F 9

```
UUU         UUU     AAAAAAAAA     FFFFFFFFFFFFFFF
UUU         UUU     AAAAAAAAA     FFFFFFFFFFFFFFF
UUU         UUU     AAAAAAAAA     FFFFFFFFFFFFFFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFFFFFFFFFFF
UUU         UUU  AAA         AAA  FFFFFFFFFFFF
UUU         UUU  AAA         AAA  FFFFFFFFFFFF
UUU         UUU  AAAAAAAAAAAAAAA  FFF
UUU         UUU  AAAAAAAAAAAAAAA  FFF
UUU         UUU  AAAAAAAAAAAAAAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUU         UUU  AAA         AAA  FFF
UUUUUUUUUUUUUUU  AAA         AAA  FFF
UUUUUUUUUUUUUUU  AAA         AAA  FFF
UUUUUUUUUUUUUUU  AAA         AAA  FFF
```

```
_$2


Val
---
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
000
```

```
**FILE**ID**UAFPARSE                                G 9

UU      UU    AAAAAA    FFFFFFFFFF  PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
UU      UU    AAAAAA    FFFFFFFFFF  PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
UU      UU  AA      AA  FF          PP      PP  AA      AA  RR      RR  SS          EE
UU      UU  AA      AA  FF          PP      PP  AA      AA  RR      RR  SS          EE
UU      UU  AA      AA  FF          PP      PP  AA      AA  RR      RR  SS          EE
UU      UU  AA      AA  FF          PP      PP  AA      AA  RR      RR  SS          EE
UU      UU  AA      AA  FFFFFFFF    PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
UU      UU  AA      AA  FFFFFFFF    PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
UU      UU  AAAAAAAAAA  FF          PP          AAAAAAAAAA  RR  RR              SS  EE
UU      UU  AAAAAAAAAA  FF          PP          AAAAAAAAAA  RR  RR              SS  EE
UU      UU  AA      AA  FF          PP          AA      AA  RR    RR            SS  EE
UU      UU  AA      AA  FF          PP          AA      AA  RR    RR            SS  EE              ....
UUUUUUUUUU  AA      AA  FF          PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE      ....
UUUUUUUUUU  AA      AA  FF          PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE      ....

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

```
0 module uafparse (
0                 language (bliss32),
0                 ident = 'V04-000',
0                 addressing_mode(external=general, nonexternal=general)) =
1 begin
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:       System Management Utility Program
1 !
1 ! ABSTRACT:       Parsing subroutines for the AUTHORIZE utility
1 !
1 ! ENVIRONMENT:
1 !
1 ! AUTHOR:         Henry M. Levy   , CREATION DATE: 1-June-1977
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-022 JRL0035         John R. Lawson, Jr.     07-Aug-1984 15:49
1 !               Fix PWD_EXPIRED login flag to be set with passwords.
1 !
1 !       V03-021 JRL0022         John R. Lawson, Jr.     09-Jul-1984 13:40
1 !               Supply missing brackets (if necessary) on /DIRECTORY.
1 !
1 !       V03-020 JRL0011         John R. Lawson, Jr.     02-Jul-1984 14:40
1 !               Add literal WORD_UNSIGNED_LENGTH (=17) for unsigned
1 !               words in WS info.
1 !
1 !       V03-019 JRL0012         John R. Lawson, Jr.     21-Jun-1984 15:56
1 !               Add /GENERATE qualifier to generate random passwords.
1 !
1 !       V03-018 JRL0008         John R. Lawson, Jr.     20-Jun-1984 14:19
1 !               Add /PWDEXPIRED qualifier to allow pre-expired passwords.
1 !
1 !       V03-017 JRL0007         John R. Lawson, Jr.     20-Jun-1984 14:18
```

```
1 !             Add /PWDLIFETIME=NONE qualifier.
1 !
1 !     V03-016 JRL0002           John R. Lawson, Jr.     15-Jun-1984 12:54
1 !             Change all messages to calls to LIB$SIGNAL, place all
1 !             messages in a .MSG file
1 !
1 !     V03-015 LY0475            Larry Yetto             9-APR-1984 08:34
1 !             If /NOEXPIRATION or /NOPWDLIFE is specified then
1 !             zero the field in the UAF record.
1 !
1 !     V03-014 LY0467            Larry Yetto             22-MAR-1984 13:53
1 !             Add support for the rights data base functions
1 !
1 !     V03-013 ACG0397           Andrew C. Goldstein,    6-Feb-1984  16:25
1 !             Add DISREPORT flag to UAF; fix bugs in storing secondary
1 !             password
1 !
1 !     V03-012 ACG0388           Andrew C. Goldstein,    12-Jan-1984  18:52
1 !             Add command input to handle new UAF features;
1 !             general code cleanup
1 !
1 !     V03-011 ACG0385           Andrew C. Goldstein,    6-Jan-1984  14:18
1 !             V4 UAF format change
1 !
1 !     V03-010 TMK0001           Todd M. Katz            11-Oct-1983
1 !             Add JTQUOTA (job-wide logical name table creation quota)
1 !             qualifier.
1 !
1 !     V03-009 LMP0153           L. Mark Pilant,         13-Sep-1983  11:41
1 !             Add minimal support for alphanumeric UICs.
1 !
1 !     008     JWT0105           Jim Teague      04-Apr-1983
1 !             Finish up long filename stuff.
1 !
1 !     007     JWT0104           Jim Teague      29-Mar-1983
1 !             Add CLITABLES qualifier.
1 !
1 !     006     WMC0001           Wayne Cardoza   15-Mar-1983
1 !             Add MAXDETACH qualifier
1 !
1 !     005     JWT0076           Jim Teague      13-Dec-1982
1 !             Fix bug in parsing UIC specs.
1 !
1 !     004     JWT0046           Jim Teague      30-Jul-1982
1 !             Check null password string earlier in GETPASSWORD
1 !
1 !     003     JWT0030           Jim Teague      03-May-1982
1 !             Accomodate ufd directory specs
1 !
1 !     002     JWT0029           Jim Teague      20-Apr-1982
1 !             Completely disable /maxacctjobs
1 !
1 !     001     JWT0023           Jim Teague      17-Mar-1982
1 !             Enforce default directory syntax more strictly by
1 !             cranking the spec through tparse.
1 !
1 !--
```

```
!
! Require files:
!
require
        'lib$:uafreq';
!
! TABLE OF CONTENTS:
!

forward routine

        update_record,                          ! modify all specified fields
        getaccess,                              ! get access flag bits
        check_primary,                          ! verify "primary" keyword
        check_secondary,                        ! verify "secondary" keyword
        set_range,                              ! set hourly restriction bits
        getcputim,                              ! get cpu time quota
        getdevice,                              ! get default device name
        getdirectory,                           ! get default directory
        checkvalue,                             ! check UFD group and member value
        checklength,                            ! check directory name length
        getflags,                               ! get flag bits
        getpflags,                              ! get old per day login flags
        getrestrict,                            ! get old hourly restriction values
        getprimedays,                           ! get primary day list
        getpriv,                                ! get process privileges
        getuic,                                 ! get user identification code
        parse_uic,                              ! obtain UIC group and member
        parse_wild,                             ! parse wildcarded user specification
        parse_wild_uic,                         ! parse wildcarded UIC
        getstring,                              ! get string from input
        getval,                                 ! get numeric value from user
        cvtnum,                                 ! convert decimal ascii to binary
        UAF$GENERATE,                           ! generate necessary passwords
        GETPASSWORD,                            ! update password fields
        AUTHORIZE_GENERATE;                     ! Interface to PL/I


external routine

        GENERATE_PASSWORDS,
        SYS$ASSIGN,
        SYS$DASSGN,
        SYS$QIOW,
        LIB$GET_VM,
        STR$UPCASE,
        STR$UPCASE,
        lib$tparse      : addressing_mode (general),
        lib$lookup_key  : addressing_mode (general),
        cli$dcl_parse,
        cli$present,
        cli$get_value,
        faoout,                                 ! output formatted message
        lgi$hpwd,                               ! hash password routine
        lib$cvt_dtime,                          ! convert ASCII to system delta time
        lib$cvt_time,                           ! convert ASCII to system abs time
        prv$setpriv;                            ! set privilege bits based on
```

```
;  173           0267  1                                                 ! privilege names
;  174           0268  1
;  175           0269  1 !
;  176           0270  1 ! INCLUDE FILES:
;  177           0271  1 !
;  178           0272  1
;  179           0273  1 library 'SYS$LIBRARY:TPAMAC';
;  180           0274  1 library 'SYS$LIBRARY:LIB';
;  181           0275  1
;  182           0276  1 !
;  183           0277  1 ! MACROS:
;  184           0278  1 !
;  185           0279  1
;  186           0280  1 macro
;  187         M 0281  1     cstring[] = (uplit byte (%charcount (%string (%remaining)),
;  188           0282  1             %string (%remaining)))%,
;  189           0283  1
;  190           0284  1 !
;  191           0285  1 ! Macro to create string descriptor for command parameters and
;  192           0286  1 !      qualifiers
;  193           0287  1 !
;  194         M 0288  1     sd[a] =
;  195           0289  1         bind %name ('SD_', a) = $descriptor (a)%;
;  196           0290  1
;  197           0291  1
;  198         P 0292  1      sd (
;  199         P 0293  1         'token1',         'token2',         'account',         'astlm',
;  200         P 0294  1         'biolm',          'bytlm',          'cli',             'cputime',
;  201         P 0295  1         'device',         'diolm',          'directory',       'enqlm',
;  202         P 0296  1         'fillm',          'flags',          'lgicmd',          'maxjobs',
;  203         P 0297  1         'owner',          'password',       'pbytlm',          'pflags',
;  204         P 0298  1         'pgflquota',      'p_restrict',     'prclm',           'priority',
;  205         P 0299  1         'privileges',     'sflags',         's_restrict',      'shrfillm',
;  206         P 0300  1         'tqelm',          'uic',            'wsdefault',       'wsextent',
;  207         P 0301  1         'wsquota',        'display',        'primedays',       'maxacctjobs',
;  208         P 0302  1         'maxdetach',      'clitables',      'jtquota',         'access',
;  209         P 0303  1         'batch',          'defprivileges','dialup',           'expiration',
;  210         P 0304  1         'interactive',    'local',          'network',         'pwdlifetime',
;  211         P 0305  1         'pwdminimum',     'quepriority',    'remote'
;  212           0306  1 );
;  213           0307  1
;  214           0308  1 !
;  215           0309  1 ! The definition table for flag bits for UAF$B_FLAGS.
;  216           0310  1 !
;  217           0311  1 literal
;  218           0312  1         num_flags       = 13,           ! number of FLAG_TABLE entries
;  219           0313  1         num_opflags     = 2,            ! number of OPFLAG_TABLE entries
;  220           0314  1         num_pdflags     = 7;            ! number of entries in prime days table
;  221           0315  1
;  222           0316  1 own
;  223           0317  1     flag_table : vector [2 * num_flags + 1, long] initial (
;  224           0318  1         2 * num_flags,
;  225           0319  1         uplit (%ascic 'DISCTLY'),       $bitposition (uaf$v_disctly),
;  226           0320  1         uplit (%ascic 'DEFCLI'),        $bitposition (uaf$v_defcli),
;  227           0321  1         uplit (%ascic 'LOCKPWD'),       $bitposition (uaf$v_lockpwd),
;  228           0322  1         uplit (%ascic 'CAPTIVE'),       $bitposition (uaf$v_captive),
;  229           0323  1         uplit (%ascic 'DISUSER'),       $bitposition (uaf$v_disacnt),
```

```
;  230        0324  1          uplit (%ascic 'DISWELCOME'),     $bitposition (uaf$v_diswelcom),
;  231        0325  1          uplit (%ascic 'DISNEWMAIL'),     $bitposition (uaf$v_dismail),
;  232        0326  1          uplit (%ascic 'DISMAIL'),        $bitposition (uaf$v_nomail),
;  233        0327  1          uplit (%ascic 'GENPWD'),         $bitposition (uaf$v_genpwd),
;  234        0328  1          uplit (%ascic 'PWD_EXPIRED'),    $bitposition (uaf$v_pwd_expired),
;  235        0329  1          uplit (%ascic 'PWD2_EXPIRED'),   $bitposition (uaf$v_pwd2_expired),
;  236        0330  1          uplit (%ascic 'AUDIT'),          $bitposition (uaf$v_audit),
;  237        0331  1          uplit (%ascic 'DISREPORT'),      $bitposition (uaf$v_disreport)
;  238        0332  1  ),
;  239        0333  1  !
;  240        0334  1  ! The definition table for flag bits for UAF$B_PDAYFLAGS and  UAF$B_SDAYFLAGS.
;  241        0335  1  !
;  242        0336  1      opflag_table : vector [2 * num_opflags + 1, long] initial (
;  243        0337  1          2 * num_opflags,
;  244        0338  1          uplit (%ascic 'DISDIALUP'),      uaf$k_disdialup,
;  245        0339  1          uplit (%ascic 'DISNETWORK'),     uaf$k_disnetwork        ! Disneyworld ??
;  246        0340  1  ),
;  247        0341  1  !
;  248        0342  1  ! The definition table for flag bits for UAF$B_PRIMEDAYS.
;  249        0343  1  !
;  250        0344  1      pdflag_table : vector [2 * num_pdflags + 1, long] initial (
;  251        0345  1          2 * num_pdflags,
;  252        0346  1          uplit (%ascic 'MONDAY'),         $bitposition (uaf$v_monday),
;  253        0347  1          uplit (%ascic 'TUESDAY'),        $bitposition (uaf$v_tuesday),
;  254        0348  1          uplit (%ascic 'WEDNESDAY'),      $bitposition (uaf$v_wednesday),
;  255        0349  1          uplit (%ascic 'THURSDAY'),       $bitposition (uaf$v_thursday),
;  256        0350  1          uplit (%ascic 'FRIDAY'),         $bitposition (uaf$v_friday),
;  257        0351  1          uplit (%ascic 'SATURDAY'),       $bitposition (uaf$v_saturday),
;  258        0352  1          uplit (%ascic 'SUNDAY'),         $bitposition (uaf$v_sunday)
;  259        0353  1  );
;  260        0354  1
;  261        0355  1
;  262        0356  1  !
;  263        0357  1  ! EQUATED SYMBOLS:
;  264        0358  1  !
;  265        0359  1
;  266        0360  1  literal
;  267        0361  1          false           = 0,            ! logical false
;  268        0362  1          true            = 1,            ! logical true
;  269        0363  1
;  270        0364  1          network_access  = 1^0,          ! mask representing /NETWORK_ACCESS
;  271        0365  1          batch_access    = 1^1,          ! mask representing /BATCH_ACCESS
;  272        0366  1          local_access    = 1^2,          ! mask representing /LOCAL_ACCESS
;  273        0367  1          dialup_access   = 1^3,          ! mask representing /DIALUP_ACCESS
;  274        0368  1          remote_access   = 1^4,          ! mask representing /REMOTE_ACCESS
;  275        0369  1          interactive_access =            ! mask representing /INTERACTIVE_ACCESS
;  276        0370  1                          local_access or dialup_access or remote_access,
;  277        0371  1          access_access   =               ! mask representing /ACCESS
;  278        0372  1                          interactive_access or network_access or batch_access,
;  279        0373  1
;  280        0374  1          byte_length     = 8,            ! bits per byte
;  281        0375  1          word_length     = 16,           ! bits per word
;  282        0376  1          word_unsigned_length
;  283        0377  1                          = 17,           ! for WS info
;  284        0378  1          long_length     = 32,           ! bits per longword
;  285        0379  1
;  286        0380  1
```

```
;  287       0381  1          blank             = ' ',               !  delimiters
;  288       0382  1          colon             = ':',
;  289       0383  1          comma             = ',',
;  290       0384  1          cr                = 13,
;  291       0385  1          equals            = '=',
;  292       0386  1          lf                = 10,
;  293       0387  1          lparen            = '(',
;  294       0388  1          rparen            = ')',
;  295       0389  1          slash             = '/',
;  296       0390  1          tab               = '     ',
;  297       0391  1          zero              = 0,
;  298       0392  1
;  299       0393  1          counted_string    = 1,                 ! counted string type indicator
;  300       0394  1          filled_string     = 2;                 ! filled string type indicator
;  301       0395  1
;  302       0396  1  own
;  303       0397  1          status,
;  304       0398  1          cli_status,
;  305       0399  1
;  306       0400  1          tparse_block      : block [tpa$c_length0, byte]
;  307       0401  1                              preset ([tpa$l_count]   = tpa$k_count0,
;  308       0402  1                                      [tpa$l_options] = tpa$m_abbrev),
;  309       0403  1          converted_uic,
;  310       0404  1          address,
;  311       0405  1          left_bit,
;  312       0406  1          right_bit,
;  313       0407  1          primary_access    : bitvector [32],
;  314       0408  1          secondary_access: bitvector [32],
;  315       0409  1          no_flag           : initial (false),
;  316       0410  1          primary           : initial (false),
;  317       0411  1          secondary         : initial (false);
;  318       0412  1
;  319       0413  1  !
;  320       0414  1  ! ASCII numbers from '0' through hex 'F'
;  321       0415  1  !
;  322       0416  1
;  323       0417  1  bind
;  324       0418  1          numbers           = uplit byte ('0123456789ABCDEF'),
;  325       0419  1
;  326       0420  1  !
;  327       0421  1  ! 'NO' prefix descriptor for negating privileges
;  328       0422  1  !
;  329       0423  1          no_dsc            = $descriptor ('NO') : vector;
;  330       0424  1
;  331       0425  1  !
;  332       0426  1  !       External messages
;  333       0427  1  !
;  334       0428  1
;  335       0429  1  external routine
;  336       0430  1
;  337       0431  1     LIB$SIGNAL: addressing_mode(general);
;  338       0432  1
;  339       0433  1  external literal
;  340       0434  1
;  341       0435  1     UAF$_BADVALUE,        UAF$_EXTRAPARM,         UAF$_INVDEV,
;  342       0436  1     UAF$_INVSTR,          UAF$_INVUSERSPEC,       UAF$_KEYNOTFND,
;  343       0437  1     UAF$_NAMETOOBIG,      UAF$_NOARG,             UAF$_NOQUOTE,
```

```
;  344         0438  1      UAF$_NOUSERSPEC,      UAF$_PRVNOTFND,           UAF$_PRVNOTUNQ,
;  345         0439  1      UAF$_PWDSYNTAX,       UAF$_PWDNCH,              UAF$_PWDNOL,
;  346         0440  1      UAF$_UICERR,          UAF$_VALTOOBIG,           UAF$_ZCONFLICT;
;  347         0441  1
;  348         0442  1
;  349         0443  1  !
;  350         0444  1  ! EXTERNAL REFERENCES:
;  351         0445  1  !
;  352         0446  1
;  353         0447  1
;  354         0448  1  external literal
;  355         0449  1
;  356         0450  1          encrypt,                            ! encryption algorithm to use
;  357         0451  1
;  358         0452  1          lib$_ambkey,                        ! status for ambiguous keyword
;  359         0453  1          lib$_unrkey,                        ! status for unrecognized keyword
;  360         0454  1
;  361         0455  1          cli$_abkeyw,                        ! message symbol for "ambiguous keyword"
;  362         0456  1          cli$_ivkeyw,                        ! message symbol for "unrecognized keyword"
;  363         0457  1          cli$_negated,                       ! qualifier explicitly negated
;  364         0458  1          CLI$_ABSENT,                        ! Qualifier not given
;  365         0459  1          prv$_notunq,
;  366         0460  1          prv$_invnam;
;  367         0461  1
;  368         0462  1  external                                    ! Global storage from UAFMAIN
;  369         0463  1          uaf$gl_ctlmsk    : block,
;  370         0464  1          authorize_commands : addressing_mode ( general ),
;  371         0465  1          cmdlindsc        : block [,byte],
;  372         0466  1          call_count,
;  373         0467  1          symbol_str,                         ! Symbolic character set
;  374         0468  1          match_tokenlen,
;  375         0469  1          match_token,
;  376         0470  1          tokendsc         : block [,byte],
;  377         0471  1          tokenlen         : word,            ! Bound to TOKENDSC[DSC$W_LENGTH]
;  378         0472  1          tokenptr,                           ! Bound to TOKENDSC[DSC$A_POINTER]
;  379         0473  1          uic_flag,                           ! Wildcard parsing flags
;  380         0474  1          grp_wild,
;  381         0475  1          mem_wild,
;  382         0476  1          str_wild,
;  383         0477  1
;  384         0478  1          time_buf         : block [,byte], ! Buffer to receive current system time
;  385         0479  1          pwd_flag,                         ! User did not supply password flag
;  386         0480  1          recbuf           : block [,byte]; ! The current UAF record
;  387         0481  1
;  388         0482  1  !
;  389         0483  1  ! TPARSE state table to parse /<access> qualifiers
;  390         0484  1  !
;  391         0485  1
;  392         0486  1  $init_state (access_states, access_keys);
;  393         0487  1
;  394       P 0488  1  $state   (,
;  395       P 0489  1           ('PRIMARY', terminal, check_primary, true, primary),
;  396       P 0490  1           ('SECONDARY', terminal, check_secondary, true, secondary),
;  397       P 0491  1           (tpa$_decimal,,,, right_bit)
;  398         0492  1          );
;  399         0493  1
;  400       P 0494  1  $state   (,
```

```
;  401        P 0495  1               ('-'),
;  402        P 0496  1               (tpa$_eos, tpa$_exit, set_range)
;  403          0497  1               );
;  404          0498  1
;  405        P 0499  1   $state      (,
;  406        P 0500  1               (tpa$_decimal,, set_range)
;  407          0501  1               );
;  408          0502  1
;  409        P 0503  1   $state      (terminal,
;  410        P 0504  1               (tpa$_eos, tpa$_exit)
;  411          0505  1               );
;  412          0506  1
;  413          0507  1   !
;  414          0508  1   ! TPARSE state table for old /P_RESTRICT and /S_RESTRICT qualifiers.
;  415          0509  1   !
;  416          0510  1
;  417          0511  1   $init_state (restrict_states, restrict_keys);
;  418          0512  1
;  419        P 0513  1   $state      (,
;  420        P 0514  1               (tpa$_decimal,,,, right_bit)
;  421          0515  1               );
;  422          0516  1
;  423        P 0517  1   $state      (,
;  424        P 0518  1               ('-'),
;  425        P 0519  1               (tpa$_eos, tpa$_exit, set_range)
;  426          0520  1               );
;  427          0521  1
;  428        P 0522  1   $state      (,
;  429        P 0523  1               (tpa$_decimal,, set_range)
;  430          0524  1               );
;  431          0525  1
;  432        P 0526  1   $state      (,
;  433        P 0527  1               (tpa$_eos, tpa$_exit)
;  434          0528  1               );
;  435          0529  1
;  436          0530  1   !
;  437          0531  1   ! TPARSE state table for UIC spec
;  438          0532  1   !
;  439          0533  1
;  440          0534  1   $init_state (uic_states, uic_keys);
;  441          0535  1
;  442        P 0536  1   $state      (,
;  443        P 0537  1               (tpa$_ident, tpa$_exit,,, converted_uic)
;  444          0538  1               );
;  445          0539  1
;  446          0540  1   !
;  447          0541  1   ! TPARSE state table for directory string
;  448          0542  1   !
;  449          0543  1
;  450          0544  1   $init_state (dir_states, dir_keys);
;  451          0545  1
;  452        P 0546  1   $state      (,
;  453        P 0547  1               ( (ufd), end_state),
;  454        P 0548  1               ( (subdir), end_state),
;  455        P 0549  1               ('['),
;  456        P 0550  1               ('<')
;  457          0551  1               );
```

```
;  458          0552   1
;  459        P 0553   1 $state   (,
;  460        P 0554   1          ( (ufd)),
;  461        P 0555   1          ( (subdir))
;  462          0556   1          );
;  463          0557   1
;  464        P 0558   1 $state   (,
;  465        P 0559   1          (']'),
;  466        P 0560   1          ('>')
;  467          0561   1          );
;  468          0562   1
;  469        P 0563   1 $state   (end_state,
;  470        P 0564   1          (tpa$_eos, tpa$_exit)
;  471          0565   1          );
;  472          0566   1
;  473        P 0567   1 $state   (ufd,
;  474        P 0568   1          (tpa$_octal,, checkvalue)
;  475          0569   1          );
;  476          0570   1
;  477        P 0571   1 $state   (,
;  478        P 0572   1          (',')
;  479          0573   1          );
;  480          0574   1
;  481        P 0575   1 $state   (,
;  482        P 0576   1          (tpa$_octal, tpa$_exit, checkvalue)
;  483          0577   1          );
;  484          0578   1
;  485        P 0579   1 $state   (subdir,
;  486        P 0580   1          (tpa$_symbol,, checklength)
;  487          0581   1          );
;  488          0582   1
;  489        P 0583   1 $state   (,
;  490        P 0584   1          ('.',subdir),
;  491        P 0585   1          (tpa$_lambda,tpa$_exit)
;  492          0586   1          );
```

