   FB 00069      CALLS   #2, SET_PASSWORD_GENERATE           ;
                           66          B8 A6   3C 00070      MOVZWL  NEW_PASSWORD, NEW_PWD_DSC           ; 0280
                   04      A6             02   C0 00074      ADDL2   #2, NEW_PWD_DSC+4                   ; 0281
                                          56   DD 00078      PUSHL   R6                                  ; 0282
                                          56   DD 0007A      PUSHL   R6                                  ;
           00000000G       00             02   FB 0007C      CALLS   #2, STR$UPCASE                      ;
                                          73   11 00083      BRB     10$                                 ; 0275
                   24      AE          B8 A6   9E 00085 3$:  MOVAB   NEW_PWD_BUF, RAB+36                 ; 0286
                                          5E   DD 0008A      PUSHL   SP                                  ; 0287
                                 0000'    CF   9F 0008C      PUSHAB  P.AAI                               ;
                           67             02   FB 00090      CALLS   #2, GET_RECORD                      ;
                           70             50   E9 00093      BLBC    R0, 11$                             ;
                           66          22 AE   3C 00096      MOVZWL  RAB+34, NEW_PWD_DSC                 ; 0289
                           1F          22 AE   B1 0009A      CMPW    RAB+34, #31                         ; 0290
                                          19   1A 0009E      BGTRU   4$                                  ;
                           1D          90 A6   E8 000A0      BLBS    SETPWD$FLAGS, 5$                    ; 0292
                           50        0172 C6   9A 000A4      MOVZBL  UAFBUF+362, R0                      ; 0293
                   22      AE             50   B1 000A9      CMPW    R0, RAB+34                          ;
                                          12   1B 000AD      BLEQU   5$                                  ;
            05     90      A6             02   E1 000AF      BBC     #2, SETPWD$FLAGS, 4$                ; 0296
```

```
                                    22   AE   B5 000B4          TSTW    RAB+34                                  ; 0297
                                         08   13 000B7          BEQL    5$
                    50 00000000G    8F   D0 000B9 4$:       MOVL    #SET$_INVPWDLEN, R0                     ; 0298
                                         04 000C0           RET
                    51              01   CE 000C1 5$:       MNEGL   #1, I                                   ; 0301
                                    2E   11 000C4           BRB     9$
                    50         B8 A641   9A 000C6 6$:       MOVZBL  NEW_PWD_BUF[I], R0                      ; 0307
                    30              50   91 000CB           CMPB    R0, #48
                                    05   1F 000CE           BLSSU   7$
                    39              50   91 000D0           CMPB    R0, #57
                                    1F   1B 000D3           BLEQU   9$
                41  8F              50   91 000D5 7$:       CMPB    R0, #65                                 ; 0308
                                    06   1F 000D9           BLSSU   8$
                5A  8F              50   91 000DB           CMPB    R0, #90
                                    13   1B 000DF           BLEQU   9$
                    24              50   91 000E1 8$:       CMPB    R0, #36                                 ; 0309
                                    0E   13 000E4           BEQL    9$
                5F  8F              50   91 000E6           CMPB    R0, #95                                 ; 0310
                                    08   13 000EA           BEQL    9$
                    50 00000000G    8F   D0 000EC           MOVL    #SET$_PWDSYNTAX, R0                     ; 0312
                                         04 000F3           RET
            CE      51              66   F2 000F4 9$:       AOBLSS  NEW_PWD_DSC, I, 6$                      ; 0301
        24  AE      D8              A6   9E 000F8 10$:      MOVAB   VFY_PWD_BUF, RAB+36                     ; 0321
                                         5E   DD 000FD      PUSHL   SP                                      ; 0322
                        0000'       CF   9F 000FF          PUSHAB  P.AAJ
                    67              02   FB 00103          CALLS   #2, GET_RECORD
                    11              50   E9 00106 11$:      BLBC    R0, 12$
    66      22  AE                  00   ED 00109          CMPZV   #0, #16, RAB+34, NEW_PWD_DSC            ; 0325
                                         09   12 0010F      BNEQ    12$
            D8  A6      04      B6      22   AE   29 00111      CMPC3   RAB+34, @NEW_PWD_DSC+4, VFY_PWD_BUF    ; 0328
                                         08   13 00118      BEQL    13$
                    50 00000000G    8F   D0 0011A 12$:      MOVL    #SET$_PWDNOTVER, R0                     ; 0332
                                         04 00121           RET
                    50              01   D0 00122 13$:      MOVL    #1, R0                                  ; 0334
                                         04 00125           RET                                             ; 0335

; Routine Size:  294 bytes,    Routine Base:  $CODE$ + 005B
```

```
;  340          0336  1 ROUTINE get_record (prompt_string, rab) =
;  341          0337  2 BEGIN
;  342          0338  2 !+++
;  343          0339  2 !
;  344          0340  2 ! Given a prompt string and a RAB address, get an input record.
;  345          0341  2 !
;  346          0342  2 ! Inputs:
;  347          0343  2 !       prompt_string - address of ASCIC prompt string
;  348          0344  2 !       rab - address of RAB
;  349          0345  2 !
;  350          0346  2 ! Outputs:
;  351          0347  2 !       The RAB's buffer will be filled in with a password.
;  352          0348  2 !
;  353          0349  2 !---
;  354          0350  2
;  355          0351  2 MAP
;  356          0352  2     rab : REF $BBLOCK,
;  357          0353  2     prompt_string : REF VECTOR[,BYTE];
;  358          0354  2
;  359          0355  2 !
;  360          0356  2 ! Set the specified prompt string and perform the $GET.
;  361          0357  2 !
;  362          0358  2 rab[rab$b_psz] = .prompt_string[0];             ! Prompt size
;  363          0359  2 rab[rab$l_pbf] = prompt_string[1];              ! and address
;  364          0360  2
;  365          0361  2 RETURN $GET(RAB = .rab);                        ! Return status of $GET
;  366          0362  1 END;