```
;  494          0587  1 %sbttl 'update_record - modify all specified fields'
;  495          0588  1 global routine update_record =
;  496          0589  2 begin
;  497          0590  2
;  498          0591  2 !++
;  499          0592  2 !
;  500          0593  2 ! FUNCTIONAL DESCRIPTION:
;  501          0594  2 !
;  502          0595  2 !       Scan the remainder of the user's input command line for parameters
;  503          0596  2 !       and parameter values.  Pull off each parameter keyword and check
;  504          0597  2 !       that it is valid, then call the associated routine to process
;  505          0598  2 !       the supplied value or string.
;  506          0599  2 !
;  507          0600  2 ! INPUTS:
;  508          0601  2 !
;  509          0602  2 !       none
;  510          0603  2 !
;  511          0604  2 ! IMPLICIT INPUTS:
;  512          0605  2 !
;  513          0606  2 !
;  514          0607  2 ! OUTPUTS:
;  515          0608  2 !
;  516          0609  2 !       none
;  517          0610  2 !
;  518          0611  2 ! IMPLICIT OUTPUTS:
;  519          0612  2 !
;  520          0613  2 !       none
;  521          0614  2 !
;  522          0615  2 ! ROUTINE VALUE:
;  523          0616  2 !
;  524          0617  2 !       true -> all keywords and parameters were specified correctly
;  525          0618  2 !
;  526          0619  2 ! SIDE EFFECTS:
;  527          0620  2 !
;  528          0621  2 !       none
;  529          0622  2 !--
;  530          0623  2 status = false;
;  531          0624  2 primary = false;
;  532          0625  2 secondary = false;
;  533          0626  2 no_flag = false;
;  534          0627  2
;  535          0628  2 !
;  536          0629  2 ! If this is part of a series of calls from WILD_USER, the command
;  537          0630  2 !    line must be reparsed after the first call
;  538          0631  2
;  539          0632  2 if .call_count neq 0
;  540          0633  2 then
;  541          0634  2     cli$dcl_parse (cmdlindsc, authorize_commands);
;  542          0635  2
;  543          0636  2
;  544          0637  2 if CLI$PRESENT(%ascid'GENERATE_PASSWORD')
;  545          0638  2 and (CLI$PRESENT(SD_PASSWORD) neq CLI$_ABSENT) then
;  546          0639  3    begin
;  547          0640  3
;  548          0641  3       LIB$SIGNAL(UAF$_ZCONFLICT);
;  549          0642  3
;  550          0643  3    end
```

```
;  551      0644  2 else if CLI$PRESENT(%ascid'GENERATE_PASSWORD')
;  552      0645  2 or (CLI$PRESENT(SD_PASSWORD) neq CLI$_ABSENT) then
;  553      0646  3     begin
;  554      0647  3
;  555      0648  3         if CLI$PRESENT(%ascid'GENERATE_PASSWORD') then
;  556      0649  3             UAF$GENERATE();
;  557      0650  3         if not GETPASSWORD() then
;  558      0651  3             return FALSE;
;  559      0652  3         RECBUF[UAF$V_PWD_EXPIRED] = FALSE;
;  560      0653  3         STATUS = TRUE;
;  561      0654  3
;  562      0655  2     end;
;  563      0656  2
;  564      0657  2
;  565      0658  2 if .uaf$gl_ctlmsk[uaf$v_rename]
;  566      0659  2 then
;  567      0660  2      return true;
;  568      0661  2
;  569      0662  2 !
;  570      0663  2 ! Check that we're not at the end of the command line. Continue
;  571      0664  2 ! processing parameters until the end is reached.
;  572      0665  2 !
;  573      0666  2
;  574      0667  2 cli_status = cli$present (sd_access);
;  575      0668  2 if .cli_status
;  576      0669  2 or .cli_status eql cli$_negated
;  577      0670  2 then
;  578      0671  3      begin
;  579      0672  3      if not getaccess (sd_access, access_access, .cli_status)
;  580      0673  3      then return false;
;  581      0674  3      status = true;
;  582      0675  2      end;
;  583      0676  2
;  584      0677  2 cli_status = cli$present (sd_interactive);
;  585      0678  2 if .cli_status
;  586      0679  2 or .cli_status eql cli$_negated
;  587      0680  2 then
;  588      0681  3      begin
;  589      0682  3      if not getaccess (sd_interactive, interactive_access, .cli_status)
;  590      0683  3      then return false;
;  591      0684  3      status = true;
;  592      0685  2      end;
;  593      0686  2
;  594      0687  2 cli_status = cli$present (sd_batch);
;  595      0688  2 if .cli_status
;  596      0689  2 or .cli_status eql cli$_negated
;  597      0690  2 then
;  598      0691  3      begin
;  599      0692  3      if not getaccess (sd_batch, batch_access, .cli_status)
;  600      0693  3      then return false;
;  601      0694  3      status = true;
;  602      0695  2      end;
;  603      0696  2
;  604      0697  2 cli_status = cli$present (sd_dialup);
;  605      0698  2 if .cli_status
;  606      0699  2 or .cli_status eql cli$_negated
;  607      0700  2 then
```

```
;  608          0701 3      begin
;  609          0702 3      if not getaccess (sd_dialup, dialup_access, .cli_status)
;  610          0703 3      then return false;
;  611          0704 3      status = true;
;  612          0705 2      end;
;  613          0706 2
;  614          0707 2  cli_status = cli$present (sd_network);
;  615          0708 2  if .cli_status
;  616          0709 2  or .cli_status eql cli$_negated
;  617          0710 2  then
;  618          0711 3      begin
;  619          0712 3      if not getaccess (sd_network, network_access, .cli_status)
;  620          0713 3      then return false;
;  621          0714 3      status = true;
;  622          0715 2      end;
;  623          0716 2
;  624          0717 2  cli_status = cli$present (sd_local);
;  625          0718 2  if .cli_status
;  626          0719 2  or .cli_status eql cli$_negated
;  627          0720 2  then
;  628          0721 3      begin
;  629          0722 3      if not getaccess (sd_local, local_access, .cli_status)
;  630          0723 3      then return false;
;  631          0724 3      status = true;
;  632          0725 2      end;
;  633          0726 2
;  634          0727 2  cli_status = cli$present (sd_remote);
;  635          0728 2  if .cli_status
;  636          0729 2  or .cli_status eql cli$_negated
;  637          0730 2  then
;  638          0731 3      begin
;  639          0732 3      if not getaccess (sd_remote, remote_access, .cli_status)
;  640          0733 3      then return false;
;  641          0734 3      status = true;
;  642          0735 2      end;
;  643          0736 2
;  644          0737 2  if cli$present (sd_account)
;  645          0738 2  then
;  646          0739 3      begin
;  647          0740 3      cli$get_value (sd_account, tokendsc);
;  648          0741 3  !***     if not getstring (recbuf[uaf$t_account], uaf$s_account, filled_string)
;  649          0742 3      if not getstring (recbuf[uaf$t_account], 8, filled_string)
;  650          0743 3      then return false;
;  651          0744 3      status = true;
;  652          0745 2      end;
;  653          0746 2
;  654          0747 2  if cli$present (sd_astlm)
;  655          0748 2  then
;  656          0749 3      begin
;  657          0750 3      cli$get_value (sd_astlm, tokendsc);
;  658          0751 3      if not getval (recbuf[uaf$w_astlm], word_length)
;  659          0752 3      then return false;
;  660          0753 3      status = true;
;  661          0754 2      end;
;  662          0755 2
;  663          0756 2  if cli$present (sd_biolm)
;  664          0757 2  then
```

```
;  665       0758  3         begin
;  666       0759  3         cli$get_value (sd_biolm, tokendsc);
;  667       0760  3         if not getval (recbuf[uaf$w_biolm], word_length)
;  668       0761  3         then return false;
;  669       0762  3         status = true;
;  670       0763  2         end;
;  671       0764  2
;  672       0765  2     if cli$present (sd_bytlm)
;  673       0766  2     then
;  674       0767  3         begin
;  675       0768  3         cli$get_value (sd_bytlm, tokendsc);
;  676       0769  3         if not getval (recbuf[uaf$l_bytlm], long_length)
;  677       0770  3         then return false;
;  678       0771  3         status = true;
;  679       0772  2         end;
;  680       0773  2
;  681       0774  2     if cli$present (sd_cli)
;  682       0775  2     then
;  683       0776  3         begin
;  684       0777  3         cli$get_value (sd_cli, tokendsc);
;  685       0778  3         if not getstring (recbuf[uaf$t_defcli], uaf$s_defcli, counted_string)
;  686       0779  3         then return false;
;  687       0780  3         uaf$gl_ctlmsk[uaf$v_clitab_pres] =
;  688       0781  3             (.vector [recbuf[uaf$t_clitables], 0; ,byte] neq 0);
;  689       0782  3         uaf$gl_ctlmsk[uaf$v_cli] = true;
;  690       0783  3         status = true;
;  691       0784  2         end;
;  692       0785  2
;  693       0786  2     if cli$present (sd_clitables)
;  694       0787  2     then
;  695       0788  3         begin
;  696       0789  3         cli$get_value (sd_clitables, tokendsc);
;  697       0790  3         if not getstring (recbuf[uaf$t_clitables], uaf$s_clitables, counted_string)
;  698       0791  3         then return false;
;  699       0792  3         uaf$gl_ctlmsk[uaf$v_clitables] = true;
;  700       0793  3         status = true;
;  701       0794  2         end;
;  702       0795  2
;  703       0796  2     if cli$present (sd_cputime)
;  704       0797  2     then
;  705       0798  3         begin
;  706       0799  3         cli$get_value (sd_cputime, tokendsc);
;  707       0800  3         if not getcputim ()
;  708       0801  3         then return false;
;  709       0802  3         status = true;
;  710       0803  2         end;
;  711       0804  2
;  712       0805  2     if cli$present (sd_defprivileges)
;  713       0806  2     then
;  714       0807  3         begin
;  715       0808  3         if not getpriv (sd_defprivileges, recbuf[uaf$q_def_priv])
;  716       0809  3         then return false;
;  717       0810  3         status = true;
;  718       0811  2         end;
;  719       0812  2
;  720       0813  2     if cli$present (sd_device)
;  721       0814  2     then
```

```
;  722        0815  3          begin
;  723        0816  3          cli$get_value (sd_device, tokendsc);
;  724        0817  3          if not getdevice ()
;  725        0818  3          then return false;
;  726        0819  3          status = true;
;  727        0820  2          end;
;  728        0821  2
;  729        0822  2     if cli$present (sd_diolm)
;  730        0823  2     then
;  731        0824  3          begin
;  732        0825  3          cli$get_value (sd_diolm, tokendsc);
;  733        0826  3          if not getval (recbuf[uaf$w_diolm], word_length)
;  734        0827  3          then return false;
;  735        0828  3          status = true;
;  736        0829  2          end;
;  737        0830  2
;  738        0831  2     if cli$present (sd_directory)
;  739        0832  2     then
;  740        0833  3          begin
;  741        0834  3          cli$get_value (sd_directory, tokendsc);
;  742        0835  3          if not getdirectory ()
;  743        0836  3          then return false;
;  744        0837  3          status = true;
;  745        0838  2          end;
;  746        0839  2
;  747        0840  2     if cli$present (sd_enqlm)
;  748        0841  2     then
;  749        0842  3          begin
;  750        0843  3          cli$get_value (sd_enqlm, tokendsc);
;  751        0844  3          if not getval (recbuf[uaf$w_enqlm], word_length)
;  752        0845  3          then return false;
;  753        0846  3          status = true;
;  754        0847  2          end;
;  755        0848  2
;  756        0849  2     cli_status = cli$present (sd_expiration) ;
;  757        0850  2     if .cli_status eql cli$_negated
;  758        0851  2     then
;  759        0852  3          begin
;  760        0853  3          ch$fill ( 0, uaf$s_expiration, recbuf[uaf$q_expiration] ) ;
;  761        0854  3          status = true ;
;  762        0855  3          end
;  763        0856  2     else
;  764        0857  2          if .cli_status
;  765        0858  2          then
;  766        0859  3               begin
;  767        0860  3               cli$get_value (sd_expiration, tokendsc);
;  768        0861  4               if not (status = lib$cvt_time (tokendsc, recbuf[uaf$q_expiration]))
;  769        0862  3               then
;  770        0863  4                    begin
;  771        0864  4                    signal (.status);
;  772        0865  4                    return false;
;  773        0866  3                    end;
;  774        0867  3               status = true;
;  775        0868  2               end;
;  776        0869  2
;  777        0870  2     if cli$present (sd_fillm)
;  778        0871  2     then
```

```
;   779         0872  3             begin
;   780         0873  3             cli$get_value (sd_fillm, tokendsc);
;   781         0874  3             if not getval (recbuf[uaf$w_fillm], word_length)
;   782         0875  3             then return false;
;   783         0876  3             status = true;
;   784         0877  2             end;
;   785         0878  2
;   786         0879  2         if cli$present (sd_flags)
;   787         0880  2         then
;   788         0881  3             begin
;   789         0882  3             if not getflags ()
;   790         0883  3             then return false;
;   791         0884  3             status = true;
;   792         0885  2             end;
;   793         0886  2
;   794         0887  2         if cli$present (sd_jtquota)
;   795         0888  2         then
;   796         0889  3             begin
;   797         0890  3             cli$get_value (sd_jtquota, tokendsc);
;   798         0891  3             if not getval (recbuf[uaf$l_jtquota], long_length)
;   799         0892  3             then return false;
;   800         0893  3             status = true;
;   801         0894  2             end;
;   802         0895  2
;   803         0896  2         if cli$present (sd_lgicmd)
;   804         0897  2         then
;   805         0898  3             begin
;   806         0899  3             cli$get_value (sd_lgicmd, tokendsc);
;   807         0900  3             if not getstring (recbuf[uaf$t_lgicmd], uaf$s_lgicmd, counted_string)
;   808         0901  3             then return false;
;   809         0902  3             status = true;
;   810         0903  2             end;
;   811         0904  2
;   812         0905  2         if cli$present (sd_maxjobs)
;   813         0906  2         then
;   814         0907  3             begin
;   815         0908  3             cli$get_value (sd_maxjobs, tokendsc);
;   816         0909  3             if not getval (recbuf[uaf$w_maxjobs], word_length)
;   817         0910  3             then return false;
;   818         0911  3             status = true;
;   819         0912  2             end;
;   820         0913  2
;   821         0914  2         if cli$present (sd_maxacctjobs)
;   822         0915  2         then
;   823         0916  3             begin
;   824         0917  3             cli$get_value (sd_maxacctjobs, tokendsc);
;   825         0918  3             if not getval (recbuf[uaf$w_maxacctjobs], word_length)
;   826         0919  3             then return false;
;   827         0920  3             status = true;
;   828         0921  2             end;
;   829         0922  2
;   830         0923  2         if cli$present (sd_maxdetach)
;   831         0924  2         then
;   832         0925  3             begin
;   833         0926  3             cli$get_value (sd_maxdetach, tokendsc);
;   834         0927  3             if not getval (recbuf[uaf$w_maxdetach], word_length)
;   835         0928  3             then return false;
```

```
;  836          0929 3          status = true;
;  837          0930 2          end;
;  838          0931 2
;  839          0932 2  if cli$present (sd_owner)
;  840          0933 2  then
;  841          0934 3          begin
;  842          0935 3          cli$get_value (sd_owner, tokendsc);
;  843          0936 3          if not getstring (recbuf[uaf$t_owner], uaf$s_owner, counted_string)
;  844          0937 3          then return false;
;  845          0938 3          status = true;
;  846          0939 2          end;
;  847          0940 2
;  848          0941 2  if cli$present (sd_pbytlm)
;  849          0942 2  then
;  850          0943 3          begin
;  851          0944 3          cli$get_value (sd_pbytlm, tokendsc);
;  852          0945 3          if not getval (recbuf[uaf$l_pbytlm], long_length)
;  853          0946 3          then return false;
;  854          0947 3          status = true;
;  855          0948 2          end;
;  856          0949 2
;  857          0950 2  if cli$present (sd_pgflquota)
;  858          0951 2  then
;  859          0952 3          begin
;  860          0953 3          cli$get_value (sd_pgflquota, tokendsc);
;  861          0954 3          if not getval (recbuf[uaf$l_pgflquota], 3 * byte_length)
;  862          0955 3          then return false;
;  863          0956 3          status = true;
;  864          0957 2          end;
;  865          0958 2
;  866          0959 2  if cli$present (sd_p_restrict)
;  867          0960 2  then
;  868          0961 3          begin
;  869          0962 3          if not getrestrict (sd_p_restrict)
;  870          0963 3          then return false;
;  871          0964 3          status = true;
;  872          0965 2          end;
;  873          0966 2
;  874          0967 2  if cli$present (sd_s_restrict)
;  875          0968 2  then
;  876          0969 3          begin
;  877          0970 3          secondary = true;
;  878          0971 3          if not getrestrict (sd_s_restrict)
;  879          0972 3          then return false;
;  880          0973 3          status = true;
;  881          0974 2          end;
;  882          0975 2
;  883          0976 2  if cli$present (sd_pflags)
;  884          0977 2  then
;  885          0978 3          begin
;  886          0979 3          primary = true;
;  887          0980 3          if not getpflags (sd_pflags)
;  888          0981 3          then return false;
;  889          0982 3          status = true;
;  890          0983 2          end;
;  891          0984 2
;  892          0985 2  if cli$present (sd_sflags)
```

```
2 then
3     begin
3     if not getpflags (sd_sflags)
3     then return false;
3     status = true;
2     end;
2
2 if cli$present (sd_prclm)
2 then
3     begin
3     cli$get_value (sd_prclm, tokendsc);
3     if not getval (recbuf[uaf$w_prccnt], word_length)
3     then return false;
3     status = true;
2     end;
2
2 if cli$present (sd_primedays)
2 then
3     begin
3     if not getprimedays ()
3     then return false;
3     status = true;
2     end;
2
2 if cli$present (sd_priority)
2 then
3     begin
3     cli$get_value (sd_priority, tokendsc);
3     if not getval (recbuf[uaf$b_pri], byte_length)
3     then return false;
3     status = true;
2     end;
2
2 if cli$present (sd_privileges)
2 then
3     begin
3     if not getpriv (sd_privileges, recbuf[uaf$q_priv])
3     then return false;
3     status = true;
2     end;
2
2 CLI_STATUS = CLI$PRESENT(%ascid'PWDEXPIRED');
2 if .CLI_STATUS eql CLI$_NEGATED then
3    begin
3       local TIME: long;
3       TIME = RECBUF[UAF$Q_PWD_DATE];
3       .TIME = 0; (.TIME+%upval) = 0;
3       RECBUF[UAF$V_PWD_EXPIRED] = FALSE;
3       STATUS = TRUE;
3    end
2 else if .CLI_STATUS then
3    begin
3       local TIME: long;
3       TIME = RECBUF[UAF$Q_PWD_DATE];
3       .TIME = -1; (.TIME+%upval) = -1;
3       RECBUF[UAF$V_PWD_EXPIRED] = FALSE;
3       STATUS = TRUE;
```

```
;   950        1043  2     end;
;   951        1044  2
;   952        1045  2  cli_status = cli$present (sd_pwdlifetime) ;
;   953        1046  2  if .cli_status eql cli$_negated
;   954        1047  2  then
;   955        1048  3      begin
;   956        1049  3      ch$fill ( 0, uaf$s_pwd_lifetime, recbuf[uaf$q_pwd_lifetime] ) ;
;   957        1050  3      RECBUF[UAF$V_PWD_EXPIRED] = FALSE;
;   958        1051  3      status = true ;
;   959        1052  3      end
;   960        1053  2  else
;   961        1054  2      if .cli_status
;   962        1055  2      then
;   963        1056  3          begin
;   964        1057  3          cli$get_value (sd_pwdlifetime, tokendsc);
;   965        1058  3          if ch$eql(.tokendsc[dsc$w_length], .tokendsc[dsc$a_pointer],
;   966        1059  3              4, uplit(%ascii'NONE') ) then
;   967        1060  4             begin
;   968        1061  4                ch$fill(0, uaf$s_pwd_lifetime, recbuf[uaf$q_pwd_lifetime] );
;   969        1062  4                status = true;
;   970        1063  4             end
;   971        1064  3          else
;   972        1065  4          if not (status = lib$cvt_dtime (tokendsc, recbuf[uaf$q_pwd_lifetime]))
;   973        1066  3          then
;   974        1067  4              begin
;   975        1068  4              signal (.status);
;   976        1069  4              return false;
;   977        1070  3              end;
;   978        1071  3          RECBUF[UAF$V_PWD_EXPIRED] = FALSE;
;   979        1072  3          status = true;
;   980        1073  2          end;
;   981        1074  2
;   982        1075  2  if cli$present (sd_pwdminimum)
;   983        1076  2  then
;   984        1077  3      begin
;   985        1078  3      cli$get_value (sd_pwdminimum, tokendsc);
;   986        1079  3      if not getval (recbuf[uaf$b_pwd_length], byte_length)
;   987        1080  3      then return false;
;   988        1081  3      status = true;
;   989        1082  2      end;
;   990        1083  2
;   991        1084  2  if cli$present (sd_quepriority)
;   992        1085  2  then
;   993        1086  3      begin
;   994        1087  3      cli$get_value (sd_quepriority, tokendsc);
;   995        1088  3      if not getval (recbuf[uaf$b_quepri], byte_length)
;   996        1089  3      then return false;
;   997        1090  3      status = true;
;   998        1091  2      end;
;   999        1092  2
;  1000        1093  2  if cli$present (sd_shrfillm)
;  1001        1094  2  then
;  1002        1095  3      begin
;  1003        1096  3      cli$get_value (sd_shrfillm, tokendsc);
;  1004        1097  3      if not getval (recbuf[uaf$w_shrfillm], word_length)
;  1005        1098  3      then return false;
;  1006        1099  3      status = true;
```

```
; 1007          1100 2          end;
; 1008          1101 2
; 1009          1102 2  if cli$present (sd_tqelm)
; 1010          1103 2  then
; 1011          1104 3          begin
; 1012          1105 3          cli$get_value (sd_tqelm, tokendsc);
; 1013          1106 3          if not getval (recbuf[uaf$w_tqcnt], word_length)
; 1014          1107 3          then return false;
; 1015          1108 3          status = true;
; 1016          1109 2          end;
; 1017          1110 2
; 1018          1111 2  if cli$present (sd_uic)
; 1019          1112 2  then
; 1020          1113 3          begin
; 1021          1114 3          cli$get_value (sd_uic, tokendsc);
; 1022          1115 3          if not getuic ()
; 1023          1116 3          then return false;
; 1024          1117 3          status = true;
; 1025          1118 2          end;
; 1026          1119 2
; 1027          1120 2  if cli$present (sd_wsdefault)
; 1028          1121 2  then
; 1029          1122 3          begin
; 1030          1123 3          cli$get_value (sd_wsdefault, tokendsc);
; 1031          1124 3          if not getval (recbuf[uaf$l_dfwscnt], word_unsigned_length)
; 1032          1125 3          then return false;
; 1033          1126 3          (recbuf[uaf$l_dfwscnt])<16, 16> = 0; !** temp until longword is supported
; 1034          1127 3          status = true;
; 1035          1128 2          end;
; 1036          1129 2
; 1037          1130 2  if cli$present (sd_wsextent)
; 1038          1131 2  then
; 1039          1132 3          begin
; 1040          1133 3          cli$get_value (sd_wsextent, tokendsc);
; 1041          1134 3          if not getval (recbuf[uaf$l_wsextent], word_unsigned_length)
; 1042          1135 3          then return false;
; 1043          1136 3          (recbuf[uaf$l_wsextent])<16, 16> = 0; !** temp until longword is supported
; 1044          1137 3          status = true;
; 1045          1138 2          end;
; 1046          1139 2
; 1047          1140 2  if cli$present (sd_wsquota)
; 1048          1141 2  then
; 1049          1142 3          begin
; 1050          1143 3          cli$get_value (sd_wsquota, tokendsc);
; 1051          1144 3          if not getval (recbuf[uaf$l_wsquota], word_unsigned_length)
; 1052          1145 3          then return false;
; 1053          1146 3          (recbuf[uaf$l_wsquota])<16, 16> = 0; !** temp until longword is supported
; 1054          1147 3          status = true;
; 1055          1148 2          end;
; 1056          1149 2
; 1057          1150 2  if .uaf$gl_ctlmsk[uaf$v_copy] or .uaf$gl_ctlmsk[uaf$v_add]
; 1058          1151 2  then
; 1059          1152 2          return true;
; 1060          1153 2
; 1061          1154 2  return .status;
; 1062          1155 1  end;
```

```
                                                     .TITLE  UAFPARSE
                                                     .IDENT  \V04-000\

                                                     .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                     00000 ;TPA$KEYST0
                                           U.2:      .BLKB   0
               59  52  41  4D  49  52  50  00000 ;TPA$KEYST
                                           U.4:      .ASCII  \PRIMARY\                          ;
                                       FF  00007     .BYTE   -1                                 ;
                                           00008 ;TPA$KEYST0
                                           U.11:     .BLKB   0
       59  52  41  44  4E  4F  43  45  53  00008 ;TPA$KEYST
                                           U.13:     .ASCII  \SECONDARY\                        ;
                                       FF  00011     .BYTE   -1                                 ;
                                       FF  00012 ;TPA$KEYFILL
                                           U.21:     .BYTE   -1                                 ;

                                                     .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                                     00000 ACCESS_STATES::
                                                     .BLKB   0
                                F100  00000 ;TPA$TYPE
                                           U.5:      .WORD   -3840                              ;
                           00000000V 00002 ;TPA$ACTION
                                           U.6:      .LONG   <<CHECK_PRIMARY-U.6>-4>            ;
                           00000000* 00006 ;TPA$ADDR
                                           U.7:      .LONG   <<PRIMARY-U.7>-4>                  ;
                           00000001  0000A ;TPA$MASK
                                           U.8:      .LONG   1                                  ;
                                0000* 0000E ;TPA$TARGET
                                           U.10:     .WORD   <<U.9-U.10>-2>                     ;
                                F101  00010 ;TPA$TYPE
                                           U.14:     .WORD   -3839                              ;
                           00000000V 00012 ;TPA$ACTION
                                           U.15:     .LONG   <<CHECK_SECONDARY-U.15>-4>         ;
                           00000000* 00016 ;TPA$ADDR
                                           U.16:     .LONG   <<SECONDARY-U.16>-4>               ;
                           00000001  0001A ;TPA$MASK
                                           U.17:     .LONG   1                                  ;
                                0000* 0001E ;TPA$TARGET
                                           U.18:     .WORD   <<U.9-U.18>-2>                     ;
                                45F3  00020 ;TPA$TYPE
                                           U.19:     .WORD   17907                              ;
                           00000000* 00022 ;TPA$ADDR
                                           U.20:     .LONG   <<RIGHT_BIT-U.20>-4>               ;
                                002D  00026 ;TPA$TYPE
                                           U.22:     .WORD   45                                 ;
                                95F7  00028 ;TPA$TYPE
                                           U.23:     .WORD   -27145                             ;
                           00000000V 0002A ;TPA$ACTION
                                           U.24:     .LONG   <<SET_RANGE-U.24>-4>               ;
                                FFFF  0002E ;TPA$TARGET
                                           U.25:     .WORD   -1                                 ;
                                85F3  00030 ;TPA$TYPE
                                           U.26:     .WORD   -31245                             ;
```

```
00000000V 00032 ;TPA$ACTION
                U.27:    .LONG   <<SET_RANGE-U.27>-4>                              ;
          00036 ;TERMINAL
                U.9:     .BLKB   0
     15F7 00036 ;TPA$TYPE
                U.28:    .WORD   5623                                              ;
     FFFF 00038 ;TPA$TARGET
                U.29:    .WORD   -1                                                ;
          0003A          .BLKB   2
          0003C RESTRICT_STATES::
                         .BLKB   0
     45F3 0003C ;TPA$TYPE
                U.31:    .WORD   17907                                             ;
00000000* 0003E ;TPA$ADDR
                U.32:    .LONG   <<RIGHT_BIT-U.32>-4>                              ;
     002D 00042 ;TPA$TYPE
                U.33:    .WORD   45                                                ;
     95F7 00044 ;TPA$TYPE
                U.34:    .WORD   -27145                                            ;
00000000V 00046 ;TPA$ACTION
                U.35:    .LONG   <<SET_RANGE-U.35>-4>                              ;
     FFFF 0004A ;TPA$TARGET
                U.36:    .WORD   -1                                                ;
     85F3 0004C ;TPA$TYPE
                U.37:    .WORD   -31245                                            ;
00000000V 0004E ;TPA$ACTION
                U.38:    .LONG   <<SET_RANGE-U.38>-4>                              ;
     15F7 00052 ;TPA$TYPE
                U.39:    .WORD   5623                                              ;
     FFFF 00054 ;TPA$TARGET
                U.40:    .WORD   -1                                                ;
          00056          .BLKB   2
          00058 UIC_STATES::
                         .BLKB   0
     55EC 00058 ;TPA$TYPE
                U.42:    .WORD   21996                                             ;
00000000* 0005A ;TPA$ADDR
                U.43:    .LONG   <<CONVERTED_UIC-U.43>-4>                          ;
     FFFF 0005E ;TPA$TARGET
                U.44:    .WORD   -1                                                ;
          00060 DIR_STATES::
                         .BLKB   0
     19F8 00060 ;TPA$TYPE
                U.46:    .WORD   6648                                              ;
    0000* 00062 ;TPA$SUBEXP
                U.48:    .WORD   <<U.47-U.48>-2>                                   ;
    0000* 00064 ;TPA$TARGET
                U.50:    .WORD   <<U.49-U.50>-2>                                   ;
     19F8 00066 ;TPA$TYPE
                U.51:    .WORD   6648                                              ;
    0000* 00068 ;TPA$SUBEXP
                U.53:    .WORD   <<U.52-U.53>-2>                                   ;
    0000* 0006A ;TPA$TARGET
                U.54:    .WORD   <<U.49-U.54>-2>                                   ;
     005B 0006C ;TPA$TYPE
                U.55:    .WORD   91                                                ;
     043C 0006E ;TPA$TYPE
```

```
                                   U.56:    .WORD    1084                                        ;
                   09F8   00070  ;TPA$TYPE
                                   U.57:    .WORD    2552                                        ;
                   0000*  00072  ;TPA$SUBEXP
                                   U.58:    .WORD    <<U.47-U.58>-2>                             ;
                   0DF8   00074  ;TPA$TYPE
                                   U.59:    .WORD    3576                                        ;
                   0000*  00076  ;TPA$SUBEXP
                                   U.60:    .WORD    <<U.52-U.60>-2>                             ;
                   005D   00078  ;TPA$TYPE
                                   U.61:    .WORD    93                                          ;
                   043E   0007A  ;TPA$TYPE
                                   U.62:    .WORD    1086                                        ;
                          0007C  ;END_STATE
                                   U.49:    .BLKB    0
                   15F7   0007C  ;TPA$TYPE
                                   U.63:    .WORD    5623                                        ;
                   FFFF   0007E  ;TPA$TARGET
                                   U.64:    .WORD    -1                                          ;
                          00080  ;UFD
                                   U.47:    .BLKB    0
                   85F4   00080  ;TPA$TYPE
                                   U.65:    .WORD    -31244                                      ;
              00000000V   00082  ;TPA$ACTION
                                   U.66:    .LONG    <<CHECKVALUE-U.66>-4>                       ;
                   042C   00086  ;TPA$TYPE
                                   U.67:    .WORD    1068                                        ;
                   95F4   00088  ;TPA$TYPE
                                   U.68:    .WORD    -27148                                      ;
              00000000V   0008A  ;TPA$ACTION
                                   U.69:    .LONG    <<CHECKVALUE-U.69>-4>                       ;
                   FFFF   0008E  ;TPA$TARGET
                                   U.70:    .WORD    -1                                          ;
                          00090  ;SUBDIR
                                   U.52:    .BLKB    0
                   85F1   00090  ;TPA$TYPE
                                   U.71:    .WORD    -31247                                      ;
              00000000V   00092  ;TPA$ACTION
                                   U.72:    .LONG    <<CHECKLENGTH-U.72>-4>                      ;
                   102E   00096  ;TPA$TYPE
                                   U.73:    .WORD    4142                                        ;
                   0000*  00098  ;TPA$TARGET
                                   U.74:    .WORD    <<U.52-U.74>-2>                             ;
                   15F6   0009A  ;TPA$TYPE
                                   U.75:    .WORD    5622                                        ;
                   FFFF   0009C  ;TPA$TARGET
                                   U.76:    .WORD    -1                                          ;

                                            .PSECT   _LIB$KEY0$,NOWRT,  SHR,  PIC,1

                          00000  ACCESS_KEYS::
                                            .BLKB    0
                          00000  ;TPA$KEY0
                                   U.1:     .BLKB    0
                   0000*  00000  ;TPA$KEY
                                   U.3:     .WORD    <U.2-U.1>                                   ;
                   0000*  00002  ;TPA$KEY
```

```
                                          U.12:    .WORD   <U.11-U.1>                                  ;
                                    00004 RESTRICT_KEYS::
                                                   .BLKB   0
                                    00004 ;TPA$KEY0
                                          U.30:    .BLKB   0
                                    00004 UIC_KEYS::
                                                   .BLKB   0
                                    00004 ;TPA$KEY0
                                          U.41:    .BLKB   0
                                    00004 DIR_KEYS::
                                                   .BLKB   0
                                    00004 ;TPA$KEY0
                                          U.45:    .BLKB   0

                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

                31  6E  65  6B  6F  74  00000 P.AAB:   .ASCII  \token1\                                ;
                                        00006          .BLKB   2
                            00000006    00008 P.AAA:   .LONG   6                                       ;
                            00000000'   0000C          .ADDRESS P.AAB                                  ;
                32  6E  65  6B  6F  74  00010 P.AAD:   .ASCII  \token2\                                ;
                                        00016          .BLKB   2
                            00000006    00018 P.AAC:   .LONG   6                                       ;
                            00000000'   0001C          .ADDRESS P.AAD                                  ;
            74  6E  75  6F  63  63  61  00020 P.AAF:   .ASCII  \account\                               ;
                                        00027          .BLKB   1
                            00000007    00028 P.AAE:   .LONG   7                                       ;
                            00000000'   0002C          .ADDRESS P.AAF                                  ;
                    6D  6C  74  73  61  00030 P.AAH:   .ASCII  \astlm\                                 ;
                                        00035          .BLKB   3
                            00000005    00038 P.AAG:   .LONG   5                                       ;
                            00000000'   0003C          .ADDRESS P.AAH                                  ;
                    6D  6C  6F  69  62  00040 P.AAJ:   .ASCII  \biolm\                                 ;
                                        00045          .BLKB   3
                            00000005    00048 P.AAI:   .LONG   5                                       ;
                            00000000'   0004C          .ADDRESS P.AAJ                                  ;
                    6D  6C  74  79  62  00050 P.AAL:   .ASCII  \bytlm\                                 ;
                                        00055          .BLKB   3
                            00000005    00058 P.AAK:   .LONG   5                                       ;
                            00000000'   0005C          .ADDRESS P.AAL                                  ;
                            69  6C  63  00060 P.AAN:   .ASCII  \cli\                                   ;
                                        00063          .BLKB   1
                            00000003    00064 P.AAM:   .LONG   3                                       ;
                            00000000'   00068          .ADDRESS P.AAN                                  ;
            65  6D  69  74  75  70  63  0006C P.AAP:   .ASCII  \cputime\                               ;
                                        00073          .BLKB   1
                            00000007    00074 P.AAO:   .LONG   7                                       ;
                            00000000'   00078          .ADDRESS P.AAP                                  ;
                65  63  69  76  65  64  0007C P.AAR:   .ASCII  \device\                                ;
                                        00082          .BLKB   2
                            00000006    00084 P.AAQ:   .LONG   6                                       ;
                            00000000'   00088          .ADDRESS P.AAR                                  ;
                    6D  6C  6F  69  64  0008C P.AAT:   .ASCII  \diolm\                                 ;
                                        00091          .BLKB   3
                            00000005    00094 P.AAS:   .LONG   5                                       ;
                            00000000'   00098          .ADDRESS P.AAT                                  ;
    79  72  6F  74  63  65  72  69  64  0009C P.AAV:   .ASCII  \directory\                             ;
```

```
                                       000A5           .BLKB    3
                            00000009.  000A8 P.AAU:    .LONG    9                                         ;
                            00000000'  000AC           .ADDRESS P.AAV                                     ;
                   6D  6C  71  6E  65  000B0 P.AAX:    .ASCII   \enqlm\                                   ;
                                       000B5           .BLKB    3
                            00000005.  000B8 P.AAW:    .LONG    5                                         ;
                            00000000'  000BC           .ADDRESS P.AAX                                     ;
                   6D  6C  6C  69  66  000C0 P.AAZ:    .ASCII   \fillm\                                   ;
                                       000C5           .BLKB    3
                            00000005.  000C8 P.AAY:    .LONG    5                                         ;
                            00000000'  000CC           .ADDRESS P.AAZ                                     ;
                   73  67  61  6C  66  000D0 P.ABB:    .ASCII   \flags\                                   ;
                                       000D5           .BLKB    3
                            00000005.  000D8 P.ABA:    .LONG    5                                         ;
                            00000000'  000DC           .ADDRESS P.ABB                                     ;
               64  6D  63  69  67  6C  000E0 P.ABD:    .ASCII   \lgicmd\                                  ;
                                       000E6           .BLKB    2
                            00000006.  000E8 P.ABC:    .LONG    6                                         ;
                            00000000'  000EC           .ADDRESS P.ABD                                     ;
           73  62  6F  6A  78  61  6D  000F0 P.ABF:    .ASCII   \maxjobs\                                 ;
                                       000F7           .BLKB    1
                            00000007.  000F8 P.ABE:    .LONG    7                                         ;
                            00000000'  000FC           .ADDRESS P.ABF                                     ;
                   72  65  6E  77  6F  00100 P.ABH:    .ASCII   \owner\                                   ;
                                       00105           .BLKB    3
                            00000005.  00108 P.ABG:    .LONG    5                                         ;
                            00000000'  0010C           .ADDRESS P.ABH                                     ;
       64  72  6F  77  73  73  61  70  00110 P.ABJ:    .ASCII   \password\                                ;
                            00000008.  00118 P.ABI:    .LONG    8                                         ;
                            00000000'  0011C           .ADDRESS P.ABJ                                     ;
               6D  6C  74  79  62  70  00120 P.ABL:    .ASCII   \pbytlm\                                  ;
                                       00126           .BLKB    2
                            00000006.  00128 P.ABK:    .LONG    6                                         ;
                            00000000'  0012C           .ADDRESS P.ABL                                     ;
               73  67  61  6C  66  70  00130 P.ABN:    .ASCII   \pflags\                                  ;
                                       00136           .BLKB    2
                            00000006.  00138 P.ABM:    .LONG    6                                         ;
                            00000000'  0013C           .ADDRESS P.ABN                                     ;
   61  74  6F  75  71  6C  66  67  70  00140 P.ABP:    .ASCII   \pgflquota\                               ;
                                       00149           .BLKB    3
                            00000009.  0014C P.ABO:    .LONG    9                                         ;
                            00000000'  00150           .ADDRESS P.ABP                                     ;
74 63  69  72  74  73  65  72  5F  70  00154 P.ABR:    .ASCII   \p_restrict\                              ;
                                       0015E           .BLKB    2
                            0000000A.  00160 P.ABQ:    .LONG    10                                        ;
                            00000000'  00164           .ADDRESS P.ABR                                     ;
                   6D  6C  63  72  70  00168 P.ABT:    .ASCII   \prclm\                                   ;
                                       0016D           .BLKB    3
                            00000005.  00170 P.ABS:    .LONG    5                                         ;
                            00000000'  00174           .ADDRESS P.ABT                                     ;
       79  74  69  72  6F  69  72  70  00178 P.ABV:    .ASCII   \priority\                                ;
                            00000008.  00180 P.ABU:    .LONG    8                                         ;
                            00000000'  00184           .ADDRESS P.ABV                                     ;
73 65  67  65  6C  69  76  69  72  70  00188 P.ABX:    .ASCII   \privileges\                              ;
                                       00192           .BLKB    2
                            0000000A.  00194 P.ABW:    .LONG    10                                        ;
                            00000000'  00198           .ADDRESS P.ABX                                     ;
```

```
                               73  67  61  6C  66  73  0019C P.ABZ:   .ASCII  \sflags\                                  ;
                                                       001A2          .BLKB   2
                                             00000006  001A4 P.ABY:   .LONG   6                                         ;
                                             00000000' 001A8          .ADDRESS P.ABZ                                    ;
       74  63  69  72  74  73  65  72  5F  73  001AC P.ACB:   .ASCII  \s_restrict\                              ;
                                                       001B6          .BLKB   2
                                             0000000A  001B8 P.ACA:   .LONG   10                                        ;
                                             00000000' 001BC          .ADDRESS P.ACB                                    ;
                   6D  6C  6C  69  66  72  68  73  001C0 P.ACD:   .ASCII  \shrfillm\                                ;
                                             00000008  001C8 P.ACC:   .LONG   8                                         ;
                                             00000000' 001CC          .ADDRESS P.ACD                                    ;
                                   6D  6C  65  71  74  001D0 P.ACF:   .ASCII  \tqelm\                                   ;
                                                       001D5          .BLKB   3
                                             00000005  001D8 P.ACE:   .LONG   5                                         ;
                                             00000000' 001DC          .ADDRESS P.ACF                                    ;
                                           63  69  75  001E0 P.ACH:   .ASCII  \uic\                                     ;
                                                       001E3          .BLKB   1
                                             00000003  001E4 P.ACG:   .LONG   3                                         ;
                                             00000000' 001E8          .ADDRESS P.ACH                                    ;
               74  6C  75  61  66  65  64  73  77  001EC P.ACJ:   .ASCII  \wsdefault\                               ;
                                                       001F5          .BLKB   3
                                             00000009  001F8 P.ACI:   .LONG   9                                         ;
                                             00000000' 001FC          .ADDRESS P.ACJ                                    ;
                   74  6E  65  74  78  65  73  77  00200 P.ACL:   .ASCII  \wsextent\                                ;
                                             00000008  00208 P.ACK:   .LONG   8                                         ;
                                             00000000' 0020C          .ADDRESS P.ACL                                    ;
                       61  74  6F  75  71  73  77  00210 P.ACN:   .ASCII  \wsquota\                                 ;
                                                       00217          .BLKB   1
                                             00000007  00218 P.ACM:   .LONG   7                                         ;
                                             00000000' 0021C          .ADDRESS P.ACN                                    ;
                       79  61  6C  70  73  69  64  00220 P.ACP:   .ASCII  \display\                                 ;
                                                       00227          .BLKB   1
                                             00000007  00228 P.ACO:   .LONG   7                                         ;
                                             00000000' 0022C          .ADDRESS P.ACP                                    ;
               73  79  61  64  65  6D  69  72  70  00230 P.ACR:   .ASCII  \primedays\                               ;
                                                       00239          .BLKB   3
                                             00000009  0023C P.ACQ:   .LONG   9                                         ;
                                             00000000' 00240          .ADDRESS P.ACR                                    ;
   73  62  6F  6A  74  63  63  61  78  61  6D  00244 P.ACT:   .ASCII  \maxacctjobs\                             ;
                                                       0024F          .BLKB   1
                                             0000000B  00250 P.ACS:   .LONG   11                                        ;
                                             00000000' 00254          .ADDRESS P.ACT                                    ;
               68  63  61  74  65  64  78  61  6D  00258 P.ACV:   .ASCII  \maxdetach\                               ;
                                                       00261          .BLKB   3
                                             00000009  00264 P.ACU:   .LONG   9                                         ;
                                             00000000' 00268          .ADDRESS P.ACV                                    ;
               73  65  6C  62  61  74  69  6C  63  0026C P.ACX:   .ASCII  \clitables\                               ;
                                                       00275          .BLKB   3
                                             00000009  00278 P.ACW:   .LONG   9                                         ;
                                             00000000' 0027C          .ADDRESS P.ACX                                    ;
                       61  74  6F  75  71  74  6A  00280 P.ACZ:   .ASCII  \jtquota\                                 ;
                                                       00287          .BLKB   1
                                             00000007  00288 P.ACY:   .LONG   7                                         ;
                                             00000000' 0028C          .ADDRESS P.ACZ                                    ;
                           73  73  65  63  63  61  00290 P.ADB:   .ASCII  \access\                                  ;
                                                       00296          .BLKB   2
                                             00000006  00298 P.ADA:   .LONG   6                                         ;
```

```
                                    00000000'  0029C           .ADDRESS P.ADB                                 ;
                        68  63  74  61  62     002A0 P.ADD:    .ASCII   \batch\                               ;
                                               002A5           .BLKB    3
                                    00000005.  002A8 P.ADC:    .LONG    5                                     ;
                                    00000000'  002AC           .ADDRESS P.ADD                                 ;
73  65  67  65  6C  69  76  69  72  70  66  65  64     002B0 P.ADF:    .ASCII   \defprivileges\               ;
                                               002BD           .BLKB    3
                                    0000000D.  002C0 P.ADE:    .LONG    13                                    ;
                                    00000000'  002C4           .ADDRESS P.ADF                                 ;
                    70  75  6C  61  69  64     002C8 P.ADH:    .ASCII   \dialup\                              ;
                                               002CE           .BLKB    2
                                    00000006.  002D0 P.ADG:    .LONG    6                                     ;
                                    00000000'  002D4           .ADDRESS P.ADH                                 ;
        6E  6F  69  74  61  72  69  70  78  65     002D8 P.ADJ:    .ASCII   \expiration\                      ;
                                               002E2           .BLKB    2
                                    0000000A.  002E4 P.ADI:    .LONG    10                                    ;
                                    00000000'  002E8           .ADDRESS P.ADJ                                 ;
    65  76  69  74  63  61  72  65  74  6E  69     002EC P.ADL:    .ASCII   \interactive\                     ;
                                               002F7           .BLKB    1
                                    0000000B.  002F8 P.ADK:    .LONG    11                                    ;
                                    00000000'  002FC           .ADDRESS P.ADL                                 ;
                        6C  61  63  6F  6C     00300 P.ADN:    .ASCII   \local\                               ;
                                               00305           .BLKB    3
                                    00000005.  00308 P.ADM:    .LONG    5                                     ;
                                    00000000'  0030C           .ADDRESS P.ADN                                 ;
                6B  72  6F  77  74  65  6E     00310 P.ADP:    .ASCII   \network\                             ;
                                               00317           .BLKB    1
                                    00000007.  00318 P.ADO:    .LONG    7                                     ;
                                    00000000'  0031C           .ADDRESS P.ADP                                 ;
    65  6D  69  74  65  66  69  6C  64  77  70     00320 P.ADR:    .ASCII   \pwdlifetime\                     ;
                                               0032B           .BLKB    1
                                    0000000B.  0032C P.ADQ:    .LONG    11                                    ;
                                    00000000'  00330           .ADDRESS P.ADR                                 ;
        6D  75  6D  69  6E  69  6D  64  77  70     00334 P.ADT:    .ASCII   \pwdminimum\                      ;
                                               0033E           .BLKB    2
                                    0000000A.  00340 P.ADS:    .LONG    10                                    ;
                                    00000000'  00344           .ADDRESS P.ADT                                 ;
    79  74  69  72  6F  69  72  70  65  75  71     00348 P.ADV:    .ASCII   \quepriority\                     ;
                                               00353           .BLKB    1
                                    0000000B.  00354 P.ADU:    .LONG    11                                    ;
                                    00000000'  00358           .ADDRESS P.ADV                                 ;
                    65  74  6F  6D  65  72     0035C P.ADX:    .ASCII   \remote\                              ;
                                               00362           .BLKB    2
                                    00000006.  00364 P.ADW:    .LONG    6                                     ;
                                    00000000'  00368           .ADDRESS P.ADX                                 ;
            59  4C  54  43  53  49  44  07     0036C P.ADY:    .ASCII   <7>\DISCTLY\                          ;
            00  49  4C  43  46  45  44  06     00374 P.ADZ:    .ASCII   <6>\DEFCLI\<0>                        ;
            44  57  50  4B  43  4F  4C  07     0037C P.AEA:    .ASCII   <7>\LOCKPWD\                          ;
            45  56  49  54  50  41  43  07     00384 P.AEB:    .ASCII   <7>\CAPTIVE\                          ;
            52  45  53  55  53  49  44  07     0038C P.AEC:    .ASCII   <7>\DISUSER\                          ;
00  45  4D  4F  43  4C  45  57  53  49  44  0A     00394 P.AED:    .ASCII   <10>\DISWELCOME\<0>               ;
00  4C  49  41  4D  57  45  4E  53  49  44  0A     003A0 P.AEE:    .ASCII   <10>\DISNEWMAIL\<0>               ;
            4C  49  41  4D  53  49  44  07     003AC P.AEF:    .ASCII   <7>\DISMAIL\                          ;
            00  44  57  50  4E  45  47  06     003B4 P.AEG:    .ASCII   <6>\GENPWD\<0>                        ;
44  45  52  49  50  58  45  5F  44  57  50  0B     003BC P.AEH:    .ASCII   <11>\PWD_EXPIRED\                 ;
00  00  44  45  52  49  50  58  45  5F  32  44  57  50  0C     003C8 P.AEI:    .ASCII   <12>\PWD2_EXPIRED\<0><0><0>   ;
                                        00     003D7
```