                                                        .EXTRN  SYS$GET

                                        0000 00000 GET_RECORD:
                                                        .WORD   Save nothing                            ;  0336
                                50      08  AC  D0 00002          MOVL    RAB, R0                        ;  0358
                            34  A0      04  BC  90 00006          MOVB    @PROMPT_STRING, 52(R0)
                        30  A0      04  AC      01  C1 0000B      ADDL3   #1, PROMPT_STRING, 48(R0)      ;  0359
                                            50  DD 00011          PUSHL   R0                             ;  0361
                                00000000G   00      01  FB 00013      CALLS   #1, SYS$GET
                                                        04 0001A        RET                             ;  0362

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 0181

;  367          0363  1
```

```
;  369          0364  1 ROUTINE update_uaf : NOVALUE =
;  370          0365  2 BEGIN
;  371          0366  2 !+++
;  372          0367  2 !
;  373          0368  2 ! Get the UAF record, verify that the user typed in the correct password,
;  374          0369  2 ! and if s/he did, then replace the old password with the new one.
;  375          0370  2 !
;  376          0371  2 ! Inputs:
;  377          0372  2 !       new_pwd_dsc - descriptor for new password
;  378          0373  2 !       old_pwd_dsc - descriptor for old password
;  379          0374  2 !
;  380          0375  2 ! Outputs:
;  381          0376  2 !       None.  The UAF is updated.
;  382          0377  2 !
;  383          0378  2 !---
;  384          0379  2
;  385          0380  2 LOCAL
;  386          0381  2     status,
;  387          0382  2     enc_buf: VECTOR[uaf$s_pwd,byte],
;  388          0383  2     enc_dsc: VECTOR[2] INITIAL(uaf$s_pwd, enc_buf),
;  389          0384  2     password,
;  390          0385  2     password_date,
;  391          0386  2     encrypt_byte : REF vector[,byte];
;  392          0387  2
;  393          0388  2 !
;  394          0389  2 ! Determine whether we are updating fields for the Primary or Secondary
;  395          0390  2 ! password.
;  396          0391  2 !
;  397          0392  2 IF NOT .setpwd$flags[qual_secondary]
;  398          0393  2 THEN
;  399          0394  3     BEGIN
;  400          0395  3     password = uafbuf[uaf$q_pwd];
;  401          0396  3     password_date = uafbuf[uaf$q_pwd_date];
;  402          0397  3     encrypt_byte = uafbuf[uaf$b_encrypt];
;  403          0398  3     END
;  404          0399  2 ELSE
;  405          0400  3     BEGIN
;  406          0401  3     password = uafbuf[uaf$q_pwd2];
;  407          0402  3     password_date = uafbuf[uaf$q_pwd2_date];
;  408          0403  3     encrypt_byte = uafbuf[uaf$b_encrypt2];
;  409          0404  2     END;
;  410          0405  2
;  411          0406  2 !
;  412          0407  2 ! Encrypt what the user says is the old password.  Then compare that to
;  413          0408  2 ! what is found in the UAF record.
;  414          0409  2 !
;  415          0410  2 LGI$HPWD(enc_dsc,                       ! Encrypt what the user typed in
;  416          0411  2      old_pwd_dsc,
;  417          0412  2      .encrypt_byte[0],
;  418          0413  2      .uafbuf[uaf$w_salt],
;  419          0414  2      user_desc);
;  420          0415  2
;  421          0416  2 IF CH$NEQ(uaf$s_pwd,                    ! Check to see if encrypted
;  422          0417  2           enc_buf,                     ! passwords match.
;  423          0418  2           uaf$s_pwd,
;  424          0419  2           .password)
;  425          0420  2 THEN                                   ! If they don't match, tell
```

```
;  426     0421  3     BEGIN                                   ! the user.
;  427     0422  3     SIGNAL(set$_pwdnotval);
;  428     0423  3     RETURN;
;  429     0424  2     END;
;  430     0425  2
;  431     0426  2
;  432     0427  2 !
;  433     0428  2 ! Compare the old password to the new password.  If they match, this is illegal,
;  434     0429  2 ! so report an error and return.
;  435     0430  2 !
;  436     0431  3 IF (.new_pwd_dsc[0] EQL .old_pwd_dsc[0])
;  437     0432  2   AND CH$EQL(.old_pwd_dsc[0],.old_pwd_dsc[1],.new_pwd_dsc[0],.new_pwd_dsc[1])
;  438     0433  2 THEN
;  439     0434  3     BEGIN
;  440     0435  3     SIGNAL(set$_pwdnotdif);
;  441     0436  3     RETURN;
;  442     0437  2     END;
;  443     0438  2
;  444     0439  2 !
;  445     0440  2 ! Encrypt the new password, and put it into the UAF record.  If the salt field
;  446     0441  2 ! in the uaf is zero, generate a new salt before the encryption.
;  447     0442  2 !
;  448     0443  2 IF .uafbuf[uaf$w_salt] EQL 0            ! Is salt zero?
;  449     0444  2 THEN
;  450     0445  3     BEGIN
;  451     0446  3     LOCAL time : VECTOR[4,WORD];        ! Get local quadword.
;  452     0447  3     $GETTIM(TIMADR = time);             ! Get current time.
;  453     0448  3     uafbuf[uaf$w_salt] = .time[1];      ! Salt is now 2nd word of time.
;  454     0449  2     END;
;  455     0450  2
;  456     0451  2 LGI$HPWD(enc_dsc,                       ! Encrypt new password
;  457     0452  2      new_pwd_dsc,
;  458     0453  2      uaf$c_purdy_v,
;  459     0454  2       uafbuf[uaf$w_salt],
;  460     0455  2      user_desc);
;  461     0456  2
;  462     0457  2 !
;  463     0458  2 ! Move new password into record, clear password expired bit, reset password
;  464     0459  2 ! change date, and make sure we always use newest encryption algorithm for
;  465     0460  2 ! PRIMARY or SECONDARY password.
;  466     0461  2 !
;  467     0462  2 CH$MOVE(uaf$s_pwd,enc_buf,.password);   ! Move new password to record.
;  468     0463  2 encrypt_byte[0] = uaf$c_purdy_v;        ! Use PURDY_V encryption algorithm.
;  469     0464  2 IF .new_pwd_dsc[0] EQL 0                ! Move new time (0 if password is is null)
;  470     0465  2     THEN CH$FILL(0,uaf$s_pwd_date,.password_date)
;  471     0466  2     ELSE $GETTIM(TIMADR = .password_date);
;  472     0467  2
;  473     0468  2 IF NOT .setpwd$flags[qual_secondary]
;  474     0469  2     THEN uafbuf[uaf$v_pwd_expired] = false      ! Clear expired bit.
;  475     0470  2     ELSE uafbuf[uaf$v_pwd2_expired] = false;    ! Clear expired bit.
;  476     0471  2
;  477     0472  3 IF NOT $PUT(RAB = uafrab)               ! Now update (UIF) the UAF file.
;  478     0473  2     THEN SIGNAL(set$_uaferr,0,          ! If an error, tell the user.
;  479     0474  2             .uafrab[rab$l_sts],
;  480     0475  2             .uafrab[rab$l_stv]);
;  481     0476  2
;  482     0477  2 $CLOSE(FAB = uaffab);                   ! Close the UAF.
```