```
                            00  00  54  49  44  55  41  05  003D8 P.AEJ:   .ASCII  <5>\AUDIT\<0><0>                     ;
            00  00  54  52  4F  50  45  52  53  49  44  09  003E0 P.AEK:   .ASCII  <9>\DISREPORT\<0><0>                 ;
            00  00  50  55  4C  41  49  44  53  49  44  09  003EC P.AEL:   .ASCII  <9>\DISDIALUP\<0><0>                 ;
            00  4B  52  4F  57  54  45  4E  53  49  44  0A  003F8 P.AEM:   .ASCII  <10>\DISNETWORK\<0>                  ;
                            00  59  41  44  4E  4F  4D  06  00404 P.AEN:   .ASCII  <6>\MONDAY\<0>                       ;
                            59  41  44  53  45  55  54  07  0040C P.AEO:   .ASCII  <7>\TUESDAY\                         ;
            00  00  59  41  44  53  45  4E  44  45  57  09  00414 P.AEP:   .ASCII  <9>\WEDNESDAY\<0><0>                 ;
            00  00  00  59  41  44  53  52  55  48  54  08  00420 P.AEQ:   .ASCII  <8>\THURSDAY\<0><0><0>               ;
                            00  59  41  44  49  52  46  06  0042C P.AER:   .ASCII  <6>\FRIDAY\<0>                       ;
            00  00  00  59  41  44  52  55  54  41  53  08  00434 P.AES:   .ASCII  <8>\SATURDAY\<0><0><0>               ;
                            00  59  41  44  4E  55  53  06  00440 P.AET:   .ASCII  <6>\SUNDAY\<0>                       ;
45  44  43  42  41  39  38  37  36  35  34  33  32  31  30  00448 P.AEU:   .ASCII  \0123456789ABCDEF\                   ;
                                                        46  00457                                                       ;
                                                    4F  4E  00458 P.AEW:   .ASCII  \NO\                                 ;
                                                            0045A          .BLKB   2                                    ;
                                                  00000002  0045C P.AEV:   .LONG   2                                    ;
                                                  00000000' 00460          .ADDRESS P.AEW                               ;
4F  57  53  53  41  50  5F  45  54  41  52  45  4E  45  47  00464 P.AEY:   .ASCII  \GENERATE_PASSWORD\<0><0><0>         ;
                                        00  00  00  44  52  00473                                                       ;
                                                  010E0011  00478 P.AEX:   .LONG   17694737                             ;
                                                  00000000' 0047C          .ADDRESS P.AEY                               ;
4F  57  53  53  41  50  5F  45  54  41  52  45  4E  45  47  00480 P.AFA:   .ASCII  \GENERATE_PASSWORD\<0><0><0>         ;
                                        00  00  00  44  52  0048F                                                       ;
                                                  010E0011  00494 P.AEZ:   .LONG   17694737                             ;
                                                  00000000' 00498          .ADDRESS P.AFA                               ;
4F  57  53  53  41  50  5F  45  54  41  52  45  4E  45  47  0049C P.AFC:   .ASCII  \GENERATE_PASSWORD\<0><0><0>         ;
                                        00  00  00  44  52  004AB                                                       ;
                                                  010E0011  004B0 P.AFB:   .LONG   17694737                             ;
                                                  00000000' 004B4          .ADDRESS P.AFC                               ;
            00  00  44  45  52  49  50  58  45  44  57  50  004B8 P.AFE:   .ASCII  \PWDEXPIRED\<0><0>                   ;
                                                  010E000A  004C4 P.AFD:   .LONG   17694730                             ;
                                                  00000000' 004C8          .ADDRESS P.AFE                               ;
                                            45  4E  4F  4E  004CC P.AFF:   .ASCII  \NONE\                               ;

                                                                           .PSECT  $OWN$,NOEXE,2

                                                  0000001A  00000 FLAG_TABLE:
                                                                           .LONG   26                                   ;
                                                  00000000' 00004          .ADDRESS P.ADY                               ;
                                                  00000000  00008          .LONG   0                                    ;
                                                  00000000' 0000C          .ADDRESS P.ADZ                               ;
                                                  00000001  00010          .LONG   1                                    ;
                                                  00000000' 00014          .ADDRESS P.AEA                               ;
                                                  00000002  00018          .LONG   2                                    ;
                                                  00000000' 0001C          .ADDRESS P.AEB                               ;
                                                  00000003  00020          .LONG   3                                    ;
                                                  00000000' 00024          .ADDRESS P.AEC                               ;
                                                  00000004  00028          .LONG   4                                    ;
                                                  00000000' 0002C          .ADDRESS P.AED                               ;
                                                  00000005  00030          .LONG   5                                    ;
                                                  00000000' 00034          .ADDRESS P.AEE                               ;
                                                  00000006  00038          .LONG   6                                    ;
                                                  00000000' 0003C          .ADDRESS P.AEF                               ;
                                                  00000007  00040          .LONG   7                                    ;
                                                  00000000' 00044          .ADDRESS P.AEG                               ;
                                                  00000008  00048          .LONG   8                                    ;
                                                  00000000' 0004C          .ADDRESS P.AEH                               ;
```

```
          00000009  00050         .LONG   9                                            ;
          00000000' 00054         .ADDRESS P.AEI                                       ;
          0000000A  00058         .LONG   10                                           ;
          00000000' 0005C         .ADDRESS P.AEJ                                       ;
          0000000B  00060         .LONG   11                                           ;
          00000000' 00064         .ADDRESS P.AEK                                       ;
          0000000C  00068         .LONG   12                                           ;
          00000004  0006C OPFLAG_TABLE:
                                  .LONG   4                                            ;
          00000000' 00070         .ADDRESS P.AEL                                       ;
          00000001  00074         .LONG   1                                            ;
          00000000' 00078         .ADDRESS P.AEM                                       ;
          00000002  0007C         .LONG   2                                            ;
          0000000E  00080 PDFLAG_TABLE:
                                  .LONG   14                                           ;
          00000000' 00084         .ADDRESS P.AEN                                       ;
          00000000  00088         .LONG   0                                            ;
          00000000' 0008C         .ADDRESS P.AEO                                       ;
          00000001  00090         .LONG   1                                            ;
          00000000' 00094         .ADDRESS P.AEP                                       ;
          00000002  00098         .LONG   2                                            ;
          00000000' 0009C         .ADDRESS P.AEQ                                       ;
          00000003  000A0         .LONG   3                                            ;
          00000000' 000A4         .ADDRESS P.AER                                       ;
          00000004  000A8         .LONG   4                                            ;
          00000000' 000AC         .ADDRESS P.AES                                       ;
          00000005  000B0         .LONG   5                                            ;
          00000000' 000B4         .ADDRESS P.AET                                       ;
          00000006  000B8         .LONG   6                                            ;
                    000BC STATUS: .BLKB   4
                    000C0 CLI_STATUS:
                                  .BLKB   4
00000002  00000008  000C4 TPARSE_BLOCK:
                                  .LONG   8, 2                                         ;
                    000CC         .BLKB   28
                    000E8 CONVERTED_UIC:
                                  .BLKB   4
                    000EC ADDRESS:.BLKB   4
                    000F0 LEFT_BIT:
                                  .BLKB   4
                    000F4 RIGHT_BIT:
                                  .BLKB   4
                    000F8 PRIMARY_ACCESS:
                                  .BLKB   4
                    000FC SECONDARY_ACCESS:
                                  .BLKB   4
          00000000  00100 NO_FLAG:.LONG   0                                            ;
          00000000  00104 PRIMARY:.LONG   0                                            ;
          00000000  00108 SECONDARY:
                                  .LONG   0                                            ;

                          SD_TOKEN1=          P.AAA
                          SD_TOKEN2=          P.AAC
                          SD_ACCOUNT=         P.AAE
                          SD_ASTLM=           P.AAG
                          SD_BIOLM=           P.AAI
                          SD_BYTLM=           P.AAK
```

```
SD_CLI=            P.AAM
SD_CPUTIME=        P.AAO
SD_DEVICE=         P.AAQ
SD_DIOLM=          P.AAS
SD_DIRECTORY=      P.AAU
SD_ENQLM=          P.AAW
SD_FILLM=          P.AAY
SD_FLAGS=          P.ABA
SD_LGICMD=         P.ABC
SD_MAXJOBS=        P.ABE
SD_OWNER=          P.ABG
SD_PASSWORD=       P.ABI
SD_PBYTLM=         P.ABK
SD_PFLAGS=         P.ABM
SD_PGFLQUOTA=      P.ABO
SD_P_RESTRICT=     P.ABQ
SD_PRCLM=          P.ABS
SD_PRIORITY=       P.ABU
SD_PRIVILEGES=     P.ABW
SD_SFLAGS=         P.ABY
SD_S_RESTRICT=     P.ACA
SD_SHRFILLM=       P.ACC
SD_TQELM=          P.ACE
SD_UIC=            P.ACG
SD_WSDEFAULT=      P.ACI
SD_WSEXTENT=       P.ACK
SD_WSQUOTA=        P.ACM
SD_DISPLAY=        P.ACO
SD_PRIMEDAYS=      P.ACQ
SD_MAXACCTJOBS=    P.ACS
SD_MAXDETACH=      P.ACU
SD_CLITABLES=      P.ACW
SD_JTQUOTA=        P.ACY
SD_ACCESS=         P.ADA
SD_BATCH=          P.ADC
SD_DEFPRIVILEGES=  P.ADE
SD_DIALUP=         P.ADG
SD_EXPIRATION=     P.ADI
SD_INTERACTIVE=    P.ADK
SD_LOCAL=          P.ADM
SD_NETWORK=        P.ADO
SD_PWDLIFETIME=    P.ADQ
SD_PWDMINIMUM=     P.ADS
SD_QUEPRIORITY=    P.ADU
SD_REMOTE=         P.ADW
NUMBERS=           P.AEU
NO_DSC=            P.AEV
        .EXTRN  GENERATE_PASSWORDS
        .EXTRN  SYS$ASSIGN, SYS$DASSGN
        .EXTRN  SYS$QIOW, LIB$GET_VM
        .EXTRN  STR$UPCASE, LIB$TPARSE
        .EXTRN  LIB$LOOKUP_KEY, CLI$DCL_PARSE
        .EXTRN  CLI$PRESENT, CLI$GET_VALUE
        .EXTRN  FAOOUT, LGI$HPWD
        .EXTRN  LIB$CVT_DTIME, LIB$CVT_TIME
        .EXTRN  PRV$SETPRIV, LIB$SIGNAL
        .EXTRN  UAF$_BADVALUE, UAF$_EXTRAPARM
```

```
                                                   .EXTRN  UAF$_INVDEV, UAF$_INVSTR
                                                   .EXTRN  UAF$_INVUSERSPEC
                                                   .EXTRN  UAF$_KEYNOTFND, UAF$_NAMETOOBIG
                                                   .EXTRN  UAF$_NOARG, UAF$_NOQUOTE
                                                   .EXTRN  UAF$_NOUSERSPEC
                                                   .EXTRN  UAF$_PRVNOTFND, UAF$_PRVNOTUNQ
                                                   .EXTRN  UAF$_PWDSYNTAX, UAF$_PWDNCH
                                                   .EXTRN  UAF$_PWDNOL, UAF$_UICERR
                                                   .EXTRN  UAF$_VALTOOBIG, UAF$_ZCONFLICT
                                                   .EXTRN  ENCRYPT, LIB$_AMBKEY
                                                   .EXTRN  LIB$_UNRKEY, CLI$_ABKEYW
                                                   .EXTRN  CLI$_IVKEYW, CLI$_NEGATED
                                                   .EXTRN  CLI$_ABSENT, PRV$_NOTUNQ
                                                   .EXTRN  PRV$_INVNAM, UAF$GL_CTLMSK
                                                   .EXTRN  AUTHORIZE_COMMANDS
                                                   .EXTRN  CMDLINDSC, CALL_COUNT
                                                   .EXTRN  SYMBOL_STR, MATCH_TOKENLEN
                                                   .EXTRN  MATCH_TOKEN, TOKENDSC
                                                   .EXTRN  TOKENLEN, TOKENPTR
                                                   .EXTRN  UIC_FLAG, GRP_WILD
                                                   .EXTRN  MEM_WILD, STR_WILD
                                                   .EXTRN  TIME_BUF, PWD_FLAG
                                                   .EXTRN  RECBUF

                                                   .PSECT  $CODE$,NOWRT,2

                            0FFC 00000             .ENTRY  UPDATE_RECORD, Save R2,R3,R4,R5,R6,R7,R8,-   ; 0588
                                                           R9,R10,R11                                   ;
            5B 00000000G   00  9E 00002            MOVAB   CLI$GET_VALUE, R11                           ;
            5A 00000000G   00  9E 00009            MOVAB   TOKENDSC, R10                                ;
            59 00000000G   00  9E 00010            MOVAB   RECBUF+468, R9                               ;
            58 00000000G   00  9E 00017            MOVAB   CLI$PRESENT, R8                              ;
            57 00000000'   00  9E 0001E            MOVAB   STATUS, R7                                   ;
            56 00000000'   00  9E 00025            MOVAB   SD_PASSWORD, R6                              ;
                           67  D4 0002C            CLRL    STATUS                                       ; 0623
                     48    A7  7C 0002E            CLRQ    PRIMARY                                      ; 0624
                     44    A7  D4 00031            CLRL    NO_FLAG                                      ; 0626
               00000000G   00  D5 00034            TSTL    CALL_COUNT                                   ; 0632
                           13  13 0003A            BEQL    1$                                           ;
               00000000G   00  9F 0003C            PUSHAB  AUTHORIZE_COMMANDS                           ; 0634
               00000000G   00  9F 00042            PUSHAB  CMDLINDSC                                    ;
00000000G   00             02  FB 00048            CALLS   #2, CLI$DCL_PARSE                            ;
                    0360   C6  9F 0004F 1$:        PUSHAB  P.AEX                                        ; 0637
            68             01  FB 00053            CALLS   #1, CLI$PRESENT                              ;
            1D             50  E9 00056            BLBC    R0, 2$                                       ;
                           56  DD 00059            PUSHL   R6                                           ; 0638
            68             01  FB 0005B            CALLS   #1, CLI$PRESENT                              ;
00000000G   8F             50  D1 0005E            CMPL    R0, #CLI$_ABSENT                             ;
                           0F  13 00065            BEQL    2$                                           ;
               00000000G   8F  DD 00067            PUSHL   #UAF$_ZCONFLICT                              ; 0641
00000000G   00             01  FB 0006D            CALLS   #1, LIB$SIGNAL                               ;
                           3A  11 00074            BRB     5$                                           ; 0637
                    037C   C6  9F 00076 2$:        PUSHAB  P.AEZ                                        ; 0644
            68             01  FB 0007A            CALLS   #1, CLI$PRESENT                              ;
            0E             50  E8 0007D            BLBS    R0, 3$                                       ;
                           56  DD 00080            PUSHL   R6                                           ; 0645
            68             01  FB 00082            CALLS   #1, CLI$PRESENT                              ;
```

```
          00000000G  8F           50  D1 00085        CMPL    R0, #CLI$_ABSENT                    ;
                                  22  13 0008C        BEQL    5$                                  ;
                           0398   C6  9F 0008E  3$:   PUSHAB  P.AFB                               ; 0648
                     68           01  FB 00092        CALLS   #1, CLI$PRESENT                     ;
                     07           50  E9 00095        BLBC    R0, 4$                              ;
          00000000V  00           00  FB 00098        CALLS   #0, UAF$GENERATE                    ; 0649
          00000000V  00           00  FB 0009F  4$:   CALLS   #0, GETPASSWORD                     ; 0650
                     63           50  E9 000A6        BLBC    R0, 10$                             ;
                01   A9           02  8A 000A9        BICB2   #2, RECBUF+469                      ; 0652
                     67           01  D0 000AD        MOVL    #1, STATUS                          ; 0653
                     03 00000000G 00  E9 000B0  5$:   BLBC    UAF$GL_CTLMSK, 6$                   ; 0658
                              0761    31 000B7        BRW     105$                                ;
                           0180   C6  9F 000BA  6$:   PUSHAB  SD_ACCESS                           ; 0667
                     68           01  FB 000BE        CALLS   #1, CLI$PRESENT                     ;
                04   A7           50  D0 000C1        MOVL    R0, CLI_STATUS                      ;
                     09           50  E8 000C5        BLBS    R0, 7$                              ; 0668
          00000000G  8F           50  D1 000C8        CMPL    R0, #CLI$_NEGATED                   ; 0669
                                  15  12 000CF        BNEQ    8$                                  ;
                                  50  DD 000D1  7$:   PUSHL   R0                                  ; 0672
                                  1F  DD 000D3        PUSHL   #31                                 ;
                           0180   C6  9F 000D5        PUSHAB  SD_ACCESS                           ;
          00000000V  00           03  FB 000D9        CALLS   #3, GETACCESS                       ;
                     55           50  E9 000E0        BLBC    R0, 13$                             ;
                     67           01  D0 000E3        MOVL    #1, STATUS                          ; 0674
                           01E0   C6  9F 000E6  8$:   PUSHAB  SD_INTERACTIVE                      ; 0677
                     68           01  FB 000EA        CALLS   #1, CLI$PRESENT                     ;
                04   A7           50  D0 000ED        MOVL    R0, CLI_STATUS                      ;
                     09           50  E8 000F1        BLBS    R0, 9$                              ; 0678
          00000000G  8F           50  D1 000F4        CMPL    R0, #CLI$_NEGATED                   ; 0679
                                  15  12 000FB        BNEQ    11$                                 ;
                                  50  DD 000FD  9$:   PUSHL   R0                                  ; 0682
                                  1C  DD 000FF        PUSHL   #28                                 ;
                           01E0   C6  9F 00101        PUSHAB  SD_INTERACTIVE                      ;
          00000000V  00           03  FB 00105        CALLS   #3, GETACCESS                       ;
                     55           50  E9 0010C  10$:  BLBC    R0, 16$                             ;
                     67           01  D0 0010F        MOVL    #1, STATUS                          ; 0684
                           0190   C6  9F 00112  11$:  PUSHAB  SD_BATCH                            ; 0687
                     68           01  FB 00116        CALLS   #1, CLI$PRESENT                     ;
                04   A7           50  D0 00119        MOVL    R0, CLI_STATUS                      ;
                     09           50  E8 0011D        BLBS    R0, 12$                             ; 0688
          00000000G  8F           50  D1 00120        CMPL    R0, #CLI$_NEGATED                   ; 0689
                                  15  12 00127        BNEQ    14$                                 ;
                                  50  DD 00129  12$:  PUSHL   R0                                  ; 0692
                                  02  DD 0012B        PUSHL   #2                                  ;
                           0190   C6  9F 0012D        PUSHAB  SD_BATCH                            ;
          00000000V  00           03  FB 00131        CALLS   #3, GETACCESS                       ;
                     55           50  E9 00138  13$:  BLBC    R0, 19$                             ;
                     67           01  D0 0013B        MOVL    #1, STATUS                          ; 0694
                           01B8   C6  9F 0013E  14$:  PUSHAB  SD_DIALUP                           ; 0697
                     68           01  FB 00142        CALLS   #1, CLI$PRESENT                     ;
                04   A7           50  D0 00145        MOVL    R0, CLI_STATUS                      ;
                     09           50  E8 00149        BLBS    R0, 15$                             ; 0698
          00000000G  8F           50  D1 0014C        CMPL    R0, #CLI$_NEGATED                   ; 0699
                                  15  12 00153        BNEQ    17$                                 ;
                                  50  DD 00155  15$:  PUSHL   R0                                  ; 0702
                                  08  DD 00157        PUSHL   #8                                  ;
                           01B8   C6  9F 00159        PUSHAB  SD_DIALUP                           ;
```

```
00000000V  00        03  FB  0015D        CALLS   #3, GETACCESS          ;
           55        50  E9  00164  16$:  BLBC    R0, 22$                ;
           67        01  D0  00167        MOVL    #1, STATUS             ; 0704
               0200  C6  9F  0016A  17$:  PUSHAB  SD_NETWORK             ; 0707
           68        01  FB  0016E        CALLS   #1, CLI$PRESENT        ;
       04  A7        50  D0  00171        MOVL    R0, CLI_STATUS         ;
           09        50  E8  00175        BLBS    R0, 18$                ; 0708
00000000G  8F        50  D1  00178        CMPL    R0, #CLI$_NEGATED      ; 0709
                     15  12  0017F        BNEQ    20$                    ;
                     50  DD  00181  18$:  PUSHL   R0                     ; 0712
                     01  DD  00183        PUSHL   #1                     ;
               0200  C6  9F  00185        PUSHAB  SD_NETWORK             ;
00000000V  00        03  FB  00189        CALLS   #3, GETACCESS          ;
           7D        50  E9  00190  19$:  BLBC    R0, 26$                ;
           67        01  D0  00193        MOVL    #1, STATUS             ; 0714
               01F0  C6  9F  00196  20$:  PUSHAB  SD_LOCAL               ; 0717
           68        01  FB  0019A        CALLS   #1, CLI$PRESENT        ;
       04  A7        50  D0  0019D        MOVL    R0, CLI_STATUS         ;
           09        50  E8  001A1        BLBS    R0, 21$                ; 0718
00000000G  8F        50  D1  001A4        CMPL    R0, #CLI$_NEGATED      ; 0719
                     15  12  001AB        BNEQ    23$                    ;
                     50  DD  001AD  21$:  PUSHL   R0                     ; 0722
                     04  DD  001AF        PUSHL   #4                     ;
               01F0  C6  9F  001B1        PUSHAB  SD_LOCAL               ;
00000000V  00        03  FB  001B5        CALLS   #3, GETACCESS          ;
           76        50  E9  001BC  22$:  BLBC    R0, 28$                ;
           67        01  D0  001BF        MOVL    #1, STATUS             ; 0724
               024C  C6  9F  001C2  23$:  PUSHAB  SD_REMOTE              ; 0727
           68        01  FB  001C6        CALLS   #1, CLI$PRESENT        ;
       04  A7        50  D0  001C9        MOVL    R0, CLI_STATUS         ;
           09        50  E8  001CD        BLBS    R0, 24$                ; 0728
00000000G  8F        50  D1  001D0        CMPL    R0, #CLI$_NEGATED      ; 0729
                     15  12  001D7        BNEQ    25$                    ;
                     50  DD  001D9  24$:  PUSHL   R0                     ; 0732
                     10  DD  001DB        PUSHL   #16                    ;
               024C  C6  9F  001DD        PUSHAB  SD_REMOTE              ;
00000000V  00        03  FB  001E1        CALLS   #3, GETACCESS          ;
           6F        50  E9  001E8        BLBC    R0, 30$                ;
           67        01  D0  001EB        MOVL    #1, STATUS             ; 0734
               FF10  C6  9F  001EE  25$:  PUSHAB  SD_ACCOUNT             ; 0737
           68        01  FB  001F2        CALLS   #1, CLI$PRESENT        ;
           1E        50  E9  001F5        BLBC    R0, 27$                ;
                     5A  DD  001F8        PUSHL   R10                    ; 0740
               FF10  C6  9F  001FA        PUSHAB  SD_ACCOUNT             ;
           6B        02  FB  001FE        CALLS   #2, CLI$GET_VALUE      ;
                     02  DD  00201        PUSHL   #2                     ; 0742
                     08  DD  00203        PUSHL   #8                     ;
               FE60  C9  9F  00205        PUSHAB  RECBUF+52              ;
00000000V  00        03  FB  00209        CALLS   #3, GETSTRING          ;
           6C        50  E9  00210  26$:  BLBC    R0, 32$                ;
           67        01  D0  00213        MOVL    #1, STATUS             ; 0744
               FF20  C6  9F  00216  27$:  PUSHAB  SD_ASTLM               ; 0747
           68        01  FB  0021A        CALLS   #1, CLI$PRESENT        ;
           1B        50  E9  0021D        BLBC    R0, 29$                ;
                     5A  DD  00220        PUSHL   R10                    ; 0750
               FF20  C6  9F  00222        PUSHAB  SD_ASTLM               ;
           6B        02  FB  00226        CALLS   #2, CLI$GET_VALUE      ;
```

```
                                                  10  DD 00229         PUSHL   #16                               ; 0751
                                          40      A9  9F 0022B         PUSHAB  RECBUF+532                        ;
                       00000000V  00              02  FB 0022E         CALLS   #2, GETVAL                        ;
                                  6F              50  E9 00235  28$:   BLBC    R0, 34$                           ;
                                  67              01  D0 00238         MOVL    #1, STATUS                        ; 0753
                                        FF30      C6  9F 0023B  29$:   PUSHAB  SD_BIOLM                          ; 0756
                                  68              01  FB 0023F         CALLS   #1, CLI$PRESENT                   ;
                                  1B              50  E9 00242         BLBC    R0, 31$                           ;
                                                  5A  DD 00245         PUSHL   R10                               ; 0759
                                        FF30      C6  9F 00247         PUSHAB  SD_BIOLM                          ;
                                  6B              02  FB 0024B         CALLS   #2, CLI$GET_VALUE                 ;
                                                  10  DD 0024E         PUSHL   #16                               ; 0760
                                          3A      A9  9F 00250         PUSHAB  RECBUF+526                        ;
                       00000000V  00              02  FB 00253         CALLS   #2, GETVAL                        ;
                                  4A              50  E9 0025A  30$:   BLBC    R0, 34$                           ;
                                  67              01  D0 0025D         MOVL    #1, STATUS                        ; 0762
                                        FF40      C6  9F 00260  31$:   PUSHAB  SD_BYTLM                          ; 0765
                                  68              01  FB 00264         CALLS   #1, CLI$PRESENT                   ;
                                  1B              50  E9 00267         BLBC    R0, 33$                           ;
                                                  5A  DD 0026A         PUSHL   R10                               ; 0768
                                        FF40      C6  9F 0026C         PUSHAB  SD_BYTLM                          ;
                                  6B              02  FB 00270         CALLS   #2, CLI$GET_VALUE                 ;
                                                  20  DD 00273         PUSHL   #32                               ; 0769
                                          5C      A9  9F 00275         PUSHAB  RECBUF+560                        ;
                       00000000V  00              02  FB 00278         CALLS   #2, GETVAL                        ;
                                  67              50  E9 0027F  32$:   BLBC    R0, 37$                           ;
                                  67              01  D0 00282         MOVL    #1, STATUS                        ; 0771
                                        FF4C      C6  9F 00285  33$:   PUSHAB  SD_CLI                            ; 0774
                                  68              01  FB 00289         CALLS   #1, CLI$PRESENT                   ;
                                  38              50  E9 0028C         BLBC    R0, 36$                           ;
                                                  5A  DD 0028F         PUSHL   R10                               ; 0777
                                        FF4C      C6  9F 00291         PUSHAB  SD_CLI                            ;
                                  6B              02  FB 00295         CALLS   #2, CLI$GET_VALUE                 ;
                                                  01  DD 00298         PUSHL   #1                                ; 0778
                                                  20  DD 0029A         PUSHL   #32                               ;
                                        FF40      C9  9F 0029C         PUSHAB  RECBUF+276                        ;
                       00000000V  00              03  FB 002A0         CALLS   #3, GETSTRING                     ;
                                  66              50  E9 002A7  34$:   BLBC    R0, 39$                           ;
                                                  50  D4 002AA         CLRL    R0                                ; 0781
                                        FF60      C9  95 002AC         TSTB    RECBUF+308                        ;
                                                  02  13 002B0         BEQL    35$                               ;
                                                  50  D6 002B2         INCL    R0                                ;
00000000G  00           01                05      50  F0 002B4  35$:   INSV    R0, #5, #1, UAF$GL_CTLMSK         ;
                       00000000G  00              08  88 002BD         BISB2   #8, UAF$GL_CTLMSK                 ; 0782
                                  67              01  D0 002C4         MOVL    #1, STATUS                        ; 0783
                                        0160      C6  9F 002C7  36$:   PUSHAB  SD_CLITABLES                      ; 0786
                                  68              01  FB 002CB         CALLS   #1, CLI$PRESENT                   ;
                                  25              50  E9 002CE         BLBC    R0, 38$                           ;
                                                  5A  DD 002D1         PUSHL   R10                               ; 0789
                                        0160      C6  9F 002D3         PUSHAB  SD_CLITABLES                      ;
                                  6B              02  FB 002D7         CALLS   #2, CLI$GET_VALUE                 ;
                                                  01  DD 002DA         PUSHL   #1                                ; 0790
                                                  20  DD 002DC         PUSHL   #32                               ;
                                        FF60      C9  9F 002DE         PUSHAB  RECBUF+308                        ;
                       00000000V  00              03  FB 002E2         CALLS   #3, GETSTRING                     ;
                                  62              50  E9 002E9  37$:   BLBC    R0, 42$                           ;
                       00000000G  00              10  88 002EC         BISB2   #16, UAF$GL_CTLMSK                ; 0792
```