```
;  483        0478  2
;  484        0479  2 RETURN;
;  485        0480  1 END;


                                                              .EXTRN  SYS$GETTIM, SYS$PUT
                                                              .EXTRN  SYS$CLOSE

                                             0FFC 00000 UPDATE_UAF:
                                                              .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0364
                         5B 00000000G  00  9E 00002           MOVAB   SYS$GETTIM, R11                             ;
                         5A 00000000G  00  9E 00009           MOVAB   LIB$SIGNAL, R10                             ;
                         59 00000000G  00  9E 00010           MOVAB   LGI$HPWD, R9                                ;
                         58     0000'  CF  9E 00017           MOVAB   UAFBUF+358, R8                              ;
                         5E            18  C2 0001C           SUBL2   #24, SP                                     ;
                   08    AE            08  D0 0001F           MOVL    #8, ENC_DSC                                 ; 0365
                   0C    AE      10    AE  9E 00023           MOVAB   ENC_BUF, ENC_DSC+4                          ;
          0E     FE22    C8            02  E0 00028           BBS     #2, SETPWD$FLAGS, 1$                        ; 0392
                         54      EE    A8  9E 0002E           MOVAB   UAFBUF+340, PASSWORD                        ; 0395
                         57      16    A8  9E 00032           MOVAB   UAFBUF+380, PASSWORD_DATE                   ; 0396
                         56      02    A8  9E 00036           MOVAB   UAFBUF+360, ENCRYPT_BYTE                    ; 0397
                                       0C  11 0003A           BRB     2$                                          ; 0392
                         54      F6    A8  9E 0003C 1$:       MOVAB   UAFBUF+348, PASSWORD                        ; 0401
                         57      1E    A8  9E 00040           MOVAB   UAFBUF+388, PASSWORD_DATE                   ; 0402
                         56      03    A8  9E 00044           MOVAB   UAFBUF+361, ENCRYPT_BYTE                    ; 0403
                               041E    C8  9F 00048 2$:       PUSHAB  USER_DESC                                   ; 0410
                         7E            68  3C 0004C           MOVZWL  UAFBUF+358, -(SP)                           ; 0413
                         7E            66  9A 0004F           MOVZBL  (ENCRYPT_BYTE), -(SP)                       ; 0412
                               FE8A    C8  9F 00052           PUSHAB  OLD_PWD_DSC                                 ; 0410
                                 18    AE  9F 00056           PUSHAB  ENC_DSC                                     ;
                         69            05  FB 00059           CALLS   #5, LGI$HPWD                                ;
          64       10    AE            08  29 0005C           CMPC3   #8, ENC_BUF, (PASSWORD)                     ; 0416
                                       08  13 00061           BEQL    3$                                          ;
                            00000000G  8F  DD 00063           PUSHL   #SET$_PWDNOTVAL                             ; 0422
                                       1F  11 00069           BRB     4$                                          ;
                 FE8A    C8    FE92    C8  D1 0006B 3$:       CMPL    NEW_PWD_DSC, OLD_PWD_DSC                    ; 0431
                                       1A  12 00072           BNEQ    5$                                          ;
FE92  C8  00     FE8E    D8    FE8A    C8  2D 00074           CMPC5   OLD_PWD_...., @OLD_PWD_DSC+4, #0, -         ; 0432
                               FE96    D8     0007F                   NEW_PWD_DSC, @NEW_PWD_DSC+4                 ;
                                       0A  12 00082           BNEQ    5$                                          ;
                            00000000G  8F  DD 00084           PUSHL   #SET$_PWDNOTDIF                             ; 0435
                         6A            01  FB 0008A 4$:       CALLS   #1, LIB$SIGNAL                              ;
                                       04     0008D           RET                                                 ; 0434
                                       68  B5 0008E 5$:       TSTW    UAFBUF+358                                  ; 0443
                                       09  12 00090           BNEQ    6$                                          ;
                                       5E  DD 00092           PUSHL   SP                                          ; 0447
                         6B            01  FB 00094           CALLS   #1, SYS$GETTIM                              ;
                         68      02    AE  B0 00097           MOVW    TIME+2, UAFBUF+358                          ; 0448
                               041E    C8  9F 0009B 6$:       PUSHAB  USER_DESC                                   ; 0451
                         7E            68  3C 0009F           MOVZWL  UAFBUF+358, -(SP)                           ; 0454
                                       02  DD 000A2           PUSHL   #2                                          ; 0451
                               FE92    C8  9F 000A4           PUSHAB  NEW_PWD_DSC                                 ;
                                 18    AE  9F 000A8           PUSHAB  ENC_DSC                                     ;
                         69            05  FB 000AB           CALLS   #5, LGI$HPWD                                ;
          64       10    AE            08  28 000AE           MOVC3   #8, ENC_BUF, (PASSWORD)                     ; 0462
                         66            02  90 000B3           MOVB    #2, (ENCRYPT_BYTE)                          ; 0463
```