```
                  67              01  D0  002F3        MOVL    #1, STATUS                   ; 0793
                           FF5C   C6  9F  002F6  38$:  PUSHAB  SD_CPUTIME                   ; 0796
                  68              01  FB  002FA        CALLS   #1, CLI$PRESENT              ;
                  16              50  E9  002FD        BLBC    R0, 40$                      ;
                                  5A  DD  00300        PUSHL   R10                          ; 0799
                           FF5C   C6  9F  00302        PUSHAB  SD_CPUTIME                   ;
                  6B              02  FB  00306        CALLS   #2, CLI$GET_VALUE            ;
       00000000V  00              00  FB  00309        CALLS   #0, GETCPUTIM                ;
                  7E              50  E9  00310  39$:  BLBC    R0, 45$                      ; 0800
                  67              01  D0  00313        MOVL    #1, STATUS                   ; 0802
                           01A8   C6  9F  00316  40$:  PUSHAB  SD_DEFPRIVILEGES             ; 0805
                  68              01  FB  0031A        CALLS   #1, CLI$PRESENT              ;
                  14              50  E9  0031D        BLBC    R0, 41$                      ;
                             D0   A9  9F  00320        PUSHAB  RECBUF+420                   ; 0808
                           01A8   C6  9F  00323        PUSHAB  SD_DEFPRIVILEGES             ;
       00000000V  00              02  FB  00327        CALLS   #2, GETPRIV                  ;
                  60              50  E9  0032E        BLBC    R0, 45$                      ;
                  67              01  D0  00331        MOVL    #1, STATUS                   ; 0810
                           FF6C   C6  9F  00334  41$:  PUSHAB  SD_DEVICE                    ; 0813
                  68              01  FB  00338        CALLS   #1, CLI$PRESENT              ;
                  16              50  E9  0033B        BLBC    R0, 43$                      ;
                                  5A  DD  0033E        PUSHL   R10                          ; 0816
                           FF6C   C6  9F  00340        PUSHAB  SD_DEVICE                    ;
                  6B              02  FB  00344        CALLS   #2, CLI$GET_VALUE            ;
       00000000V  00              00  FB  00347        CALLS   #0, GETDEVICE                ; 0817
                  63              50  E9  0034E  42$:  BLBC    R0, 47$                      ;
                  67              01  D0  00351        MOVL    #1, STATUS                   ; 0819
                           FF7C   C6  9F  00354  43$:  PUSHAB  SD_DIOLM                     ; 0822
                  68              01  FB  00358        CALLS   #1, CLI$PRESENT              ;
                  1B              50  E9  0035B        BLBC    R0, 44$                      ;
                                  5A  DD  0035E        PUSHL   R10                          ; 0825
                           FF7C   C6  9F  00360        PUSHAB  SD_DIOLM                     ;
                  6B              02  FB  00364        CALLS   #2, CLI$GET_VALUE            ;
                                  10  DD  00367        PUSHL   #16                          ; 0826
                             3C   A9  9F  00369        PUSHAB  RECBUF+528                   ;
       00000000V  00              02  FB  0036C        CALLS   #2, GETVAL                   ;
                  3E              50  E9  00373        BLBC    R0, 47$                      ;
                  67              01  D0  00376        MOVL    #1, STATUS                   ; 0828
                             90   A6  9F  00379  44$:  PUSHAB  SD_DIRECTORY                 ; 0831
                  68              01  FB  0037C        CALLS   #1, CLI$PRESENT              ;
                  15              50  E9  0037F        BLBC    R0, 46$                      ;
                                  5A  DD  00382        PUSHL   R10                          ; 0834
                             90   A6  9F  00384        PUSHAB  SD_DIRECTORY                 ;
                  6B              02  FB  00387        CALLS   #2, CLI$GET_VALUE            ;
       00000000V  00              00  FB  0038A        CALLS   #0, GETDIRECTORY             ; 0835
                  20              50  E9  00391  45$:  BLBC    R0, 47$                      ;
                  67              01  D0  00394        MOVL    #1, STATUS                   ; 0837
                             A0   A6  9F  00397  46$:  PUSHAB  SD_ENQLM                     ; 0840
                  68              01  FB  0039A        CALLS   #1, CLI$PRESENT              ;
                  1A              50  E9  0039D        BLBC    R0, 48$                      ;
                                  5A  DD  003A0        PUSHL   R10                          ; 0843
                             A0   A6  9F  003A2        PUSHAB  SD_ENQLM                     ;
                  6B              02  FB  003A5        CALLS   #2, CLI$GET_VALUE            ;
                                  10  DD  003A8        PUSHL   #16                          ; 0844
                             42   A9  9F  003AA        PUSHAB  RECBUF+534                   ;
       00000000V  00              02  FB  003AD        CALLS   #2, GETVAL                   ;
                  77              50  E9  003B4  47$:  BLBC    R0, 53$                      ;
```

```
                                67             01  D0 003B7         MOVL    #1, STATUS                              ; 0846
                                        01CC   C6  9F 003BA 48$:    PUSHAB  SD_EXPIRATION                           ; 0849
                                68             01  FB 003BE         CALLS   #1, CLI$PRESENT                         ;
                          04    A7             50  D0 003C1         MOVL    R0, CLI_STATUS                          ;
                   00000000G    8F             50  D1 003C5         CMPL    R0, #CLI$_NEGATED                       ; 0850
                                09             09  12 003CC         BNEQ    49$                                     ;
      08            00          6E             00  2C 003CE         MOVC5   #0, (SP), #0, #8, RECBUF+364            ; 0853
                                          98   A9     003D3                                                         ;
                                               21  11 003D5         BRB     50$                                     ; 0854
                                21             50  E9 003D7 49$:    BLBC    R0, 51$                                 ; 0857
                                               5A  DD 003DA         PUSHL   R10                                     ; 0860
                                        01CC   C6  9F 003DC         PUSHAB  SD_EXPIRATION                           ;
                                6B             02  FB 003E0         CALLS   #2, CLI$GET_VALUE                       ;
                                          98   A9  9F 003E3         PUSHAB  RECBUF+364                              ; 0861
                                               5A  DD 003E6         PUSHL   R10                                     ;
                   00000000G    00             02  FB 003E8         CALLS   #2, LIB$CVT_TIME                        ;
                                67             50  D0 003EF         MOVL    R0, STATUS                              ;
                                03             50  E8 003F2         BLBS    R0, 50$                                 ;
                                       02D4    31     003F5         BRW     91$                                     ;
                                67             01  D0 003F8 50$:    MOVL    #1, STATUS                              ; 0867
                                          B0   A6  9F 003FB 51$:    PUSHAB  SD_FILLM                                ; 0870
                                68             01  FB 003FE         CALLS   #1, CLI$PRESENT                         ;
                                1A             50  E9 00401         BLBC    R0, 52$                                 ;
                                               5A  DD 00404         PUSHL   R10                                     ; 0873
                                          B0   A6  9F 00406         PUSHAB  SD_FILLM                                ;
                                6B             02  FB 00409         CALLS   #2, CLI$GET_VALUE                       ;
                                               10  DD 0040C         PUSHL   #16                                     ; 0874
                                          44   A9  9F 0040E         PUSHAB  RECBUF+536                              ;
                   00000000V    00             02  FB 00411         CALLS   #2, GETVAL                              ;
                                60             50  E9 00418         BLBC    R0, 56$                                 ;
                                67             01  D0 0041B         MOVL    #1, STATUS                              ; 0876
                                          C0   A6  9F 0041E 52$:    PUSHAB  SD_FLAGS                                ; 0879
                                68             01  FB 00421         CALLS   #1, CLI$PRESENT                         ;
                                0D             50  E9 00424         BLBC    R0, 54$                                 ;
                   00000000V    00             00  FB 00427         CALLS   #0, GETFLAGS                            ; 0882
                                6D             50  E9 0042E 53$:    BLBC    R0, 58$                                 ;
                                67             01  D0 00431         MOVL    #1, STATUS                              ; 0884
                                        0170   C6  9F 00434 54$:    PUSHAB  SD_JTQUOTA                              ; 0887
                                68             01  FB 00438         CALLS   #1, CLI$PRESENT                         ;
                                1B             50  E9 0043B         BLBC    R0, 55$                                 ;
                                               5A  DD 0043E         PUSHL   R10                                     ; 0890
                                        0170   C6  9F 00440         PUSHAB  SD_JTQUOTA                              ;
                                6B             02  FB 00444         CALLS   #2, CLI$GET_VALUE                       ;
                                               20  DD 00447         PUSHL   #32                                     ; 0891
                                          64   A9  9F 00449         PUSHAB  RECBUF+568                              ;
                   00000000V    00             02  FB 0044C         CALLS   #2, GETVAL                              ;
                                6D             50  E9 00453         BLBC    R0, 60$                                 ;
                                67             01  D0 00456         MOVL    #1, STATUS                              ; 0893
                                          D0   A6  9F 00459 55$:    PUSHAB  SD_LGICMD                               ; 0896
                                68             01  FB 0045C         CALLS   #1, CLI$PRESENT                         ;
                                1F             50  E9 0045F         BLBC    R0, 57$                                 ;
                                               5A  DD 00462         PUSHL   R10                                     ; 0899
                                          D0   A6  9F 00464         PUSHAB  SD_LGICMD                               ;
                                6B             02  FB 00467         CALLS   #2, CLI$GET_VALUE                       ;
                                               01  DD 0046A         PUSHL   #1                                      ; 0900
                                7E        40   8F  9A 0046C         MOVZBL  #64, -(SP)                              ;
                                        FF00   C9  9F 00470         PUSHAB  RECBUF+212                              ;
```

```
00000000V  00            03 FB 00474          CALLS   #3, GETSTRING              ;
           6A            50 E9 0047B  56$:    BLBC    R0, 62$                    ;
           67            01 D0 0047E          MOVL    #1, STATUS                 ; 0902
                   E0    A6 9F 00481  57$:    PUSHAB  SD_MAXJOBS                 ; 0905
           68            01 FB 00484          CALLS   #1, CLI$PRESENT            ;
           1A            50 E9 00487          BLBC    R0, 59$                    ;
                         5A DD 0048A          PUSHL   R10                        ; 0908
                   E0    A6 9F 0048C          PUSHAB  SD_MAXJOBS                 ;
           6B            02 FB 0048F          CALLS   #2, CLI$GET_VALUE          ;
                         10 DD 00492          PUSHL   #16                        ; 0909
                   32    A9 9F 00494          PUSHAB  RECBUF+518                 ;
00000000V  00            02 FB 00497          CALLS   #2, GETVAL                 ;
           6D            50 E9 0049E  58$:    BLBC    R0, 64$                    ;
           67            01 D0 004A1          MOVL    #1, STATUS                 ; 0911
                 0138    C6 9F 004A4  59$:    PUSHAB  SD_MAXACCTJOBS             ; 0914
           68            01 FB 004A8          CALLS   #1, CLI$PRESENT            ;
           1B            50 E9 004AB          BLBC    R0, 61$                    ;
                         5A DD 004AE          PUSHL   R10                        ; 0917
                 0138    C6 9F 004B0          PUSHAB  SD_MAXACCTJOBS             ;
           6B            02 FB 004B4          CALLS   #2, CLI$GET_VALUE          ;
                         10 DD 004B7          PUSHL   #16                        ; 0918
                   34    A9 9F 004B9          PUSHAB  RECBUF+520                 ;
00000000V  00            02 FB 004BC          CALLS   #2, GETVAL                 ;
           6B            50 E9 004C3  60$:    BLBC    R0, 66$                    ;
           67            01 D0 004C6          MOVL    #1, STATUS                 ; 0920
                 014C    C6 9F 004C9  61$:    PUSHAB  SD_MAXDETACH               ; 0923
           68            01 FB 004CD          CALLS   #1, CLI$PRESENT            ;
           1B            50 E9 004D0          BLBC    R0, 63$                    ;
                         5A DD 004D3          PUSHL   R10                        ; 0926
                 014C    C6 9F 004D5          PUSHAB  SD_MAXDETACH               ;
           6B            02 FB 004D9          CALLS   #2, CLI$GET_VALUE          ;
                         10 DD 004DC          PUSHL   #16                        ; 0927
                   36    A9 9F 004DE          PUSHAB  RECBUF+522                 ;
00000000V  00            02 FB 004E1          CALLS   #2, GETVAL                 ;
           69            50 E9 004E8  62$:    BLBC    R0, 68$                    ;
           67            01 D0 004EB          MOVL    #1, STATUS                 ; 0929
                   F0    A6 9F 004EE  63$:    PUSHAB  SD_OWNER                   ; 0932
           68            01 FB 004F1          CALLS   #1, CLI$PRESENT            ;
           1D            50 E9 004F4          BLBC    R0, 65$                    ;
                         5A DD 004F7          PUSHL   R10                        ; 0935
                   F0    A6 9F 004F9          PUSHAB  SD_OWNER                   ;
           6B            02 FB 004FC          CALLS   #2, CLI$GET_VALUE          ;
                         01 DD 004FF          PUSHL   #1                         ; 0936
                         20 DD 00501          PUSHL   #32                        ;
                 FE80    C9 9F 00503          PUSHAB  RECBUF+84                  ;
00000000V  00            03 FB 00507          CALLS   #3, GETSTRING              ;
           7B            50 E9 0050E  64$:    BLBC    R0, 71$                    ;
           67            01 D0 00511          MOVL    #1, STATUS                 ; 0938
                   10    A6 9F 00514  65$:    PUSHAB  SD_PBYTLM                  ; 0941
           68            01 FB 00517          CALLS   #1, CLI$PRESENT            ;
           1A            50 E9 0051A          BLBC    R0, 67$                    ;
                         5A DD 0051D          PUSHL   R10                        ; 0944
                   10    A6 9F 0051F          PUSHAB  SD_PBYTLM                  ;
           6B            02 FB 00522          CALLS   #2, CLI$GET_VALUE          ;
                         20 DD 00525          PUSHL   #32                        ; 0945
                   60    A9 9F 00527          PUSHAB  RECBUF+564                 ;
00000000V  00            02 FB 0052A          CALLS   #2, GETVAL                 ;
```

```
             75          50  E9 00531 66$:    BLBC    R0, 73$                              ;
             67          01  D0 00534         MOVL    #1, STATUS                           ; 0947
                   34    A6  9F 00537 67$:    PUSHAB  SD_PGFLQUOTA                         ; 0950
             68          01  FB 0053A         CALLS   #1, CLI$PRESENT                      ;
             1A          50  E9 0053D         BLBC    R0, 69$                              ;
                         5A  DD 00540         PUSHL   R10                                  ; 0953
                   34    A6  9F 00542         PUSHAB  SD_PGFLQUOTA                         ;
             6B          02  FB 00545         CALLS   #2, CLI$GET_VALUE                    ;
                         18  DD 00548         PUSHL   #24                                  ; 0954
                   54    A9  9F 0054A         PUSHAB  RECBUF+552                           ;
  00000000V  00          02  FB 0054D         CALLS   #2, GETVAL                           ;
             6D          50  E9 00554 68$:    BLBC    R0, 75$                              ;
             67          01  D0 00557         MOVL    #1, STATUS                           ; 0956
                   48    A6  9F 0055A 69$:    PUSHAB  SD_P_RESTRICT                        ; 0959
             68          01  FB 0055D         CALLS   #1, CLI$PRESENT                      ;
             10          50  E9 00560         BLBC    R0, 70$                              ;
                   48    A6  9F 00563         PUSHAB  SD_P_RESTRICT                        ; 0962
  00000000V  00          01  FB 00566         CALLS   #1, GETRESTRICT                      ;
             77          50  E9 0056D         BLBC    R0, 77$                              ;
             67          01  D0 00570         MOVL    #1, STATUS                           ; 0964
                 00A0    C6  9F 00573 70$:    PUSHAB  SD_S_RESTRICT                        ; 0967
             68          01  FB 00577         CALLS   #1, CLI$PRESENT                      ;
             15          50  E9 0057A         BLBC    R0, 72$                              ;
         4C  A7          01  D0 0057D         MOVL    #1, SECONDARY                        ; 0970
                 00A0    C6  9F 00581         PUSHAB  SD_S_RESTRICT                        ; 0971
  00000000V  00          01  FB 00585         CALLS   #1, GETRESTRICT                      ;
             6F          50  E9 0058C 71$:    BLBC    R0, 79$                              ;
             67          01  D0 0058F         MOVL    #1, STATUS                           ; 0973
                   20    A6  9F 00592 72$:    PUSHAB  SD_PFLAGS                            ; 0976
             68          01  FB 00595         CALLS   #1, CLI$PRESENT                      ;
             14          50  E9 00598         BLBC    R0, 74$                              ;
         48  A7          01  D0 0059B         MOVL    #1, PRIMARY                          ; 0979
                   20    A6  9F 0059F         PUSHAB  SD_PFLAGS                            ; 0980
  00000000V  00          01  FB 005A2         CALLS   #1, GETPFLAGS                        ;
             75          50  E9 005A9 73$:    BLBC    R0, 81$                              ;
             67          01  D0 005AC         MOVL    #1, STATUS                           ; 0982
                 008C    C6  9F 005AF 74$:    PUSHAB  SD_SFLAGS                            ; 0985
             68          01  FB 005B3         CALLS   #1, CLI$PRESENT                      ;
             11          50  E9 005B6         BLBC    R0, 76$                              ;
                 008C    C6  9F 005B9         PUSHAB  SD_SFLAGS                            ; 0988
  00000000V  00          01  FB 005BD         CALLS   #1, GETPFLAGS                        ;
             76          50  E9 005C4 75$:    BLBC    R0, 83$                              ;
             67          01  D0 005C7         MOVL    #1, STATUS                           ; 0990
                   58    A6  9F 005CA 76$:    PUSHAB  SD_PRCLM                             ; 0993
             68          01  FB 005CD         CALLS   #1, CLI$PRESENT                      ;
             1A          50  E9 005D0         BLBC    R0, 78$                              ;
                         5A  DD 005D3         PUSHL   R10                                  ; 0996
                   58    A6  9F 005D5         PUSHAB  SD_PRCLM                             ;
             6B          02  FB 005D8         CALLS   #2, CLI$GET_VALUE                    ;
                         10  DD 005DB         PUSHL   #16                                  ; 0997
                   38    A9  9F 005DD         PUSHAB  RECBUF+524                           ;
  00000000V  00          02  FB 005E0         CALLS   #2, GETVAL                           ;
             53          50  E9 005E7 77$:    BLBC    R0, 83$                              ;
             67          01  D0 005EA         MOVL    #1, STATUS                           ; 0999
                 0124    C6  9F 005ED 78$:    PUSHAB  SD_PRIMEDAYS                         ; 1002
             68          01  FB 005F1         CALLS   #1, CLI$PRESENT                      ;
             0D          50  E9 005F4         BLBC    R0, 80$                              ;
```

```
                        00000000V 00         00 FB 005F7      CALLS   #0, GETPRIMEDAYS               ; 1005
                                  3C         50 E9 005FE 79$: BLBC    R0, 83$                        ;
                                  67         01 D0 00601      MOVL    #1, STATUS                     ; 1007
                                          68 A6 9F 00604 80$: PUSHAB  SD_PRIORITY                    ; 1010
                                  68         01 FB 00607      CALLS   #1, CLI$PRESENT                ;
                                  1A         50 E9 0060A      BLBC    R0, 82$                        ;
                                             5A DD 0060D      PUSHL   R10                            ; 1013
                                          68 A6 9F 0060F      PUSHAB  SD_PRIORITY                    ;
                                  6B         02 FB 00612      CALLS   #2, CLI$GET_VALUE              ;
                                             08 DD 00615      PUSHL   #8                             ; 1014
                                          30 A9 9F 00617      PUSHAB  RECBUF+516                     ;
                        00000000V 00         02 FB 0061A      CALLS   #2, GETVAL                     ;
                                  19         50 E9 00621 81$: BLBC    R0, 83$                        ;
                                  67         01 D0 00624      MOVL    #1, STATUS                     ; 1016
                                          7C A6 9F 00627 82$: PUSHAB  SD_PRIVILEGES                  ; 1019
                                  68         01 FB 0062A      CALLS   #1, CLI$PRESENT                ;
                                  16         50 E9 0062D      BLBC    R0, 85$                        ;
                                          C8 A9 9F 00630      PUSHAB  RECBUF+412                     ; 1022
                                          7C A6 9F 00633      PUSHAB  SD_PRIVILEGES                  ;
                        00000000V 00         02 FB 00636      CALLS   #2, GETPRIV                    ;
                                  03         50 E8 0063D 83$: BLBS    R0, 84$                        ;
                                           01E0 31 00640      BRW     107$                           ;
                                  67         01 D0 00643 84$: MOVL    #1, STATUS                     ; 1024
                                        03AC C6 9F 00646 85$: PUSHAB  P.AFD                          ; 1027
                                  68         01 FB 0064A      CALLS   #1, CLI$PRESENT                ;
                               04 A7         50 D0 0064D      MOVL    R0, CLI_STATUS                 ;
                        00000000G 8F         50 D1 00651      CMPL    R0, #CLI$_NEGATED              ; 1028
                                             08 12 00658      BNEQ    86$                            ;
                                  50      A8 A9 9E 0065A      MOVAB   RECBUF+380, TIME               ; 1031
                                             60 7C 0065E      CLRQ    (TIME)                         ; 1032
                                             0E 11 00660      BRB     87$                            ; 1033
                                  12         50 E9 00662 86$: BLBC    R0, 88$                        ; 1036
                                  50      A8 A9 9E 00665      MOVAB   RECBUF+380, TIME               ; 1039
                                  60         01 CE 00669      MNEGL   #1, (TIME)                     ; 1040
                               04 A0         01 CE 0066C      MNEGL   #1, 4(TIME)                    ;
                               01 A9         02 8A 00670 87$: BICB2   #2, RECBUF+469                 ; 1041
                                  67         01 D0 00674      MOVL    #1, STATUS                     ; 1042
                                        0214 C6 9F 00677 88$: PUSHAB  SD_PWDLIFETIME                 ; 1045
                                  68         01 FB 0067B      CALLS   #1, CLI$PRESENT                ;
                               04 A7         50 D0 0067E      MOVL    R0, CLI_STATUS                 ;
                        00000000G 8F         50 D1 00682      CMPL    R0, #CLI$_NEGATED              ; 1046
                                             09 12 00689      BNEQ    89$                            ;
08          00                    6E         00 2C 0068B      MOVC5   #0, (SP), #0, #8, RECBUF+372   ; 1049
                                          A0 A9    00690                                             ;
                                             44 11 00692      BRB     93$                            ; 1050
                                  48         50 E9 00694 89$: BLBC    R0, 94$                        ; 1054
                                             5A DD 00697      PUSHL   R10                            ; 1057
                                        0214 C6 9F 00699      PUSHAB  SD_PWDLIFETIME                 ;
                                  6B         02 FB 0069D      CALLS   #2, CLI$GET_VALUE              ;
                                  50      04 AA D0 006A0      MOVL    TOKENDSC+4, R0                 ; 1058
04          00                    60         6A 2D 006A4      CMPC5   TOKENDSC, (R0), #0, #4, P.AFF  ;
                                        03B4 C6    006A9                                             ;
                                             0C 12 006AC      BNEQ    90$                            ;
08          00                    6E         00 2C 006AE      MOVC5   #0, (SP), #0, #8, RECBUF+372   ; 1061
                                          A0 A9    006B3                                             ;
                                  67         01 D0 006B5      MOVL    #1, STATUS                     ; 1062
                                             1E 11 006B8      BRB     93$                            ; 1058
```

```
                     A0   A9  9F 006BA 90$:    PUSHAB  RECBUF+372            ; 1065
                          5A  DD 006BD         PUSHL   R10                   ;
00000000G  00             02  FB 006BF         CALLS   #2, LIB$CVT_DTIME     ;
           67             50  D0 006C6         MOVL    R0, STATUS            ;
           0C             50  E8 006C9         BLBS    R0, 93$               ;
                          67  DD 006CC 91$:    PUSHL   STATUS                ; 1068
00000000G  00             01  FB 006CE         CALLS   #1, LIB$SIGNAL        ;
                        014B  31 006D5 92$:    BRW     107$                  ; 1069
       01  A9             02  8A 006D8 93$:    BICB2   #2, RECBUF+469        ; 1071
           67             01  D0 006DC         MOVL    #1, STATUS            ; 1072
                   0228   C6  9F 006DF 94$:    PUSHAB  SD_PWDMINIMUM         ; 1075
           68             01  FB 006E3         CALLS   #1, CLI$PRESENT       ;
           1B             50  E9 006E6         BLBC    R0, 95$               ;
                          5A  DD 006E9         PUSHL   R10                   ; 1078
                   0228   C6  9F 006EB         PUSHAB  SD_PWDMINIMUM         ;
           6B             02  FB 006EF         CALLS   #2, CLI$GET_VALUE     ;
                          08  DD 006F2         PUSHL   #8                    ; 1079
                     96   A9  9F 006F4         PUSHAB  RECBUF+362            ;
00000000V  00             02  FB 006F7         CALLS   #2, GETVAL            ;
           D4             50  E9 006FE         BLBC    R0, 92$               ;
           67             01  D0 00701         MOVL    #1, STATUS            ; 1081
                   023C   C6  9F 00704 95$:    PUSHAB  SD_QUEPRIORITY        ; 1084
           68             01  FB 00708         CALLS   #1, CLI$PRESENT       ;
           1B             50  E9 0070B         BLBC    R0, 96$               ;
                          5A  DD 0070E         PUSHL   R10                   ; 1087
                   023C   C6  9F 00710         PUSHAB  SD_QUEPRIORITY        ;
           6B             02  FB 00714         CALLS   #2, CLI$GET_VALUE     ;
                          08  DD 00717         PUSHL   #8                    ; 1088
                     31   A9  9F 00719         PUSHAB  RECBUF+517            ;
00000000V  00             02  FB 0071C         CALLS   #2, GETVAL            ;
           AF             50  E9 00723         BLBC    R0, 92$               ;
           67             01  D0 00726         MOVL    #1, STATUS            ; 1090
                   00B0   C6  9F 00729 96$:    PUSHAB  SD_SHRFILLM           ; 1093
           68             01  FB 0072D         CALLS   #1, CLI$PRESENT       ;
           1B             50  E9 00730         BLBC    R0, 97$               ;
                          5A  DD 00733         PUSHL   R10                   ; 1096
                   00B0   C6  9F 00735         PUSHAB  SD_SHRFILLM           ;
           6B             02  FB 00739         CALLS   #2, CLI$GET_VALUE     ;
                          10  DD 0073C         PUSHL   #16                   ; 1097
                     46   A9  9F 0073E         PUSHAB  RECBUF+538            ;
00000000V  00             02  FB 00741         CALLS   #2, GETVAL            ;
           8A             50  E9 00748         BLBC    R0, 92$               ;
           67             01  D0 0074B         MOVL    #1, STATUS            ; 1099
                   00C0   C6  9F 0074E 97$:    PUSHAB  SD_TQELM              ; 1102
           68             01  FB 00752         CALLS   #1, CLI$PRESENT       ;
           1B             50  E9 00755         BLBC    R0, 98$               ;
                          5A  DD 00758         PUSHL   R10                   ; 1105
                   00C0   C6  9F 0075A         PUSHAB  SD_TQELM              ;
           6B             02  FB 0075E         CALLS   #2, CLI$GET_VALUE     ;
                          10  DD 00761         PUSHL   #16                   ; 1106
                     3E   A9  9F 00763         PUSHAB  RECBUF+530            ;
00000000V  00             02  FB 00766         CALLS   #2, GETVAL            ;
           6A             50  E9 0076D         BLBC    R0, 101$              ;
           67             01  D0 00770         MOVL    #1, STATUS            ; 1108
                   00CC   C6  9F 00773 98$:    PUSHAB  SD_UIC                ; 1111
           68             01  FB 00777         CALLS   #1, CLI$PRESENT       ;
           16             50  E9 0077A         BLBC    R0, 99$               ;
```

```
                                   5A  DD 0077D          PUSHL   R10                                  ; 1114
                        00CC       C6  9F 0077F          PUSHAB  SD_UIC                               ;
                   6B              02  FB 00783          CALLS   #2, CLI$GET_VALUE                    ;
         00000000V 00              00  FB 00786          CALLS   #0, GETUIC                           ; 1115
                   72              50  E9 0078D          BLBC    R0, 103$                             ;
                   67              01  D0 00790          MOVL    #1, STATUS                           ; 1117
                        00E0       C6  9F 00793 99$:     PUSHAB  SD_WSDEFAULT                         ; 1120
                   68              01  FB 00797          CALLS   #1, CLI$PRESENT                      ;
                   1E              50  E9 0079A          BLBC    R0, 100$                             ;
                                   5A  DD 0079D          PUSHL   R10                                  ; 1123
                        00E0       C6  9F 0079F          PUSHAB  SD_WSDEFAULT                         ;
                   6B              02  FB 007A3          CALLS   #2, CLI$GET_VALUE                    ;
                                   11  DD 007A6          PUSHL   #17                                  ; 1124
                          4C       A9  9F 007A8          PUSHAB  RECBUF+544                           ;
         00000000V 00              02  FB 007AB          CALLS   #2, GETVAL                           ;
                   6E              50  E9 007B2          BLBC    R0, 107$                             ;
                          4E       A9  B4 007B5          CLRW    RECBUF+546                           ; 1126
                   67              01  D0 007B8          MOVL    #1, STATUS                           ; 1127
                        00F0       C6  9F 007BB 100$:    PUSHAB  SD_WSEXTENT                          ; 1130
                   68              01  FB 007BF          CALLS   #1, CLI$PRESENT                      ;
                   1E              50  E9 007C2          BLBC    R0, 102$                             ;
                                   5A  DD 007C5          PUSHL   R10                                  ; 1133
                        00F0       C6  9F 007C7          PUSHAB  SD_WSEXTENT                          ;
                   6B              02  FB 007CB          CALLS   #2, CLI$GET_VALUE                    ;
                                   11  DD 007CE          PUSHL   #17                                  ; 1134
                          50       A9  9F 007D0          PUSHAB  RECBUF+548                           ;
         00000000V 00              02  FB 007D3          CALLS   #2, GETVAL                           ;
                   46              50  E9 007DA 101$:    BLBC    R0, 107$                             ;
                          52       A9  B4 007DD          CLRW    RECBUF+550                           ; 1136
                   67              01  D0 007E0          MOVL    #1, STATUS                           ; 1137
                        0100       C6  9F 007E3 102$:    PUSHAB  SD_WSQUOTA                           ; 1140
                   68              01  FB 007E7          CALLS   #1, CLI$PRESENT                      ;
                   1E              50  E9 007EA          BLBC    R0, 104$                             ;
                                   5A  DD 007ED          PUSHL   R10                                  ; 1143
                        0100       C6  9F 007EF          PUSHAB  SD_WSQUOTA                           ;
                   6B              02  FB 007F3          CALLS   #2, CLI$GET_VALUE                    ;
                                   11  DD 007F6          PUSHL   #17                                  ; 1144
                          48       A9  9F 007F8          PUSHAB  RECBUF+540                           ;
         00000000V 00              02  FB 007FB          CALLS   #2, GETVAL                           ;
                   1E              50  E9 00802 103$:    BLBC    R0, 107$                             ;
                          4A       A9  B4 00805          CLRW    RECBUF+542                           ; 1146
                   67              01  D0 00808          MOVL    #1, STATUS                           ; 1147
      08 00000000G 00              02  E0 0080B 104$:    BBS     #2, UAF$GL_CTLMSK, 105$              ; 1150
      04 00000000G 00              01  E1 00813          BBC     #1, UAF$GL_CTLMSK, 106$              ;
                   50              01  D0 0081B 105$:    MOVL    #1, R0                               ; 1152
                                       04 0081E          RET                                          ;
                   50              67  D0 0081F 106$:    MOVL    STATUS, R0                           ; 1154
                                       04 00822          RET                                          ;
                                   50  D4 00823 107$:    CLRL    R0                                   ; 1155
                                       04 00825          RET                                          ;

; Routine Size:  2086 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %sbttl 'getaccess - get hourly and daily access control flags'
1 routine getaccess (qual_name, access_type, qual_status) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Parse the value list for hourly login restrictions on the
2 !       selected access modes.
2 !
2 ! INPUTS:
2 !
2 !       qual_name - descriptor of qualifier name to use
2 !       access_type - bitvector representing the access modes to be affected
2 !       qual_status - status from cli$present for qualifier
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       none
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> all keywords and parameters were specified correctly
2 !
2 ! SIDE EFFECTS:
2 !
2 !       none
2 !--
2
2 map
2       access_type     : bitvector;
2
2 STRUCTURE
2     threebytevector [i; n, ext=0] =
2       [n*3]
2       (threebytevector+i*3)<0, 24, ext>;
2
2 $ASSUME ($BYTEOFFSET (uaf$b_network_access_s), EQL, $BYTEOFFSET (uaf$b_network_access_p)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_batch_access_p), EQL, $BYTEOFFSET (uaf$b_network_access_s)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_batch_access_s), EQL, $BYTEOFFSET (uaf$b_batch_access_p)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_local_access_p), EQL, $BYTEOFFSET (uaf$b_batch_access_s)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_local_access_s), EQL, $BYTEOFFSET (uaf$b_local_access_p)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_dialup_access_p), EQL, $BYTEOFFSET (uaf$b_local_access_s)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_dialup_access_s), EQL, $BYTEOFFSET (uaf$b_dialup_access_p)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_remote_access_p), EQL, $BYTEOFFSET (uaf$b_dialup_access_s)+3);
2 $ASSUME ($BYTEOFFSET (uaf$b_remote_access_s), EQL, $BYTEOFFSET (uaf$b_remote_access_p)+3);
2
2 bind
```

```
2           access_vector   = recbuf[uaf$b_network_access_p]
2                           : threebytevector;
2
2 !
2 ! Initialize output values.
2 !
2 primary_access = -1;
2 secondary_access = -1;
2 if not .qual_status then no_flag = true;
2
2 !
2 ! Fetch value list items and parse them.
2 !
2 while true
2 do
3     begin
3     if not cli$get_value (.qual_name, tokendsc)
3     then exitloop;
3
3     tparse_block[tpa$l_stringcnt] = .tokenlen;
3     tparse_block[tpa$l_stringptr] = .tokenptr;
3     if not lib$tparse (tparse_block, access_states, access_keys)
3     then return LIB$SIGNAL(UAF$_BADVALUE, 2, .tokenlen, .tokenptr);
2     end;
2
2 !
2 ! Default omitted values.
2 !
2 if not .primary and not .secondary
2 then secondary_access = .primary_access;
2 if .primary and not .secondary
2 then secondary_access = 0;
2 if .secondary and not .primary and .primary_access eql -1
2 then primary_access = 0;
2
2 if not .no_flag
2 then
3     begin
3     primary_access = not .primary_access;
3     secondary_access = not .secondary_access;
2     end;
2
2 !
2 ! Write the specified fields in the UAF record.
2 !
2 incr j from 0 to 4
2 do
3     begin
3     if .access_type[.j]
3     then
4         begin
4         access_vector [.j*2+0] = .primary_access;
4         access_vector [.j*2+1] = .secondary_access;
3         end;
2     end;
2
2 true
```

```
; 1178          1270  1 end;



                                                        003C 00000  GETACCESS:
                                                                            .WORD   Save R2,R3,R4,R5                              ; 1157
                                        55 00000000G  00  9E 00002          MOVAB   TOKENPTR, R5                                  ;
                                        54 00000000G  00  9E 00009          MOVAB   TOKENLEN, R4                                  ;
                                        53 00000000'  00  9E 00010          MOVAB   PRIMARY_ACCESS, R3                            ;
                                        63            01  CE 00017          MNEGL   #1, PRIMARY_ACCESS                            ; 1219
                                   04   A3            01  CE 0001A          MNEGL   #1, SECONDARY_ACCESS                          ; 1220
                                        04        0C  AC  E8 0001E          BLBS    QUAL_STATUS, 1$                               ; 1221
                                   08   A3            01  D0 00022          MOVL    #1, NO_FLAG                                   ;
                                           00000000G  00  9F 00026  1$:     PUSHAB  TOKENDSC                                      ; 1229
                                                  04  AC  DD 0002C          PUSHL   QUAL_NAME                                     ;
                             00000000G  00            02  FB 0002F          CALLS   #2, CLI$GET_VALUE                             ;
                                        36            50  E9 00036          BLBC    R0, 2$                                        ;
                                   D4   A3            64  3C 00039          MOVZWL  TOKENLEN, TPARSE_BLOCK+8                      ; 1232
                                   D8   A3            65  D0 0003D          MOVL    TOKENPTR, TPARSE_BLOCK+12                     ; 1233
                                           00000000'  00  9F 00041          PUSHAB  ACCESS_KEYS                                   ; 1234
                                           00000000'  00  9F 00047          PUSHAB  ACCESS_STATES                                 ;
                                                  CC  A3  9F 0004D          PUSHAB  TPARSE_BLOCK                                  ;
                             00000000G  00            03  FB 00050          CALLS   #3, LIB$TPARSE                                ;
                                        CC            50  E8 00057          BLBS    R0, 1$                                        ;
                                                      65  DD 0005A          PUSHL   TOKENPTR                                      ; 1235
                                        7E            64  3C 0005C          MOVZWL  TOKENLEN, -(SP)                               ;
                                                      02  DD 0005F          PUSHL   #2                                            ;
                                           00000000G  8F  DD 00061          PUSHL   #UAF$_BADVALUE                                ;
                             00000000G  00            04  FB 00067          CALLS   #4, LIB$SIGNAL                                ;
                                                          04 0006E          RET                                                   ;
                                        50        0C  A3  D0 0006F  2$:     MOVL    PRIMARY, R0                                   ; 1241
                                        0B            50  E8 00073          BLBS    R0, 4$                                        ;
                                        04        10  A3  E8 00076          BLBS    SECONDARY, 3$                                 ;
                                   04   A3            63  D0 0007A          MOVL    PRIMARY_ACCESS, SECONDARY_ACCESS              ; 1242
                                        07            50  E9 0007E  3$:     BLBC    R0, 5$                                        ; 1243
                                        07        10  A3  E8 00081  4$:     BLBS    SECONDARY, 6$                                 ;
                                                  04  A3  D4 00085          CLRL    SECONDARY_ACCESS                              ; 1244
                                        0E        10  A3  E9 00088  5$:     BLBC    SECONDARY, 7$                                 ; 1245
                                        0B            50  E8 0008C  6$:     BLBS    R0, 7$                                        ;
                             FFFFFFFF   8F            63  D1 0008F          CMPL    PRIMARY_ACCESS, #-1                           ;
                                                      02  12 00096          BNEQ    7$                                            ;
                                                      63  D4 00098          CLRL    PRIMARY_ACCESS                                ; 1246
                                        08        08  A3  E8 0009A  7$:     BLBS    NO_FLAG, 8$                                   ; 1248
                                        63            63  D2 0009E          MCOML   PRIMARY_ACCESS, PRIMARY_ACCESS                ; 1251
                                   04   A3        04  A3  D2 000A1          MCOML   SECONDARY_ACCESS, SECONDARY_ACCESS            ; 1252
                                                      50  D4 000A6  8$:     CLRL    J                                             ; 1258
                          21       08   AC            50  E1 000A8  9$:     BBC     J, ACCESS_TYPE, 10$                           ; 1261
                          52            50            01  78 000AD          ASHL    #1, J, R2                                     ; 1264
                          51            52            03  C5 000B1          MULL3   #3, R2, R1                                    ;
00000000G0041             18            00            63  F0 000B5          INSV    PRIMARY_ACCESS, #0, #24, ACCESS_VECTOR[R1]    ;
                          51            52            03  C5 000BF          MULL3   #3, R2, R1                                    ; 1265
00000000G0041             18            00        04  A3  F0 000C3          INSV    SECONDARY_ACCESS, #0, #24, ACCESS_VECTOR+3-   ;
                                                                                    [R1]

                          D6            50            04  F3 000CE  10$:    AOBLEQ  #4, J, 9$                                     ; 1258
                                        50            01  D0 000D2          MOVL    #1, R0                                        ; 1270
```

```
                                    04 000D5          RET                                          ;

; Routine Size:  214 bytes,    Routine Base:  $CODE$ + 0826
```

```
; 1180          1271  1 %sbttl 'check_primary - check presence of PRIMARY keyword'
; 1181          1272  1 routine check_primary =
; 1182          1273  2 begin
; 1183          1274  2 not .primary and not .secondary
; 1184          1275  1 end;



                                              0000 00000 CHECK_PRIMARY:
                                                                   .WORD   Save nothing                                          ; 1272
                   50 00000000'  00 00000000'  00  C9 00002         BISL3   SECONDARY, PRIMARY, R0                                ; 1274
                          50                   50  D2 0000E         MCOML   R0, R0                                                ; 1273
                                                   04 00011         RET                                                           ; 1275

; Routine Size:  18 bytes,    Routine Base:  $CODE$ + 08FC
```

```
; 1186          1276  1 %sbttl 'check_secondary - check presence of SECONDARY keyword'
; 1187          1277  1 routine check_secondary =
; 1188          1278  2 begin
; 1189          1279  2 not .secondary
; 1190          1280  1 end;



                                                      0000 00000 CHECK_SECONDARY:
                                                                       .WORD   Save nothing                                          ; 1277
                                        50 00000000'  00  D2 00002     MCOML   SECONDARY, R0                                         ; 1278
                                                          04 00009     RET                                                           ; 1280

; Routine Size:  10 bytes,    Routine Base:  $CODE$ + 090E
```

```
1 %sbttl 'set_range - set hourly restriction bits'
1 global routine set_range  =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Set or clear appropriate hourly login restriction bits
2 !
2 ! INPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> successful completion
2 !       false -> error in flag name
2 !--
2
2 builtin
2       ap;
2
2 map
2       ap              : ref block [,byte];
2
2 local
2       address,
2       width,
2       leftmost,
2       wrap;
2
2 !
2 ! Point to the flags longword to use.
2 !
2 if .secondary
2 then address = secondary_access
2 else address = primary_access;
2
2 !
2 ! Get the ending bit - note that this is the same as the starting
2 ! bit for a single hour.
2 !
2 left_bit = .ap[tpa$l_number];
2
2 !
2 ! Make sure hour is not out of range
2 !
2 if .left_bit  gtru 23
2 or .right_bit gtru 23
```

```
; 1249        1338  2 then
; 1250        1339  2     return false;
; 1251        1340  2
; 1252        1341  2 !
; 1253        1342  2 ! Clear out the default field value if this is the first real value.
; 1254        1343  2 !
; 1255        1344  2 if ..address eql -1
; 1256        1345  2 then .address = 0;
; 1257        1346  2
; 1258        1347  2 wrap = false;
; 1259        1348  2 leftmost = .left_bit;
; 1260        1349  2
; 1261        1350  2 !
; 1262        1351  2 ! If the starting bit is greater than the ending bit, allow field to wrap
; 1263        1352  2 !
; 1264        1353  2 if .left_bit lss .right_bit
; 1265        1354  2 then
; 1266        1355  3     begin
; 1267        1356  3     leftmost = 23;
; 1268        1357  3     wrap = true;
; 1269        1358  2     end;
; 1270        1359  2
; 1271        1360  2 !
; 1272        1361  2 ! Set field width(s) and set appropriate bits
; 1273        1362  2 !
; 1274        1363  2 width = .leftmost - .right_bit + 1;
; 1275        1364  2
; 1276        1365  2 (.address)<.right_bit, .width> = -1;
; 1277        1366  2
; 1278        1367  2 if .wrap
; 1279        1368  2 then (.address)<0, .left_bit+1> = -1;
; 1280        1369  2
; 1281        1370  2 return true;
; 1282        1371  2
; 1283        1372  1 end;



                                     003C 00000          .ENTRY  SET_RANGE, Save R2,R3,R4,R5                    ; 1282
              55 00000000'      00   9E 00002          MOVAB   LEFT_BIT, R5                                   ;
              06        18      A5   E9 00009          BLBC    SECONDARY, 1$                                  ; 1323
              53        0C      A5   9E 0000D          MOVAB   SECONDARY_ACCESS, ADDRESS                      ; 1324
                                04   11 00011          BRB     2$                                             ;
              53        08      A5   9E 00013 1$:      MOVAB   PRIMARY_ACCESS, ADDRESS                        ; 1325
              65        1C      AC   D0 00017 2$:      MOVL    28(AP), LEFT_BIT                               ; 1331
              54                65   D0 0001B          MOVL    LEFT_BIT, R4                                   ; 1336
              17                54   D1 0001E          CMPL    R4, #23                                        ;
                                47   1A 00021          BGTRU   6$                                             ;
              17        04      A5   D1 00023          CMPL    RIGHT_BIT, #23                                 ; 1337
                                41   1A 00027          BGTRU   6$                                             ;
         FFFFFFFF   8F          63   D1 00029          CMPL    (ADDRESS), #-1                                 ; 1344
                                02   12 00030          BNEQ    3$                                             ;
                                63   D4 00032          CLRL    (ADDRESS)                                      ; 1345
                                52   D4 00034 3$:      CLRL    WRAP                                           ; 1347
              50                54   D0 00036          MOVL    R4, LEFTMOST                                   ; 1348
```

```
                            51        04  A5  D0 00039           MOVL    RIGHT_BIT, R1                                ; 1353
                            51            54  D1 0003D           CMPL    R4, RT                                       ;
                                          06  18 00040           BGEQ    4$                                           ;
                            50            17  D0 00042           MOVL    #23, LEFTMOST                                ; 1356
                            52            01  D0 00045           MOVL    #1, WRAP                                     ; 1357
                            50            51  C2 00048 4$:       SUBL2   R1, R0                                       ; 1363
                                          50  D6 0004B           INCL    WIDTH                                        ;
       63            50     51 FFFFFFFF   8F  F0 0004D           INSV    #-1, R1, WIDTH, (ADDRESS)                    ; 1365
                            0D            52  E9 00056           BLBC    WRAP, 5$                                     ; 1367
                            50        01  A4  9E 00059           MOVAB   1(R4), R0                                    ; 1368
       63            50     00 FFFFFFFF   8F  F0 0005D           INSV    #-1, #0, R0, (ADDRESS)                       ;
                            50            01  D0 00066 5$:       MOVL    #1, R0                                       ; 1370
                                              04 00069           RET                                                  ;
                                          50  D4 0006A 6$:       CLRL    R0                                           ; 1372
                                              04 0006C           RET                                                  ;

; Routine Size:  109 bytes,    Routine Base:  $CODE$ + 0918
```

```
; 1285            1373  1 %sbttl 'getcputim - get cpu time quota'
; 1286            1374  1 global routine getcputim =
; 1287            1375  2 begin
; 1288            1376  2
; 1289            1377  2 !++
; 1290            1378  2 !
; 1291            1379  2 ! FUNCTIONAL DESCRIPTION:
; 1292            1380  2 !
; 1293            1381  2 !       Fill in the CPU time limit in the proper field in RECBUF.
; 1294            1382  2 !
; 1295            1383  2 ! INPUTS:
; 1296            1384  2 !
; 1297            1385  2 !       none
; 1298            1386  2 !
; 1299            1387  2 ! OUTPUTS:
; 1300            1388  2 !
; 1301            1389  2 !       none
; 1302            1390  2 !
; 1303            1391  2 ! ROUTINE VALUE:
; 1304            1392  2 !
; 1305            1393  2 !       true -> success
; 1306            1394  2 !       false -> invalid value supplied
; 1307            1395  2 !
; 1308            1396  2 !--
; 1309            1397  2
; 1310            1398  2 local
; 1311            1399  2         delta_time      : vector [2, long], ! system delta time temporary
; 1312            1400  2         cpu_time,                           ! CPU time work variables
; 1313            1401  2         remainder;
; 1314            1402  2
; 1315            1403  2 builtin
; 1316            1404  2         ediv;
; 1317            1405  2
; 1318            1406  2 !
; 1319            1407  2 ! Convert ASCII to system delta time
; 1320            1408  2 !
; 1321            1409  2
; 1322            1410  2 if not lib$cvt_dtime (tokendsc, delta_time)
; 1323            1411  2 then
; 1324            1412  2     return LIB$SIGNAL(UAF$_BADVALUE, 2, .tokenlen, .tokenptr);
; 1325            1413  2
; 1326            1414  2 !
; 1327            1415  2 ! Convert system delta time to a longword value which is in .01 sec. units
; 1328            1416  2 !
; 1329            1417  2
; 1330            1418  2 if (ediv (%ref (-200000), delta_time, cpu_time, remainder) and psl$m_v) neq 0
; 1331            1419  2 then
; 1332            1420  2     return LIB$SIGNAL(UAF$_VALTOOBIG, 2, .tokenlen, .tokenptr);
; 1333            1421  2
; 1334            1422  2 recbuf[uaf$l_cputim] = (.cpu_time ^ 1) + (if .remainder eql 0 then 0 else 1);
; 1335            1423  2
; 1336            1424  2 true
; 1337            1425  1 end;
```