```
                                        FE92    C8  D5 000B6          TSTL    NEW_PWD_DSC                                     ; 0464
                                                C8  12 000BA          BNEQ    7$
            08          00              6E      00  2C 000BC          MOVC5   #0, (SP), #0, #8, (PASSWORD_DATE)               ; 0465
                                                67     000C1
                                                05  11 000C2          BRB     8$
                                                57  DD 000C4 7$:      PUSHL   PASSWORD_DATE                                   ; 0466
                                            6B  01  FB 000C6          CALLS   #1, SYS$GETTIM
                                06      FE22    C8  02  E0 000C9 8$:      BBS     #2, SETPWD$FLAGS, 9$                        ; 0468
                                          6F    A8  02  8A 000CF          BICB2   #2, UAFBUF+469                              ; 0469
                                                04  11 000D3          BRB     10$
                                          6F    A8  04  8A 000D5 9$:      BICB2   #4, UAFBUF+469                              ; 0470
                                                0476    C8  9F 000D9 10$:     PUSHAB  UAFRAB                                  ; 0472
                                    00000000G   00  01  FB 000DD          CALLS   #1, SYS$PUT
                                                10  50  E8 000E4          BLBS    R0, 11$
                                                7E      047E    C8  7D 000E7          MOVQ    UAFRAB+8, -(SP)                 ; 0474
                                                        7E  D4 000EC          CLRL    -(SP)                                   ; 0473
                                                00000000G   8F  DD 000EE          PUSHL   #SET$_UAFERR
                                                6A  04  FB 000F4          CALLS   #4, LIB$SIGNAL
                                                0426    C8  9F 000F7 11$:     PUSHAB  UAFFAB                                  ; 0477
                                    00000000G   00  01  FB 000FB          CALLS   #1, SYS$CLOSE
                                                    04  00102          RET                                                     ; 0480

; Routine Size:  259 bytes,    Routine Base:  $CODE$ + 019C
```

```
;  487          0481  1 ROUTINE update_sys : NOVALUE =
;  488          0482  2 BEGIN
;  489          0483  2 !+++
;  490          0484  2 !
;  491          0485  2 ! Update the encrypted system password.
;  492          0486  2 !
;  493          0487  2 ! Inputs:
;  494          0488  2 !       old_pwd_dsc - descriptor of old password
;  495          0489  2 !       new_pwd_dsc = descriptor of new password
;  496          0490  2 !
;  497          0491  2 !--
;  498          0492  2
;  499          0493  2 !
;  500          0494  2 ! First, check to see if the user has the SECURITY privilege.
;  501          0495  2 !
;  502          0496  2 IF NOT .ctl$gq_procpriv[prv$v_security] THEN
;  503          0497  2     SIGNAL_STOP(set$_syspwderr, ss$_nosecurity);
;  504          0498  2
;  505          0499  2 !
;  506          0500  2 ! Make sure the special record has everthing it needs
;  507          0501  2 !
;  508          0502  2 uafbuf[uaf$b_encrypt] = uaf$c_purdy_v;
;  509          0503  2
;  510          0504  2 RETURN;
;  511          0505  1 END;

                                0000 00000 UPDATE_SYS:
                                           .WORD   Save nothing                          ; 0481
                12 00000000G    00         06   E0 00002        BBS     #6, CTL$GQ_PROCPRIV+4, 1$  ; 0496
                                7E    2934 8F   3C 0000A        MOVZWL  #10548, -(SP)              ; 0497
                                00000000G  8F   DD 0000F        PUSHL   #SET$_SYSPWDERR            ;
                   00000000G    00         02   FB 00015        CALLS   #2, LIB$STOP               ;
                       0000'    CF         02   90 0001C 1$:     MOVB    #2, UAFBUF+360             ; 0502
                                           04   00021           RET                                ; 0505

; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 029F
```

```
;  513          0506  1 ROUTINE check_qualifiers =
;  514          0507  2 BEGIN
;  515          0508  2 !+++
;  516          0509  2 !
;  517          0510  2 ! Get the minimum password length prior to generating passwords.
;  518          0511  2 !
;  519          0512  2 ! Inputs:
;  520          0513  2 !
;  521          0514  2 !       none
;  522          0515  2 !
;  523          0516  2 ! Outputs:
;  524          0517  2 !
;  525          0518  2 !       min_pwd_length = minimum password length to generate.
;  526          0519  2 !
;  527          0520  2 !---
;  528          0521  2
;  529          0522  2 LOCAL
;  530          0523  2     infile_desc: $BBLOCK [dsc$c_s_bln] INITIAL(0),
;  531          0524  2     status;
;  532          0525  2
;  533          0526  2 infile_desc[dsc$b_class] = dsc$k_class_d;              ! Make descriptor dynamic
;  534          0527  2
;  535          0528  2 !
;  536          0529  2 ! Get qualifiers
;  537          0530  2 !
;  538          0531  2 setpwd$flags[qual_generate]     = cli$present($descriptor('GENERATE'));
;  539          0532  2 setpwd$flags[qual_system]       = cli$present($descriptor('SYSTEM'));
;  540          0533  2 setpwd$flags[qual_secondary]    = cli$present($descriptor('SECONDARY'))
;  541          0534  2                                   AND NOT setpwd$flags[qual_system];
;  542          0535  2
;  543          0536  2 !
;  544          0537  2 ! If GENPWD bit set in uaf record, then user must use password generator.
;  545          0538  2 !
;  546          0539  2 IF NOT .setpwd$flags[qual_system]
;  547          0540  2     THEN setpwd$flags[qual_generate] = .setpwd$flags[qual_generate] OR .uafbuf[uaf$v_genpwd];
;  548          0541  2
;  549          0542  2 ! If /GENERATE not set, we are done.
;  550          0543  2 !
;  551          0544  2 IF NOT .setpwd$flags[qual_generate]                    ! /GENERATE specified?
;  552          0545  2     THEN RETURN TRUE;                                 ! No, exit with success.
;  553          0546  2
;  554          0547  2 !
;  555          0548  2 ! Calculate minimum length of word to be generated by password generator.
;  556          0549  2 !
;  557          0550  2 IF status = cli$get_value($descriptor('GENERATE'), infile_desc) ! Get value for /GENERATE if one specified.
;  558          0551  2 THEN
;  559          0552  3     BEGIN
;  560          0553  3     lib$cvt_dtb(.infile_desc [dsc$w_length],          ! Move value into Global variable
;  561          0554  3                 .infile_desc [dsc$a_pointer], min_pwd_length);
;  562          0555  4     IF (.min_pwd_length LEQ 0) OR (.min_pwd_length GTR 10)   ! If size LEQ 0 or GTR 10
;  563          0556  3             THEN SIGNAL_STOP (set$_badvalue,1,infile_desc);  !  report an error.
;  564          0557  3     END
;  565          0558  2 ELSE
;  566          0559  2     min_pwd_length = 6;                           ! No value, default to 6.
;  567          0560  2
;  568          0561  2 !
;  569          0562  2 ! Minimum password length for /GENERATE is larger of current minimum password length versus
```

```
;  570          0563  2 ! minimum password length specified in UAF, but no greater than 10.
;  571          0564  2 !
;  572          0565  2 IF NOT setpwd$flags[qual_system] THEN
;  573          0566  2     min_pwd_length = MAX(.uafbuf[uaf$b_pwd_length],.min_pwd_length);
;  574          0567  2 min_pwd_length = MIN(.min_pwd_length,10);
;  575          0568  2
;  576          0569  2 RETURN true;
;  577          0570  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                    45  54  41  52  45  4E  45  47  000F9 P.AAL:  .ASCII  \GENERATE\                    ;
                                                00101           .BLKB   3                             ;
                                    00000008    00104 P.AAK:    .LONG   8                             ;
                                    00000000'   00108           .ADDRESS P.AAL                        ;
                            4D  45  54  53  59  53  0010C P.AAN:  .ASCII  \SYSTEM\                     ;
                                                00112           .BLKB   2                             ;
                                    00000006    00114 P.AAM:    .LONG   6                             ;
                                    00000000'   00118           .ADDRESS P.AAN                        ;
                59  52  41  44  4E  4F  43  45  53  0011C P.AAP:  .ASCII  \SECONDARY\                  ;
                                                00125           .BLKB   3                             ;
                                    00000009    00128 P.AAO:    .LONG   9                             ;
                                    00000000'   0012C           .ADDRESS P.AAP                        ;
                    45  54  41  52  45  4E  45  47  00130 P.AAR:  .ASCII  \GENERATE\                   ;
                                    00000008    00138 P.AAQ:    .LONG   8                             ;
                                    00000000'   0013C           .ADDRESS P.AAR                        ;


                                                                .PSECT  $CODE$,NOWRT,2

                                                0001C 00000 CHECK_QUALIFIERS:
                                                            .WORD   Save R2,R3,R4                     ; 0506
                        54 00000000G    00  9E  00002       MOVAB   CLI$PRESENT, R4                   ;
                        53      0000'   CF  9E  00009       MOVAB   SETPWD$FLAGS, R3                  ;
                        5E              04  C2  0000E       SUBL2   #4, SP                            ;
                                        7E  D4  00011       CLRL    INFILE_DESC                       ; 0507
                                    04  AE  D4  00013       CLRL    INFILE_DESC+4                     ;
                    03  AE              02  90  00016       MOVB    #2, INFILE_DESC+3                 ; 0526
                            0000'   CF  9F  0001A           PUSHAB  P.AAK                             ; 0531
                        64              01  FB  0001E       CALLS   #1, CLI$PRESENT                   ;
            63          01              01  F0  00021       INSV    R0, #1, #1, SETPWD$FLAGS          ;
                            0000'   CF  9F  00026           PUSHAB  P.AAM                             ; 0532
                        64              01  FB  0002A       CALLS   #1, CLI$PRESENT                   ;
            63          01              00  F0  0002D       INSV    R0, #0, #1, SETPWD$FLAGS          ;
                            0000'   CF  9F  00032           PUSHAB  P.AAO                             ; 0533
                        64              01  FB  00036       CALLS   #1, CLI$PRESENT                   ;
                        51              63  9E  00039       MOVAB   SETPWD$FLAGS, R1                  ; 0534
                    52                  50  51  8B  0003C   BICB3   R1, R0, R2                       ;
            63          01              02  52  F0  00040   INSV    R2, #2, #1, SETPWD$FLAGS          ;
                                        14  63  E8  00045   BLBS    SETPWD$FLAGS, 1$                  ; 0539
            50          63              01  01  EF  00048   EXTZV   #1, #1, SETPWD$FLAGS, R0          ; 0540
            51  024D    C3              01  00  EF  0004D   EXTZV   #0, #1, UAFBUF+469, R1            ;
                                        50  51  88  00054   BISB2   R1, R0                            ;
            63          01              01  50  F0  00057   INSV    R0, #1, #1, SETPWD$FLAGS          ;
```

```
                6C           63              01 E1 0005C 1$:     BBC     #1, SETPWD$FLAGS, 8$                    ; 0544
                                             5E DD 00060         PUSHL   SP                                      ; 0550
                                       0000' CF 9F 00062         PUSHAB  P.AAQ                                   ;
                    00000000G  00            02 FB 00066         CALLS   #2, CLI$GET_VALUE                       ;
                               2F            50 E9 00069         BLBC    STATUS, 3$                              ;
                                        04   A3 9F 00070         PUSHAB  MIN_PWD_LENGTH                          ; 0553
                                        08   AE DD 00073         PUSHL   INFILE_DESC+4                           ; 0554
                               7E       08   AE 3C 00076         MOVZWL  INFILE_DESC, -(SP)                      ; 0553
                    00000000G  00            03 FB 0007A         CALLS   #3, LIB$CVT_DTB                         ;
                               50       04   A3 D0 00081         MOVL    MIN_PWD_LENGTH, R0                      ; 0555
                                             05 15 00085         BLEQ    2$                                      ;
                               0A            50 D1 00087         CMPL    R0, #10                                 ;
                                             17 15 0008A         BLEQ    4$                                      ;
                                             5E DD 0008C 2$:     PUSHL   SP                                      ; 0556
                                             01 DD 0008E         PUSHL   #1                                      ;
                                   00771112  3F DD 00090         PUSHL   #7803154                                ;
                    00000000G  00            03 FB 00096         CALLS   #3, LIB$STOP                            ;
                                             04 11 0009D         BRB     4$                                      ; 0550
                          04   A3            06 D0 0009F 3$:     MOVL    #6, MIN_PWD_LENGTH                      ; 0559
                               50            63 9E 000A3 4$:     MOVAB   SETPWD$FLAGS, R0                        ; 0565
                               13            50 E8 000A6         BLBS    R0, 6$                                  ;
                               50       01E2 C3 9A 000A9         MOVZBL  UAFBUF+362, R0                          ; 0566
                          04   A3            50 D1 000AE         CMPL    R0, MIN_PWD_LENGTH                      ;
                                             04 18 000B2         BGEQ    5$                                      ;
                               50       04   A3 D0 000B4         MOVL    MIN_PWD_LENGTH, R0                      ;
                          04   A3            50 D0 000B8 5$:     MOVL    R0, MIN_PWD_LENGTH                      ;
                               50       04   A3 D0 000BC 6$:     MOVL    MIN_PWD_LENGTH, R0                      ; 0567
                               0A            50 D1 000C0         CMPL    R0, #10                                 ;
                                             03 15 000C3         BLEQ    7$                                      ;
                               50            0A D0 000C5         MOVL    #10, R0                                 ;
                          04   A3            50 D0 000C8 7$:     MOVL    R0, MIN_PWD_LENGTH                      ;
                               50            01 D0 000CC 8$:     MOVL    #1, R0                                  ; 0569
                                             04 000CF            RET                                             ; 0570