```
                                                        003C  00000         .ENTRY  GETCPUTIM, Save R2,R3,R4,R5                  ; 1374
                                        55 00000000G  00 9E   00002         MOVAB   TOKENLEN, R5                                 ;
                                        54 00000000G  00 9E   00009         MOVAB   TOKENPTR, R4                                 ;
                                        5E            08 C2   00010         SUBL2   #8, SP                                       ;
                                                      5E DD   00013         PUSHL   SP                                           ; 1410
                                           00000000G  00 9F   00015         PUSHAB  TOKENDSC                                     ;
                             00000000G  00            02 FB   0001B         CALLS   #2, LIB$CVT_DTIME                            ;
                                        0F            50 E8   00022         BLBS    R0, 1$                                       ;
                                                      64 DD   00025         PUSHL   TOKENPTR                                     ; 1412
                                        7E            65 3C   00027         MOVZWL  TOKENLEN, -(SP)                              ;
                                                      02 DD   0002A         PUSHL   #2                                           ;
                                           00000000G  8F DD   0002C         PUSHL   #UAF$_BADVALUE                               ;
                                                      1C 11   00032         BRB     2$                                           ;
           53            52             6E FFFCF2C0   8F 7B   00034 1$:     EDIV    #-200000, DELTA_TIME, CPU_TIME, REMAINDER    ; 1418
                                                      50 DC   0003D         MOVPSL  R0                                           ;
                         15             50            01 E1   0003F         BBC     #1, R0, 3$                                   ;
                                                      64 DD   00043         PUSHL   TOKENPTR                                     ; 1420
                                        7E            65 3C   00045         MOVZWL  TOKENLEN, -(SP)                              ;
                                                      02 DD   00048         PUSHL   #2                                           ;
                                           00000000G  8F DD   0004A         PUSHL   #UAF$_VALTOOBIG                              ;
                             00000000G  00            04 FB   00050 2$:     CALLS   #4, LIB$SIGNAL                               ;
                                                         04   00057         RET                                                  ;
                                                      53 D5   00058 3$:     TSTL    REMAINDER                                    ; 1422
                                                      04 12   0005A         BNEQ    4$                                           ;
                                                      50 D4   0005C         CLRL    R0                                           ;
                                                      03 11   0005E         BRB     5$                                           ;
                                        50            01 D0   00060 4$:     MOVL    #1, R0                                       ;
                             00000000G  00          6042 3E   00063 5$:     MOVAW   (R0)[CPU_TIME], RECBUF+556                   ;
                                        50            01 D0   0006B         MOVL    #1, R0                                       ; 1425
                                                         04   0006E         RET                                                  ;

; Routine Size:  111 bytes,    Routine Base:  $CODE$ + 0985
```

```
1 %sbttl 'getdevice - get default device name'
1 global routine getdevice =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Fill in the default device name in RECBUF.
2 !       Will also append an ending colon if omitted.
2 !
2 ! INPUTS:
2 !
2 !       none
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> update successful
2 !       false -> error in specification
2 !--
2
2 local
2       strlen;
2
2 if not getstring (recbuf[uaf$t_defdev], uaf$s_defdev, counted_string)
2 then
2     return false;
2
2 !
2 ! Check to see that string has ending ':'
2 ! and add one if not.
2 ! If string length is zero, then field has been removed so
2 ! there's no need for the check.
2 !
2
2 strlen = .(recbuf[uaf$t_defdev])<0,8>;                        ! get string length
2
2 if .strlen eql 0
2 then
2     return true;
2
2 if .(recbuf[uaf$t_defdev] + .strlen)<0,8> neq colon
2 then
3     begin
3     if .strlen geq uaf$s_defdev-1
3     then
3         return LIB$SIGNAL(UAF$_INVDEV, 2, .tokenlen, .tokenptr);
3     (recbuf[uaf$t_defdev]+.strlen+1)<0,8> = colon;
3     (recbuf[uaf$t_defdev])<0,8> = .strlen + 1;
2     end;
2 return true;
1 end;
```

```
                                                       000C 00000         .ENTRY  GETDEVICE, Save R2,R3               ; 1427
                                       53 00000000G  00  9E 00002         MOVAB   RECBUF+116, R3                      ;
                                                     01  DD 00009         PUSHL   #1                                  ; 1454
                                                     20  DD 0000B         PUSHL   #32                                 ;
                                                     53  DD 0000D         PUSHL   R3                                  ;
                            00000000V  00            03  FB 0000F         CALLS   #3, GETSTRING                       ;
                                       3A            50  E9 00016         BLBC    R0, 3$                              ;
                                       52            63  9A 00019         MOVZBL  RECBUF+116, STRLEN                  ; 1465
                                                     31  13 0001C         BEQL    2$                                  ; 1467
                                       3A          6342  91 0001E         CMPB    RECBUF+116[STRLEN], #58             ; 1471
                                                     2B  13 00022         BEQL    2$                                  ;
                                       1F            52  D1 00024         CMPL    STRLEN, #31                         ; 1474
                                                     1D  19 00027         BLSS    1$                                  ;
                                          00000000G  00  DD 00029         PUSHL   TOKENPTR                            ; 1476
                                       7E 00000000G  00  3C 0002F         MOVZWL  TOKENLEN, -(SP)                     ;
                                                     02  DD 00036         PUSHL   #2                                  ;
                                          00000000G  8F  DD 00038         PUSHL   #UAF$_INVDEV                        ;
                            00000000G  00            04  FB 0003E         CALLS   #4, LIB$SIGNAL                      ;
                                                         04 00045         RET                                         ;
                                  01 A342            3A  90 00046 1$:     MOVB    #58, RECBUF+117[STRLEN]             ; 1477
                         63            52            01  81 0004B         ADDB3   #1, STRLEN, RECBUF+116              ; 1478
                                       50            01  D0 0004F 2$:     MOVL    #1, R0                              ; 1480
                                                         04 00052         RET                                         ;
                                                     50  D4 00053 3$:     CLRL    R0                                  ; 1481
                                                         04 00055         RET                                         ;

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 09F4
```

```
1 %sbttl 'getdirectory - get default directory'
1 global routine getdirectory =
2 begin
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine parses the user default directory and places
2 !       it into the buffer if the form is correct.  If delimiters
2 !       are not present, they are added.
2 !
2 ! INPUTS:
2 !
2 !       none
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       TOKENLEN = length of default directory string
2 !       TOKENPTR = pointer to default directory string
2 !
2 !--
2 local BRACKETLESS;
2
2 !
2 ! Crank directory spec through tparse
2 !
2 tparse_block [tpa$l_stringcnt] = .tokenlen;
2 tparse_block [tpa$l_stringptr] = .tokenptr;
2
2 if not lib$tparse (tparse_block, dir_states, dir_keys)
2 then
2     return LIB$SIGNAL(UAF$_BADVALUE, 2, .tokenlen, .tokenptr);
2
2 !
2 ! Did the string have enclosing brackets (matching is guaranteed
2 ! by the command parser) ??
2 !
2 BRACKETLESS = (.(.tokenptr)<0,8> neq '[') and (.(.tokenptr)<0,8> neq '<');
2
2 !
2 ! Make sure that the string isn't too long
2 !
3 if (((.tokenlen+1)  geq  uaf$s_defdir) and not .BRACKETLESS)
3 or ((.tokenlen+3) geq uaf$s_defdir)
2 then
2     return LIB$SIGNAL(UAF$_INVSTR, 2, .tokenlen, .tokenptr);
2
2 !
2 ! Move the string into the UAF record
2 !
2 if .BRACKETLESS then
3    begin
3       vector[recbuf[uaf$t_defdir], 0;, byte] = .tokenlen + 2;
```

```
; 1453          1539 3              vector[recbuf[uaf$t_defdir], 1;, byte] = '[';
; 1454          1540 3              ch$copy(.tokenlen, .tokenptr, ' ',
; 1455          1541 3                  uaf$s_defdir-2, recbuf[uaf$t_defdir]+2);
; 1456          1542 3              vector[recbuf[uaf$t_defdir], .tokenlen+2;, byte] = ']';
; 1457          1543 3          end
; 1458          1544 2      else
; 1459          1545 3          begin
; 1460          1546 3              (recbuf[uaf$t_defdir])<0,8> = .tokenlen;
; 1461          1547 3              ch$copy(.tokenlen, .tokenptr, ' ',
; 1462          1548 3                  uaf$s_defdir-1, recbuf[uaf$t_defdir]+1);
; 1463          1549 2          end;
; 1464          1550 2
; 1465          1551 2      return true;
; 1466          1552 2
; 1467          1553 1      end;



                                  07FC 00000          .ENTRY  GETDIRECTORY, Save R2,R3,R4,R5,R6,R7,R8,R9,-;  1483
                                                              R10                                        :
                5A 00000000'   00  9E 00002           MOVAB   TPARSE_BLOCK+8, R10                        :
                59 00000000G   00  9E 00009           MOVAB   TOKENLEN, R9                               :
                58 00000000G   00  9E 00010           MOVAB   TOKENPTR, R8                               :
                57 00000000G   00  9E 00017           MOVAB   RECBUF+148, R7                             :
                6A             69  3C 0001E           MOVZWL  TOKENLEN, TPARSE_BLOCK+8                   ;  1512
          04    AA             68  D0 00021           MOVL    TOKENPTR, TPARSE_BLOCK+12                  ;  1513
                   00000000'   00  9F 00025           PUSHAB  DIR_KEYS                                   ;  1515
                   00000000'   00  9F 0002B           PUSHAB  DIR_STATES                                 :
                          F8   AA  9F 00031           PUSHAB  TPARSE_BLOCK                               :
     00000000G  00             03  FB 00034           CALLS   #3, LIB$TPARSE                             :
                0F             50  E8 0003B           BLBS    R0, 1$                                     :
                               68  DD 0003E           PUSHL   TOKENPTR                                   ;  1517
                7E             69  3C 00040           MOVZWL  TOKENLEN, -(SP)                            :
                               02  DD 00043           PUSHL   #2                                         :
                   00000000G   8F  DD 00045           PUSHL   #UAF$_BADVALUE                             :
                               3F  11 0004B           BRB     6$                                         :
                52             68  D0 0004D 1$:       MOVL    TOKENPTR, R2                               ;  1523
                               51  D4 00050           CLRL    R1                                         :
          5B    8F             62  91 00052           CMPB    (R2), #91                                  :
                               02  13 00056           BEQL    2$                                         :
                               51  D6 00058           INCL    R1                                         :
                               50  D4 0005A 2$:       CLRL    R0                                         :
                3C             62  91 0005C           CMPB    (R2), #60                                  :
                               02  13 0005F           BEQL    3$                                         :
                               50  D6 00061           INCL    R0                                         :
                53             51  D2 00063 3$:       MCOML   R1, BRACKETLESS                            :
     53         50             53  CB 00066           BICL3   BRACKETLESS, R0, BRACKETLESS               :
                51             69  3C 0006A           MOVZWL  TOKENLEN, R1                               ;  1528
                50        01   A1  9E 0006D           MOVAB   1(R1), R0                                  :
                3F             50  D1 00071           CMPL    R0, #63                                    :
                               03  15 00074           BLEQ    4$                                         :
                09             53  E9 00076           BLBC    BRACKETLESS, 5$                            :
                50        03   A1  9E 00079 4$:       MOVAB   3(R1), R0                                  ;  1529
                3F             50  D1 0007D           CMPL    R0, #63                                    :
                               12  15 00080           BLEQ    7$                                         :
```

```
                                                     06 BB 00082 5$:      PUSHR   #^M<R1,R2>                                  ; 1531
                                                     02 DD 00084          PUSHL   #2                                          ;
                                          00000000G  8F DD 00086          PUSHL   #UAF$_INVSTR                                ;
                            00000000G  00            04 FB 0008C 6$:      CALLS   #4, LIB$SIGNAL                              ;
                                                        04 00093          RET                                                 ;
                                       50            69 3C 00094 7$:      MOVZWL  TOKENLEN, R0                                ; 1538
                                       51            68 D0 00097          MOVL    TOKENPTR, R1                                ; 1540
                                       1A            53 E9 0009A          BLBC    BRACKETLESS, 8$                             ; 1536
                                       56       02   A0 9E 0009D          MOVAB   2(R0), R6                                   ; 1538
                                       67            56 90 000A1          MOVB    R6, RECBUF+148                              ;
                                  01   A7       5B   8F 90 000A4          MOVB    #91, RECBUF+149                             ; 1539
           3E            20            61            50 2C 000A9          MOVC5   R0, (R1), #32, #62, RECBUF+150              ; 1541
                                                02   A7    000AE                                                              ;
                                     6746       5D   8F 90 000B0          MOVB    #93, RECBUF+148[R6]                         ; 1542
                                                     0A 11 000B5          BRB     9$                                          ; 1536
                                       67            50 90 000B7 8$:      MOVB    R0, RECBUF+148                              ; 1546
           3F            20            61            50 2C 000BA          MOVC5   R0, (R1), #32, #63, RECBUF+149              ; 1548
                                                01   A7    000BF                                                              ;
                                       50            01 D0 000C1 9$:      MOVL    #1, R0                                      ; 1551
                                                        04 000C4          RET                                                 ; 1553

; Routine Size:  197 bytes,    Routine Base:  $CODE$ + 0A4A
```

```
; 1469          1554  1 %sbttl 'checkvalue - check UFD group and member value'
; 1470          1555  1 routine checkvalue =
; 1471          1556  2 begin
; 1472          1557  2 builtin ap;
; 1473          1558  2 map ap: ref block [,byte];
; 1474          1559  2 if .ap[tpa$l_number] gtr %o'377' then return false else return true;
; 1475          1560  1 end;



                                          0000 00000 CHECKVALUE:
                                                          .WORD    Save nothing                                         ; 1555
                   000000FF   8F       1C    AC  D1 00002       CMPL     28(AP), #255                                        ; 1559
                                             03  15 0000A       BLEQ     1$                                                  ;
                                             50  D4 0000C       CLRL     R0                                                  ;
                                                 04 0000E       RET                                                          ;
                              50             01  D0 0000F 1$:   MOVL     #1, R0                                              ;
                                                 04 00012       RET                                                          ; 1560

; Routine Size:  19 bytes,    Routine Base:  $CODE$ + 0B0F
```

```
; 1477          1561   1 %sbttl 'checklength - check directory name length'
; 1478          1562   1 routine checklength =
; 1479          1563   2 begin
; 1480          1564   2 builtin ap;
; 1481          1565   2 map ap: ref block [,byte];
; 1482          1566   2 if .ap[tpa$l_tokencnt] gtr 39 then return false else return true;
; 1483          1567   1 end;



                                                  0000 00000 CHECKLENGTH:
                                                                   .WORD    Save nothing                                  ; 1562
                     27          10       AC   D1 00002            CMPL     16(AP), #39                                   ; 1566
                                          03   15 00006            BLEQ     1$                                            ;
                                          50   D4 00008            CLRL     R0                                            ;
                                               04 0000A            RET                                                    ;
                     50                   01   D0 0000B 1$:        MOVL     #1, R0                                        ;
                                               04 0000E            RET                                                    ; 1567

; Routine Size:  15 bytes,    Routine Base:  $CODE$ + 0B22
```

```
1 %sbttl 'getflags - get flag bits'
1 global routine getflags =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Get login flag bit settings.
2 !
2 ! INPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       FLAG_TABLE - table of flag bit numbers and ascii names
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> successful completion
2 !       false -> error in flag name
2 !--
2
2 local
2       status,
2       keyword_dsc     : vector [2],
2       no_flag,
2       code,                           ! return from find_val
2       ientry;
2
2 !
2 ! Retrieve flag values as long as there are any present
2 !
2
2 while cli$get_value (sd_flags, tokendsc) do
2     !
2     ! Initialize keyword descriptor
2     !
3     begin
3     keyword_dsc [0] = .tokenlen;
3     keyword_dsc [1] = .tokenptr;
3     no_flag = false;
3
3     !
3     ! If 'NO' prefix specified, set no_flag and adjust the
3     !   keyword descriptor to remove the 'NO'
3     !
3     if ch$eql (2, .no_dsc [1], 2, .keyword_dsc [1])
3     then
4         begin
4         keyword_dsc [0] = .keyword_dsc [0] - 2;
4         keyword_dsc [1] = .keyword_dsc [1] + 2;
```

```
; 1542          1625 4          no_flag = true;
; 1543          1626 3          end;
; 1544          1627 3
; 1545          1628 4      if (status = lib$lookup_key (keyword_dsc, flag_table, code))
; 1546          1629 3      then
; 1547          1630 3          !
; 1548          1631 3          ! valid flag, set bit
; 1549          1632 3          !
; 1550          1633 3          (recbuf[uaf$l_flags])<.code,1> = not .no_flag
; 1551          1634 3      else
; 1552          1635 3          select .status of
; 1553          1636 3              set
; 1554          1637 3              !
; 1555          1638 3              ! ambiguous keyword?
; 1556          1639 3              !
; 1557          1640 3              [lib$_ambkey] : signal (cli$_abkeyw);
; 1558          1641 3              !
; 1559          1642 3              ! unrecognized keyword?
; 1560          1643 3              !
; 1561          1644 3              [lib$_unrkey] : signal (cli$_ivkeyw);
; 1562          1645 3              !
; 1563          1646 3              ! whatever, always return an error if we're in here
; 1564          1647 3              !
; 1565          1648 3              [always]      : return false;
; 1566          1649 3              tes;
; 1567          1650 3
; 1568          1651 2      end;
; 1569          1652 2
; 1570          1653 2 return true;
; 1571          1654 1 end;



                              001C 00000         .ENTRY  GETFLAGS, Save R2,R3,R4                         ; 1569
         54 00000000G    00    9E 00002         MOVAB   LIB$SIGNAL, R4                                  ;
         5E              0C    C2 00009         SUBL2   #12, SP                                         ;
            00000000G    00    9F 0000C 1$:     PUSHAB  TOKENDSC                                        ; 1607
            00000000'    00    9F 00012         PUSHAB  SD_FLAGS                                        ;
00000000G    00          02    FB 00018         CALLS   #2, CLI$GET_VALUE                               ;
             76          50    E9 0001F         BLBC    R0, 5$                                          ;
      04     AE 00000000G 00   3C 00022         MOVZWL  TOKENLEN, KEYWORD_DSC                           ; 1612
      08     AE 00000000G 00   D0 0002A         MOVL    TOKENPTR, KEYWORD_DSC+4                         ; 1613
             52          D4 00032               CLRL    NO_FLAG                                         ; 1614
             50 00000000' 00   D0 00034         MOVL    NO_DSC+4, R0                                    ; 1620
      08     BE          60    B1 0003B         CMPW    (R0), @KEYWORD_DSC+4                            ;
             0B          12 0003F               BNEQ    2$                                              ;
      04     AE          02    C2 00041         SUBL2   #2, KEYWORD_DSC                                 ; 1623
      08     AE          02    C0 00045         ADDL2   #2, KEYWORD_DSC+4                               ; 1624
             52          01    D0 00049         MOVL    #1, NO_FLAG                                     ; 1625
                         5E    DD 0004C 2$:     PUSHL   SP                                              ; 1628
            00000000'    00    9F 0004E         PUSHAB  FLAG_TABLE                                      ;
                   0C    AE    9F 00054         PUSHAB  KEYWORD_DSC                                     ;
00000000G    00          03    FB 00057         CALLS   #3, LIB$LOOKUP_KEY                              ;
             53          50    D0 0005E         MOVL    R0, STATUS                                      ;
             0E          53    E9 00061         BLBC    STATUS, 3$                                      ;
```

```
                                         50          52  D2 00064          MCOML   NO_FLAG, R0                          ; 1633
00000000G  00           01               6E          50  F0 00067          INSV    R0, CODE, #1, RECBUF+468             ;
                                                     9A  11 00070          BRB     1$                                   ;
                         00000000G       8F          53  D1 00072 3$:      CMPL    STATUS, #LIB$_AMBKEY                 ; 1640
                                                     09  12 00079          BNEQ    4$                                   ;
                                     00000000G       8F  DD 0007B          PUSHL   #CLI$_ABKEYW                         ;
                                         64          01  FB 00081          CALLS   #1, LIB$SIGNAL                       ;
                         00000000G       8F          53  D1 00084 4$:      CMPL    STATUS, #LIB$_UNRKEY                 ; 1644
                                                     0F  12 0008B          BNEQ    6$                                   ;
                                     00000000G       8F  DD 0008D          PUSHL   #CLI$_IVKEYW                         ;
                                         64          01  FB 00093          CALLS   #1, LIB$SIGNAL                       ;
                                                     04  11 00096          BRB     6$                                   ; 1648
                                         50          01  D0 00098 5$:      MOVL    #1, R0                               ; 1653
                                                         04 0009B          RET                                          ;
                                                     50  D4 0009C 6$:      CLRL    R0                                   ; 1654
                                                         04 0009E          RET                                          ;

; Routine Size:  159 bytes,    Routine Base:  $CODE$ + 0B31
```

```
1 %sbttl 'getpflags - get old per day login flags'
1 global routine getpflags (qual_name) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Get prime day login flag bit settings.
2 !
2 ! INPUTS:
2 !
2 !       qual_name - address of descriptor for qualifier name
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       OPFLAG_TABLE - table of flag bit numbers and ascii names
2 !       PRIMARY - true if getting primary day, false if secondary
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> successful completion
2 !       false -> error in flag name
2 !--
2
2 local
2       status,
2       keyword_dsc     : vector [2],
2       no_flag,
2       code,                                   ! return from find_val
2       ientry;
2
2 !
2 ! Retrieve keywords as long as any are present
2 !
2 while cli$get_value (.qual_name, tokendsc) do
2
3     begin
3     keyword_dsc [0] = .tokenlen;
3     keyword_dsc [1] = .tokenptr;
3     no_flag = 0;
3
3     !
3     ! Test for presence of 'NO'prefix.  If present, adjust descriptor
3     !    to remove it, and set no_flag.
3     !
3     if ch$eql (2, .no_dsc [1], 2, .keyword_dsc [1])
3     then
3
4       begin
4       keyword_dsc [0] = .keyword_dsc [0] - 2;
4       keyword_dsc [1] = .keyword_dsc [1] + 2;
4       no_flag = -1;
```

```
; 1630          1712  3         end;
; 1631          1713  3
; 1632          1714  4      if (status = lib$lookup_key (keyword_dsc, opflag_table, code))
; 1633          1715  3      then
; 1634          1716  3         !
; 1635          1717  3         ! valid flag, set bit
; 1636          1718  3         !
; 1637          1719  3         case .code from uaf$k_disdialup to uaf$k_disnetwork of
; 1638          1720  3            set
; 1639          1721  3            [uaf$k_disdialup]:
; 1640          1722  4               begin
; 1641          1723  4               if .primary
; 1642          1724  4               then recbuf[uaf$b_dialup_access_p] = not .no_flag
; 1643          1725  4               else recbuf[uaf$b_dialup_access_s] = not .no_flag;
; 1644          1726  3               end;
; 1645          1727  3            [uaf$k_disnetwork]:
; 1646          1728  4               begin
; 1647          1729  4               if .primary
; 1648          1730  4               then recbuf[uaf$b_remote_access_p] = not .no_flag
; 1649          1731  4               else recbuf[uaf$b_remote_access_s] = not .no_flag;
; 1650          1732  3               end;
; 1651          1733  3            tes
; 1652          1734  3      else
; 1653          1735  3         select .status of
; 1654          1736  3            set
; 1655          1737  3            !
; 1656          1738  3            ! ambiguous keyword?
; 1657          1739  3            !
; 1658          1740  3            [lib$_ambkey] : signal (cli$_abkeyw);
; 1659          1741  3            !
; 1660          1742  3            ! unrecognized keyword?
; 1661          1743  3            !
; 1662          1744  3            [lib$_unrkey] : signal (cli$_ivkeyw);
; 1663          1745  3            !
; 1664          1746  3            ! whatever, always return an error if we're in here
; 1665          1747  3            !
; 1666          1748  3            [always]      : return false;
; 1667          1749  3            tes;
; 1668          1750  3
; 1669          1751  2      end;
; 1670          1752  2
; 1671          1753  2 return true;
; 1672          1754  1 end;



                                 003C 00000          .ENTRY  GETPFLAGS, Save R2,R3,R4,R5                          ; 1656
           55 00000000G   00     9E 00002            MOVAB   LIB$SIGNAL, R5                                       ;
           54 00000000G   00     9E 00009            MOVAB   RECBUF+490, R4                                       ;
           5E             0C     C2 00010            SUBL2   #12, SP                                              ;
              00000000G   00     9F 00013 1$:        PUSHAB  TOKENDSC                                             ; 1694
                     04   AC     DD 00019            PUSHL   QUAL_NAME                                            ;
     00000000G  00                02     FB 0001C    CALLS   #2, CLI$GET_VALUE                                    ;
                03                50     E8 00023    BLBS    R0, 2$                                               ;
                               009F    31 00026      BRW     12$                                                  ;
```