; Routine Size:  208 bytes,    Routine Base:  $CODE$ + 02C1

;  578          0571  1
```

```
;  580          0572  1 ROUTINE open_uaf =
;  581          0573  2 BEGIN
;  582          0574  2 !***
;  583          0575  2 !
;  584          0576  2 ! Open the UAF file, Connect to it, and stuff the username of this process
;  585          0577  2 ! into the rab in preparation for the get of his record.
;  586          0578  2 !
;  587          0579  2 ! Inputs:
;  588          0580  2 !       uaffab - fab to access the UAF.
;  589          0581  2 !       uafrab - rab to access the UAF.
;  590          0582  2 !
;  591          0583  2 ! Outputs:
;  592          0584  2 !       None.
;  593          0585  2 !
;  594          0586  2 !---
;  595          0587  2
;  596          0588  2 LOCAL
;  597          0589  2     status,
;  598          0590  2     iosb : VECTOR[4,WORD],
;  599          0591  2     item_list : $ITMLST_DECL(ITEMS = 1);
;  600          0592  2
;  601          0593  2 !
;  602          0594  2 ! Only use EXEC logical name table when opening the authorization file.
;  603          0595  2 !
;  604          0596  2 uaffab[fab$v_lnm_mode] = psl$c_exec;
;  605          0597  2
;  606          0598  2 !
;  607          0599  2 ! Obtain the username of this process or the system.
;  608          0600  2 ! Must fill username buffer with blanks first.
;  609          0601  2 !
;  610          0602  2 CH$FILL (' ',uaf$s_username,uafbuf[uaf$t_username]);
;  611          0603  2
;  612          0604  2 IF cli$present(%ascid'SYSTEM') THEN
;  613          0605  3     BEGIN
;  614          0606  3         CH$MOVE(17, UPLIT('<System+Password>'), uafbuf[uaf$t_username]);
;  615          0607  3         user_desc[0] = 17;
;  616          0608  3     END
;  617          0609  2 ELSE
;  618          0610  3     BEGIN
;  619        P 0611  3         $ITMLST_INIT(ITMLST = item_list,
;  620        P 0612  3                      (ITMCOD = jpi$_username,
;  621        P 0613  3                       BUFSIZ = jib$s_username,
;  622        P 0614  3                       BUFADR = uafbuf[uaf$t_username],
;  623          0615  3                       RETLEN = user_desc[0]));
;  624        P 0616  3         $GETJPIW(ITMLST = item_list,
;  625          0617  3                  IOSB = IOSB);
;  626          0618  2     END;
;  627          0619  2
;  628          0620  2
;  629          0621  2 !
;  630          0622  2 ! Try to open the UAF.  If a file-sharing problem, try one more time.
;  631          0623  2 ! If that open doesn't work, then give up.
;  632          0624  2 !
;  633          0625  3 IF NOT (status = $OPEN(FAB = uaffab))
;  634          0626  2 THEN
;  635          0627  3     BEGIN
;  636          0628  3     IF .status EQL rms$_sne
```

```
3       OR .status EQL rms$_spe
3       OR .status EQL rms$_dme
3       THEN
4           BEGIN
4           uaffab[fab$b_shr] = 0;
5           IF NOT (status = $OPEN(FAB = uaffab))
4           THEN SIGNAL(set$_uaferr,0,
4                       .uaffab[fab$l_sts],
4                       .uaffab[fab$l_stv]);
4           END
3       ELSE SIGNAL(set$_uaferr,0,
3                   .uaffab[fab$l_sts],
3                   .uaffab[fab$l_stv]);
2       END;
2
2 !
2 ! Return if the open failed.
2 !
2 IF NOT .status
2     THEN RETURN .status;
2
2 !
2 ! Do the same sort of stuff for the connect.
2 !
3 IF NOT (status = $CONNECT(RAB = uafrab))
2 THEN
3     BEGIN
3     IF .status EQL rms$_crmp
3     THEN
4         BEGIN
4         uaffab[fab$w_gbc] = 0;
5         IF NOT (status = $CONNECT(RAB = uafrab))
4         THEN SIGNAL(set$_uaferr,0,
4                     .uafrab[rab$l_sts],
4                     .uafrab[rab$l_stv]);
4         END
3     ELSE SIGNAL(set$_uaferr,0,
3                 .uafrab[rab$l_sts],
3                 .uafrab[rab$l_stv]);
3     END;
2
2 RETURN .status;
1 END;
```

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

                     00 00 4D 45 54 53 59 53  00140 P.AAT:  .ASCII  \SYSTEM\<0><0>
                                   010E0006  00148 P.AAS:  .LONG   17694726
                                  00000000'  0014C          .ADDRESS P.AAT
72 6F 77 73 73 61 50 2B 6D 65 74 73 79 53 3C  00150 P.AAU:  .ASCII  \<System+Password>\<0><0><0>
                           00 00 00 3E 64  0015F

                                                            .EXTRN  SYS$GETJPIW

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

```
                                             07FC 00000 OPEN_UAF:
                                                              .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10                 ; 0572
                        5A 00000000G  00   9E 00002           MOVAB   SYS$CONNECT, R10                                 ;
                        59 00000000G  00   9E 00009           MOVAB   LIB$SIGNAL, R9                                   ;
                        58 00000000G  8F   D0 00010           MOVL    #SET$_UAFERR, R8                                 ;
                        57 00000000G  00   9E 00017           MOVAB   SYS$OPEN, R7                                     ;
                        56       0000'  CF   9E 0001E         MOVAB   UAFBUF+4, R6                                     ;
                        5E              18   C2 00023         SUBL2   #24, SP                                          ;
 05D2   C6        02    00              01   F0 00026         INSV    #1, #0, #2, UAFFAB+74                            ; 0596
        20        20    6E              00   2C 0002D         MOVC5   #0, (SP), #32, #32, UAFBUF+4                     ; 0602
                        66                 00032
                                 0000'  CF   9F 00033         PUSHAB  P.AAS                                            ; 0604
                 00000000G  00          01   FB 00037         CALLS   #1, CLI$PRESENT                                  ;
                            0D          50   E9 0003E         BLBC    R0, 1$                                           ;
                  66   0000'  CF        11   28 00041         MOVC3   #17, P.AAU, UAFBUF+4                             ; 0606
                       0580   C6        11   D0 00047         MOVL    #17, USER_DESC                                   ; 0607
                                        27   11 0004C         BRB     2$                                               ; 0604
                        50              6E   9E 0004E 1$:     MOVAB   ITEM_LIST, $$ITMBLKPTR                           ; 0615
                        80 0202000C     8F   D0 00051         MOVL    #33685516, ($$ITMBLKPTR)+                        ;
                        80              66   9E 00058         MOVAB   UAFBUF+4, ($$ITMBLKPTR)+                         ;
                        80       0580   C6   9E 0005B         MOVAB   USER_DESC, ($$ITMBLKPTR)+                        ;
                                        80   D4 00060         CLRL    ($$ITMBLKPTR)+                                   ;
                                        7E   7C 00062         CLRQ    -(SP)                                            ; 0617
                                   18   AE   9F 00064         PUSHAB  IOSB                                             ;
                                   0C   AE   9F 00067         PUSHAB  ITEM_LIST                                        ;
                                        7E   7C 0006A         CLRQ    -(SP)                                            ;
                                        7E   D4 0006C         CLRL    -(SP)                                            ;
                 00000000G  00          07   FB 0006E         CALLS   #7, SYS$GETJPIW                                  ;
                                 0588   C6   9F 00075 2$:     PUSHAB  UAFFAB                                           ; 0625
                        67              01   FB 00079         CALLS   #1, SYS$OPEN                                     ;
                        52              50   D0 0007C         MOVL    R0, STATUS                                       ;
                        3B              52   E8 0007F         BLBS    STATUS, 5$                                       ;
                 0001879C   8F          52   D1 00082         CMPL    STATUS, #100252                                  ; 0628
                                        12   13 00089         BEQL    3$                                               ;
                 000187A4   8F          52   D1 0008B         CMPL    STATUS, #100260                                  ; 0629
                                        09   13 00092         BEQL    3$                                               ;
                 000184D4   8F          52   D1 00094         CMPL    STATUS, #99540                                   ; 0630
                                        11   12 0009B         BNEQ    4$                                               ;
                                 059F   C6   94 0009D 3$:     CLRB    UAFFAB+23                                        ; 0633
                                 0588   C6   9F 000A1         PUSHAB  UAFFAB                                           ; 0634
                        67              01   FB 000A5         CALLS   #1, SYS$OPEN                                     ;
                        52              50   D0 000A8         MOVL    R0, STATUS                                       ;
                        0F              52   E8 000AB         BLBS    STATUS, 5$                                       ;
                        7E       0590   C6   7D 000AE 4$:     MOVQ    UAFFAB+8, -(SP)                                  ; 0640
                                        7E   D4 000B3         CLRL    -(SP)                                            ; 0639
                                        58   DD 000B5         PUSHL   R8                                               ;
                        69              04   FB 000B7         CALLS   #4, LIB$SIGNAL                                   ;
                        33              52   E9 000BA         BLBC    STATUS, 7$                                       ; 0647
                                 05D8   C6   9F 000BD 5$:     PUSHAB  UAFRAB                                           ; 0653
                        6A              01   FB 000C1         CALLS   #1, SYS$CONNECT                                  ;
                        52              50   D0 000C4         MOVL    R0, STATUS                                       ;
                        26              52   E8 000C7         BLBS    STATUS, 7$                                       ;
                 0001C14C   8F          52   D1 000CA         CMPL    STATUS, #115020                                  ; 0656
                                        11   12 000D1         BNEQ    6$                                               ;
                                 05D0   C6   B4 000D3         CLRW    UAFFAB+72                                        ; 0659
                                 05D8   C6   9F 000D7         PUSHAB  UAFRAB                                           ; 0660
```

```
                6A              01  FB 000DB          CALLS   #1, SYS$CONNECT                          ;
                52              50  D0 000DE          MOVL    R0, STATUS                              ;
                0C              52  E8 000E1          BLBS    STATUS, 7$                              ;
                7E      05E0    C6  7D 000E4 6$:      MOVQ    UAFRAB+8, -(SP)                         ; 0666
                                7E  D4 000E9          CLRL    -(SP)                                   ; 0665
                                58  DD 000EB          PUSHL   R8                                      ;
                69              04  FB 000ED          CALLS   #4, LIB$SIGNAL                          ;
                50              52  D0 000F0 7$:      MOVL    STATUS, R0                              ; 0670
                                    04 000F3          RET                                             ; 0671

; Routine Size:  244 bytes,    Routine Base:  $CODE$ + 0391

;  680          0672  1
```

```
;  682          0673  1 ROUTINE get_uaf_record =
;  683          0674  2 BEGIN
;  684          0675  2 !+++
;  685          0676  2 !
;  686          0677  2 ! Try to read the user record.  In the event that we get an error that says
;  687          0678  2 ! the record was locked, then go into nap mode, scheduling this process to
;  688          0679  2 ! wake up in 2 milliseconsds.  If, after two naps, we still have a problem,
;  689          0680  2 ! then forget it.
;  690          0681  2 !
;  691          0682  2 ! Inputs:
;  692          0683  2 !       uafrab - rab to access the UAF.
;  693          0684  2 !
;  694          0685  2 ! Outputs:
;  695          0686  2 !       None.
;  696          0687  2 !
;  697          0688  2 !---
;  698          0689  2
;  699          0690  2 LOCAL
;  700          0691  2     status;
;  701          0692  2
;  702          0693  3 IF NOT (status = $GET(RAB = uafrab))
;  703          0694  2 THEN
;  704          0695  3     BEGIN
;  705          0696  3     IF .status EQL rms$_rlk
;  706          0697  3     THEN INCR i FROM 1 TO 2 DO
;  707          0698  4         BEGIN
;  708          0699  5         IF $SCHDWK(DAYTIM = UPLIT(-200000, -1))
;  709          0700  4         THEN $HIBER();
;  710          0701  4         status = $GET(RAB = uafrab);
;  711          0702  4         IF .status NEQ rms$_rlk
;  712          0703  4         THEN EXITLOOP;
;  713          0704  4         END
;  714          0705  3     ELSE IF .status EQL rms$_rnf THEN
;  715          0706  4        BEGIN
;  716          0707  4           UAFRAB[RAB$L_RBF] = UAFBUF;
;  717          0708  4           UAFRAB[RAB$W_RSZ] = UAF$C_LENGTH;
;  718          0709  4           RETURN rms$_normal;
;  719          0710  3        END;
;  720          0711  2     END;
;  721          0712  2
;  722          0713  2 IF NOT .status
;  723          0714  2 THEN
;  724          0715  2     SIGNAL(set$_uaferr,0,
;  725          0716  2             .uafrab[rab$l_sts],
;  726          0717  2             .uafrab[rab$l_stv]);
;  727          0718  2
;  728          0719  2 RETURN .status;
;  729          0720  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
                            FFFFFFFF  FFFCF2C0  00164 P.AAV:  .LONG   -200000, -1                                ;
                                                        .EXTRN  SYS$SCHDWK, SYS$HIBER
```

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

                                        003C 00000 GET_UAF_RECORD:
                                                .WORD   Save R2,R3,R4,R5                ; 0673
                    55 00000000G    00  9E 00002         MOVAB   SYS$GET, R5
                    54      0000'   CF  9E 00009         MOVAB   UAFRAB, R4
                                    54  DD 0000E         PUSHL   R4                      ; 0693
                    65              01  FB 00010         CALLS   #1, SYS$GET
                    53              50  D0 00013         MOVL    R0, STATUS
                    72              53  E8 00016         BLBS    STATUS, 5$
            000182AA    8F          53  D1 00019         CMPL    STATUS, #98986          ; 0696
                                    36  12 00020         BNEQ    3$
                    52              01  D0 00022         MOVL    #1, I                   ; 0697
                                    7E  D4 00025 '$:      CLRL    -(SP)                   ; 0699
                            0000'   CF  9F 00027         PUSHAB  P.AAV
                                    7F  7C 0002B         CLRQ    -(SP)
            00000000G   00          04  FB 0002D         CALLS   #4, SYS$SCHDWK
                        0A          50  E9 00034         BLBC    R0, 2$
            00000000G   00          00  FB 00037         CALLS   #0, SYS$HIBER           ; 0700
                        60          00  FB 0003E         CALLS   #0, (R0)
                                    54  DD 00041 2$:      PUSHL   R4                      ; 0701
                    65              01  FB 00043         CALLS   #1, SYS$GET
                    53              50  D0 00046         MOVL    R0, STATUS
            000182AA    8F          53  D1 00049         CMPL    STATUS, #98986          ; 0702
                                    23  12 0004C         BNEQ    4$
                CF  52              02  F3 0004E         AOBLEQ  #2, I, 1$               ; 0697
                                    1D  11 00056         BRB     4$
            000182B2    8F          53  D1 00058 3$:      CMPL    STATUS, #98994          ; 0705
                                    14  12 0005F         BNEQ    4$
                28  A4      FA24    C4  9E 00061         MOVAB   UAFBUF, UAFRAB+40       ; 0707
                22  A4      0584    8F  B0 00067         MOVW    #1412, UAFRAB+34        ; 0708
                        50 00010001 8F  D0 0006D         MOVL    #65537, R0              ; 0709
                                        04 00074         RET
                    13              53  E8 00075 4$:      BLBS    STATUS, 5$              ; 0713
                    7E      08      A4  7D 00078         MOVQ    UAFRAB+8, -(SP)         ; 0716
                                    7E  D4 0007C         CLRL    -(SP)                   ; 0715
                        00000000G   8F  DD 0007E         PUSHL   #SET$_UAFERR
                00000000G   00          04  FB 00084         CALLS   #4, LIB$SIGNAL
                    50              53  D0 0008B 5$:      MOVL    STATUS, R0              ; 0719
                                        04 0008E         RET                             ; 0720

; Routine Size:  143 bytes,    Routine Base:  $CODE$ + 0485

;  730          0721  1
```

```
;  732          0722  1 END
;  733          0723  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                                PSECT SUMMARY
;
;      Name                     Bytes                          Attributes
;
;  $OWN$                         1688  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                         364  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        1300  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       108         0        1000        00:01.8


;                                      COMMAND QUALIFIERS

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

; Size:          1300 code + 2052 data bytes
; Run Time:         00:25.5
; Elapsed Time:     01:26.0
; Lines/CPU Min:     1702
; Lexemes/CPU-Min:  27230
; Memory Used:   165 pages
; Compilation Complete
```

SETPROCES
LIS

SETSHOBRO
LIS

SETVOLUME
LIS

SETPWD
LIS

SETTERM
LIS

SETQUEUE
LIS

SETTIME
LIS