```
                                    04   AE 00000000G   00 3C 00029 2$:     MOVZWL  TOKENLEN, KEYWORD_DSC                         ; 1697
                                    08   AE 00000000G   00 D0 00031         MOVL    TOKENPTR, KEYWORD_DSC+4                       ; 1698
                                                        52 D4 00039         CLRL    NO_FLAG                                       ; 1699
                                         50 00000000'   00 D0 0003B         MOVL    NO_DSC+4, R0                                  ; 1705
                                    08   BE             60 B1 00042         CMPW    (R0), @KEYWORD_DSC+4                          ;
                                                        0B 12 00046         BNEQ    3$                                            ;
                                    04   AE             02 C2 00048         SUBL2   #2, KEYWORD_DSC                               ; 1709
                                    08   AE             02 C0 0004C         ADDL2   #2, KEYWORD_DSC+4                             ; 1710
                                         52             01 CE 00050         MNEGL   #1, NO_FLAG                                   ; 1711
                                                        5E DD 00053 3$:     PUSHL   SP                                            ; 1714
                                            00000000'   00 9F 00055         PUSHAB  OPFLAG_TABLE                                  ;
                                                   0C   AE 9F 0005B         PUSHAB  KEYWORD_DSC                                   ;
                              00000000G  00             03 FB 0005E         CALLS   #3, LIB$LOOKUP_KEY                            ;
                                         53             50 D0 00065         MOVL    R0, STATUS                                    ;
                                         37             53 E9 00068         BLBC    STATUS, 10$                                   ;
                                         51 00000000'   00 D0 0006B         MOVL    PRIMARY, R1                                   ; 1723
                                         50             52 D2 00072         MCOML   NO_FLAG, R0                                   ; 1724
                           01            01             6E CF 00075         CASEL   CODE, #1, #1                                  ; 1719
                                       0016           0004    00079 4$:     .WORD   5$-4$,-                                       ;
                                                                                    8$-4$                                         ;
                                         07             51 E9 0007D 5$:     BLBC    R1, 7$                                        ; 1723
            64             18            00             50 F0 00080         INSV    R0, #0, #24, RECBUF+490                       ; 1724
                                                        8C 11 00085 6$:     BRB     1$                                            ;
       03   A4             18            00             50 F0 00087 7$:     INSV    R0, #0, #24, RECBUF+493                       ; 1725
                                                        84 11 0008D         BRB     1$                                            ; 1719
                                         08             51 E9 0008F 8$:     BLBC    R1, 9$                                        ; 1729
       06   A4             18            00             50 F0 00092         INSV    R0, #0, #24, RECBUF+496                       ; 1730
                                                        EB 11 00098         BRB     6$                                            ;
       09   A4             18            00             50 F0 0009A 9$:     INSV    R0, #0, #24, RECBUF+499                       ; 1731
                                                        E3 11 000A0         BRB     6$                                            ; 1719
                              00000000G  8F             53 D1 000A2 10$:    CMPL    STATUS, #LIB$_AMBKEY                          ; 1740
                                                        09 12 000A9         BNEQ    11$                                           ;
                                            00000000G   8F DD 000AB         PUSHL   #CLI$_ABKEYW                                  ;
                                         65             01 FB 000B1         CALLS   #1, LIB$SIGNAL                                ;
                              00000000G  8F             53 D1 000B4 11$:    CMPL    STATUS, #LIB$_UNRKEY                          ; 1744
                                                        0F 12 000BB         BNEQ    13$                                           ;
                                            00000000G   8F DD 000BD         PUSHL   #CLI$_IVKEYW                                  ;
                                         65             01 FB 000C3         CALLS   #1, LIB$SIGNAL                                ;
                                                        04 11 000C6         BRB     13$                                           ; 1748
                                         50             01 D0 000C8 12$:    MOVL    #1, R0                                        ; 1753
                                                           04 000CB         RET                                                   ;
                                                        50 D4 000CC 13$:    CLRL    R0                                            ; 1754
                                                           04 000CE         RET                                                   ;

; Routine Size:  207 bytes,    Routine Base:  $CODE$ + 0BD0
```

```
1 %sbttl 'getrestrict - parse hourly restriction ranges'
1 global routine getrestrict (qual_name) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Parse hourly login restrictions for primary or secondary days
2 !
2 ! INPUTS:
2 !
2 !       qual_name - descriptor of qualifier name to use
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> successful completion
2 !       false -> error in flag name
2 !--
2
2 !
2 ! Initialize output values.
2 !
2 primary_access = 0;
2 secondary_access = 0;
2
2 !
2 ! Fetch value list items and parse them.
2 !
2 while true
2 do
3     begin
3     if not cli$get_value (.qual_name, tokendsc)
3     then exitloop;
3
3     tparse_block[tpa$l_stringcnt] = .tokenlen;
3     tparse_block[tpa$l_stringptr] = .tokenptr;
3     if not lib$tparse (tparse_block, access_states, access_keys)
3     then return LIB$SIGNAL(UAF$_BADVALUE, 2, .tokenlen, .tokenptr);
3     end;
2
2 !
2 ! Write the modified fields into the UAF record.
2 !
2 if not .secondary
2 then
3     begin
3     recbuf[uaf$b_network_access_p] = .primary_access;
3     recbuf[uaf$b_batch_access_p] = .primary_access;
3     recbuf[uaf$b_local_access_p] = .primary_access;
```

```
; 1731          1812  3       recbuf[uaf$b_dialup_access_p] = .primary_access;
; 1732          1813  3       recbuf[uaf$b_remote_access_p] = .primary_access;
; 1733          1814  3       end
; 1734          1815  2  else
; 1735          1816  3       begin
; 1736          1817  3       recbuf[uaf$b_network_access_s] = .secondary_access;
; 1737          1818  3       recbuf[uaf$b_batch_access_s] = .secondary_access;
; 1738          1819  3       recbuf[uaf$b_local_access_s] = .secondary_access;
; 1739          1820  3       recbuf[uaf$b_dialup_access_s] = .secondary_access;
; 1740          1821  3       recbuf[uaf$b_remote_access_s] = .secondary_access;
; 1741          1822  2       end;
; 1742          1823  2
; 1743          1824  2  true
; 1744          1825  1  end;
```

```
                                                         003C 00000         .ENTRY  GETRESTRICT, Save R2,R3,R4,R5                ; 1756
                                        55 00000000G  00  9E  00002         MOVAB   TOKENPTR, R5                                 ;
                                        54 00000000G  00  9E  00009         MOVAB   TOKENLEN, R4                                 ;
                                        53 00000000'  00  9E  00010         MOVAB   PRIMARY_ACCESS, R3                           ;
                                        52 00000000G  00  9E  00017         MOVAB   RECBUF+472, R2                               ;
                                                      63  7C  0001E         CLRQ    PRIMARY_ACCESS                               ; 1785
                                           00000000G  00  9F  00020 1$:     PUSHAB  TOKENDSC                                     ; 1794
                                                  04  AC  DD  00026         PUSHL   QUAL_NAME                                    ;
                             00000000G  00              02  FB  00029         CALLS   #2, CLI$GET_VALUE                            ;
                                        36              50  E9  00030         BLBC    R0, 2$                                       ;
                                   D4   A3              64  3C  00033         MOVZWL  TOKENLEN, TPARSE_BLOCK+8                     ; 1797
                                   D8   A3              65  D0  00037         MOVL    TOKENPTR, TPARSE_BLOCK+12                    ; 1798
                                           00000000'  00  9F  0003B         PUSHAB  ACCESS_KEYS                                  ; 1799
                                           00000000'  00  9F  00041         PUSHAB  ACCESS_STATES                                ;
                                                  CC  A3  9F  00047         PUSHAB  TPARSE_BLOCK                                 ;
                             00000000G  00              03  FB  0004A         CALLS   #3, LIB$TPARSE                               ;
                                        CC              50  E8  00051         BLBS    R0, 1$                                       ;
                                                      65  DD  00054         PUSHL   TOKENPTR                                     ; 1800
                                        7E              64  3C  00056         MOVZWL  TOKENLEN, -(SP)                              ;
                                                      02  DD  00059         PUSHL   #2                                           ;
                                           00000000G  8F  DD  0005B         PUSHL   #UAF$_BADVALUE                               ;
                             00000000G  00              04  FB  00061         CALLS   #4, LIB$SIGNAL                               ;
                                                          04  00068         RET                                                  ;
                                        22        10  A3  E8  00069 2$:     BLBS    SECONDARY, 3$                                ; 1806
                                        50              63  D0  0006D         MOVL    PRIMARY_ACCESS, R0                           ; 1809
            62            18            00              50  F0  00070         INSV    R0, #0, #24, RECBUF+472                      ;
      06    A2            18            00              50  F0  00075         INSV    R0, #0, #24, RECBUF+478                      ; 1810
      0C    A2            18            00              50  F0  0007B         INSV    R0, #0, #24, RECBUF+484                      ; 1811
      12    A2            18            00              50  F0  00081         INSV    R0, #0, #24, RECBUF+490                      ; 1812
      18    A2            18            00              50  F0  00087         INSV    R0, #0, #24, RECBUF+496                      ; 1813
                                                      22  11  0008D         BRB     4$                                           ; 1806
                                        50        04  A3  D0  0008F 3$:     MOVL    SECONDARY_ACCESS, R0                         ; 1817
      03    A2            18            00              50  F0  00093         INSV    R0, #0, #24, RECBUF+475                      ;
      09    A2            18            00              50  F0  00099         INSV    R0, #0, #24, RECBUF+481                      ; 1818
      0F    A2            18            00              50  F0  0009F         INSV    R0, #0, #24, RECBUF+487                      ; 1819
      15    A2            18            00              50  F0  000A5         INSV    R0, #0, #24, RECBUF+493                      ; 1820
      1B    A2            18            00              50  F0  000AB         INSV    R0, #0, #24, RECBUF+499                      ; 1821
                                        50              01  D0  000B1 4$:     MOVL    #1, R0                                       ; 1825
```

```
                                             04 000B4          RET                                                         ;

; Routine Size:  181 bytes,     Routine Base:  $CODE$ + 0C9F
```

```
1 %sbttl 'getprimedays - get primary day list'
1 global routine getprimedays =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Get list of day considered prime days.
2 !
2 ! INPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       PDFLAG_TABLE - table of flag bit numbers and ascii names
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> successful completion
2 !       false -> error in flag name
2 !--
2
2 local
2         status,
2         keyword_dsc     : vector [2],
2         no_flag,
2         code,                                       ! return from find_val
2         ientry;
2
2 !
2 ! Retrieve keywords as long as they are present
2 !
2 while cli$get_value (sd_primedays, tokendsc) do
2
3     begin
3     keyword_dsc [0] = .tokenlen;
3     keyword_dsc [1] = .tokenptr;
3     no_flag = false;
3
3     !
3     ! Test for presence of the 'NO' prefix.  If present, adjust the
3     !   descriptor to remove it, and set no_flag
3     !
3     if ch$eql (2, .no_dsc [1], 2, .keyword_dsc [1])
3     then
4         begin
4         keyword_dsc [0] = .keyword_dsc [0] - 2;
4         keyword_dsc [1] = .keyword_dsc [1] + 2;
4         no_flag = true;
3         end;
3
```

```
; 1803          1883  4        if (status = lib$lookup_key (keyword_dsc, pdflag_table, code))
; 1804          1884  3        then
; 1805          1885  3            !
; 1806          1886  3            ! valid flag, set bit
; 1807          1887  3            !
; 1808          1888  3            (recbuf[uaf$b_primedays])<.code,1> =  .no_flag
; 1809          1889  3        else
; 1810          1890  3            select .status of
; 1811          1891  3                set
; 1812          1892  3                !
; 1813          1893  3                ! ambiguous keyword?
; 1814          1894  3                !
; 1815          1895  3                [lib$_ambkey] : signal (cli$_abkeyw);
; 1816          1896  3                !
; 1817          1897  3                ! unrecognized keyword?
; 1818          1898  3                !
; 1819          1899  3                [lib$_unrkey] : signal (cli$_ivkeyw);
; 1820          1900  3                !
; 1821          1901  3                ! whatever, always return an error if we're in here
; 1822          1902  3                !
; 1823          1903  3                [always]       : return false;
; 1824          1904  3                tes;
; 1825          1905  3
; 1826          1906  2        end;
; 1827          1907  2
; 1828          1908  2   return true;
; 1829          1909  1   end;




                                                001C 00000          .ENTRY   GETPRIMEDAYS, Save R2,R3,R4                        ; 1827
                        54 00000000G        00   9E 00002          MOVAB    LIB$SIGNAL, R4                                     ;
                        5E                  0C   C2 00009          SUBL2    #12, SP                                            ;
                           00000000G        00   9F 0000C 1$:      PUSHAB   TOKENDSC                                           ; 1864
                           00000000'        00   9F 00012          PUSHAB   SD_PRIMEDAYS                                       ;
                00000000G  00               02   FB 00018          CALLS    #2, CLI$GET_VALUE                                  ;
                           73               50   E9 0001F          BLBC     R0, 5$                                             ;
                     04    AE 00000000G     00   3C 00022          MOVZWL   TOKENLEN, KEYWORD_DSC                              ; 1867
                     08    AE 00000000G     00   D0 0002A          MOVL     TOKENPTR, KEYWORD_DSC+4                            ; 1868
                                            52   D4 00032          CLRL     NO_FLAG                                            ; 1869
                        50 00000000'        00   D0 00034          MOVL     NO_DSC+4, R0                                       ; 1875
                     08    BE               60   B1 0003B          CMPW     (R0), @KEYWORD_DSC+4                               ;
                                            0B   12 0003F          BNEQ     2$                                                 ;
                     04    AE               02   C2 00041          SUBL2    #2, KEYWORD_DSC                                    ; 1878
                     08    AE               02   C0 00045          ADDL2    #2, KEYWORD_DSC+4                                  ; 1879
                           52               01   D0 00049          MOVL     #1, NO_FLAG                                        ; 1880
                                            5E   DD 0004C 2$:      PUSHL    SP                                                 ; 1883
                           00000000'        00   9F 0004E          PUSHAB   PDFLAG_TABLE                                       ;
                                  0C        AE   9F 00054          PUSHAB   KEYWORD_DSC                                        ;
                00000000G  00               03   FB 00057          CALLS    #3, LIB$LOOKUP_KEY                                 ;
                           53               50   D0 0005E          MOVL     R0, STATUS                                         ;
                           0B               53   E9 00061          BLBC     STATUS, 3$                                         ;
00000000G  00         01   6E               52   F0 00064          INSV     NO_FLAG, CODE, #1, RECBUF+514                      ; 1888
                                            9D   11 0006D          BRB      1$                                                 ;
                00000000G  8F               53   D1 0006F 3$:      CMPL     STATUS, #LIB$_AMBKEY                               ; 1895
```

```
                                          09  12 00076          BNEQ    4$                                    ;
                              00000000G   8F  DD 00078          PUSHL   #CLI$_ABKEYW                          ;
                                     64   01  FB 0007E          CALLS   #1, LIB$SIGNAL                        ;
                 00000000G   8F           53  D1 00081 4$:      CMPL    STATUS, #LIB$_UNRKEY                  ; 1899
                                          0F  12 00088          BNEQ    6$                                    ;
                              00000000G   8F  DD 0008A          PUSHL   #CLI$_IVKEYW                          ;
                                     64   01  FB 00090          CALLS   #1, LIB$SIGNAL                        ;
                                          04  11 00093          BRB     6$                                    ; 1903
                                     50   01  D0 00095 5$:      MOVL    #1, R0                                ; 1908
                                              04 00098          RET                                           ;
                                          50  D4 00099 6$:      CLRL    R0                                    ; 1909
                                              04 0009B          RET                                           ;

; Routine Size:  156 bytes,    Routine Base:  $CODE$ + 0D54
```

```
1 %sbttl 'getpriv - get process privileges'
1 global routine getpriv (qual_name, privadr) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Process privilege mask specification and set or
2 !       clear proper bits in privilege quadword in RECBUF.
2 !
2 ! INPUTS:
2 !
2 !       qual_name - descriptor of qualifier name to use
2 !       privadr - address to store privilege bits
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       none
2 !
2 ! OUTPUTS:
2 !
2 !       none
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> input processed successfully
2 !       false -> error in privilege specification
2 !
2 ! SIDE EFFECTS:
2 !
2 !       none
2 !--
2
2 !
2 ! Loop through if this is a list and check all of the names.
2 ! Set or clear appropriate bits depending on the 'NO' prefix.
2 !
2
2 while cli$get_value (.qual_name, tokendsc) do
3     begin
3
3     select prv$setpriv (tokendsc, .privadr) of
3
3         set
3         !
3         ! Privilege name not found
3         !
3         [prv$_invnam]:
3             return LIB$SIGNAL(UAF$_PRVNOTFND, 2, .tokenlen, .tokenptr);
3
3         !
3         ! Privilege name not unique
```

```
; 1888     1967  3          !
; 1889     1968  3          [prv$_notunq]:
; 1890     1969  3              return LIB$SIGNAL(UAF$_PRVNOTUNQ, 2, .tokenlen, .tokenptr);
; 1891     1970  3
; 1892     1971  3      tes;
; 1893     1972  3
; 1894     1973  2      end;
; 1895     1974  2
; 1896     1975  2  return true;
; 1897     1976  1  end;
```

```
                        003C 00000     .ENTRY  GETPRIV, Save R2,R3,R4,R5        ; 1911
          55 00000000G 00 9E 00002     MOVAB   TOKENDSC, R5                     ;
          54 00000000G 00 9E 00009     MOVAB   TOKENLEN, R4                     ;
          53 00000000G 00 9E 00010     MOVAB   TOKENPTR, R3                     ;
                       55 DD 00017 1$: PUSHL   R5                               ; 1953
                    04 AC DD 00019     PUSHL   QUAL_NAME                        ;
00000000G 00           02 FB 0001C     CALLS   #2, CLI$GET_VALUE                ;
          45           50 E9 00023     BLBC    R0, 4$                           ;
                    08 AC DD 00026     PUSHL   PRIVADR                          ; 1956
                       55 DD 00029     PUSHL   R5                               ;
00000000G 00           02 FB 0002B     CALLS   #2, PRV$SETPRIV                  ;
          52           50 D0 00032     MOVL    R0, R2                           ;
00000000G 8F           52 D1 00035     CMPL    R2, #PRV$_INVNAM                 ; 1962
                       0F 12 0003C     BNEQ    2$                               ;
                       63 DD 0003E     PUSHL   TOKENPTR                         ; 1963
          7E           64 3C 00040     MOVZWL  TOKENLEN, -(SP)                  ;
                       02 DD 00043     PUSHL   #2                               ;
             00000000G 8F DD 00045     PUSHL   #UAF$_PRVNOTFND                  ;
                       16 11 0004B     BRB     3$                               ;
00000000G 8F           52 D1 0004D 2$: CMPL    R2, #PRV$_NOTUNQ                 ; 1968
                       C1 12 00054     BNEQ    1$                               ;
                       63 DD 00056     PUSHL   TOKENPTR                         ; 1969
          7E           64 3C 00058     MOVZWL  TOKENLEN, -(SP)                  ;
                       02 DD 0005B     PUSHL   #2                               ;
             00000000G 8F DD 0005D     PUSHL   #UAF$_PRVNOTUNQ                  ;
00000000G 00           04 FB 00063 3$: CALLS   #4, LIB$SIGNAL                   ;
                          04 0006A     RET                                      ;
          50           01 D0 0006B 4$: MOVL    #1, R0                           ; 1975
                          04 0006E     RET                                      ; 1976

; Routine Size:  111 bytes,    Routine Base:  $CODE$ + 0DF0
```

```
; 1899          1977  1 %sbttl 'getuic - get user identification code'
; 1900          1978  1 global routine getuic =
; 1901          1979  2 begin
; 1902          1980  2
; 1903          1981  2 !++
; 1904          1982  2 !
; 1905          1983  2 ! FUNCTIONAL DESCRIPTION:
; 1906          1984  2 !
; 1907          1985  2 !       Process UIC specification
; 1908          1986  2 !
; 1909          1987  2 ! INPUTS:
; 1910          1988  2 !
; 1911          1989  2 !       none
; 1912          1990  2 !
; 1913          1991  2 ! OUTPUTS:
; 1914          1992  2 !
; 1915          1993  2 !       none
; 1916          1994  2 !
; 1917          1995  2 ! ROUTINE VALUE:
; 1918          1996  2 !
; 1919          1997  2 !       none
; 1920          1998  2 !--
; 1921          1999  2
; 1922          2000  2 !
; 1923          2001  2 ! Parse UIC and return error if invalid.
; 1924          2002  2 ! Do not allow wild card group or member numbers
; 1925          2003  2 !
; 1926          2004  2
; 1927          2005  2 if not parse_uic (recbuf[uaf$w_grp], recbuf[uaf$w_mem], false)
; 1928          2006  2 then return LIB$SIGNAL(UAF$_UICERR, 2, .tokenlen, .tokenptr);
; 1929          2007  2
; 1930          2008  2 true
; 1931          2009  1 end;



                                                      0000 00000           .ENTRY   GETUIC, Save nothing                       ; 1978
                                                     7E D4 00002           CLRL     -(SP)                                      ; 2005
                                           00000000G 00 9F 00004           PUSHAB   RECBUF+36                                  ;
                                           00000000G 00 9F 0000A           PUSHAB   RECBUF+38                                  ;
                            00000000V   00           03 FB 00010           CALLS    #3, PARSE_UIC                              ;
                                        1D           50 E8 00017           BLBS     R0, 1$                                     ;
                                           00000000G 00 DD 0001A           PUSHL    TOKENPTR                                   ; 2006
                                        7E 00000000G 00 3C 00020           MOVZWL   TOKENLEN, -(SP)                            ;
                                                     02 DD 00027           PUSHL    #2                                         ;
                                           00000000G 8F DD 00029           PUSHL    #UAF$_UICERR                               ;
                            00000000G   00           04 FB 0002F           CALLS    #4, LIB$SIGNAL                             ;
                                                        04 00036           RET                                                 ;
                                        50           01 D0 00037 1$:       MOVL     #1, R0                                     ; 2009
                                                        04 0003A           RET                                                 ;

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 0E5F
```

```
; 1933          2010  1 %sbttl 'parse_uic - obtain UIC group and member'
; 1934          2011  1 global routine parse_uic (retgrp, retmem, allow_wild) =
; 1935          2012  2 begin
; 1936          2013  2
; 1937          2014  2 !++
; 1938          2015  2 !
; 1939          2016  2 ! FUNCTIONAL DESCRIPTION:
; 1940          2017  2 !
; 1941          2018  2 !       Routine to parse and validate a UIC or directory string.
; 1942          2019  2 !       The UIC is returned as group and member elements.
; 1943          2020  2 !
; 1944          2021  2 ! INPUTS:
; 1945          2022  2 !
; 1946          2023  2 !       RETGRP          - address of a word to receive the group number
; 1947          2024  2 !       RETMEM          - address of a word to receive the member number
; 1948          2025  2 !       ALLOW_WILD      - TRUE => allow wild card group/member numbers
; 1949          2026  2 !
; 1950          2027  2 ! IMPLICIT INPUTS:
; 1951          2028  2 !
; 1952          2029  2 !       TOKENPTR - address of first character past delimiter
; 1953          2030  2 !
; 1954          2031  2 ! OUTPUTS:
; 1955          2032  2 !
; 1956          2033  2 !       RETGRP and RETMEM receive group and member numbers if
; 1957          2034  2 !       no errors are encountered.
; 1958          2035  2 !
; 1959          2036  2 ! IMPLICIT OUTPUTS:
; 1960          2037  2 !
; 1961          2038  2 !       none
; 1962          2039  2 !
; 1963          2040  2 ! ROUTINE VAUE:
; 1964          2041  2 !
; 1965          2042  2 !       true -> no errors found
; 1966          2043  2 !       false -> error in UIC specification
; 1967          2044  2 !
; 1968          2045  2 ! SIDE EFFECTS:
; 1969          2046  2 !
; 1970          2047  2 !       None
; 1971          2048  2 !--
; 1972          2049  2
; 1973          2050  2 !
; 1974          2051  2 ! Crank UIC through tparse
; 1975          2052  2 !
; 1976          2053  2 tparse_block [tpa$l_stringcnt] = .tokenlen;
; 1977          2054  2 tparse_block [tpa$l_stringptr] = .tokenptr;
; 1978          2055  2
; 1979          2056  2 if not lib$tparse (tparse_block, uic_states, uic_keys)
; 1980          2057  2 then return false;
; 1981          2058  2 if .converted_uic<00,16> gtru uic$k_wild_member
; 1982          2059  2 or .converted_uic<16,16> gtru uic$k_wild_group
; 1983          2060  2 then return false;
; 1984          2061  2 if not .allow_wild
; 1985          2062  2 then
; 1986          2063  2     if .converted_uic<00,16> eqlu uic$k_wild_member
; 1987          2064  2     or .converted_uic<16,16> eqlu uic$k_wild_group
; 1988          2065  2     then return false;
; 1989          2066  2
```

```
; 1990          2067  2  !
; 1991          2068  2  ! UIC was successfully parsed.  Return correct data, plus true indicaton.
; 1992          2069  2  !
; 1993          2070  2
; 1994          2071  2  (.retgrp)<0,16> = .converted_uic<16,16>;
; 1995          2072  2  (.retmem)<0,16> = .converted_uic<0,16>;
; 1996          2073  2  return true;
; 1997          2074  1  end;



                                         0004 00000             .ENTRY  PARSE_UIC, Save R2                          ; 2011
                   52 00000000'    00    9E 00002               MOVAB   CONVERTED_UIC, R2                           ;
              E4   A2 00000000G    00    3C 00009               MOVZWL  TOKENLEN, TPARSE_BLOCK+8                    ; 2053
              E8   A2 00000000G    00    D0 00011               MOVL    TOKENPTR, TPARSE_BLOCK+12                   ; 2054
                      00000000'    00    9F 00019               PUSHAB  UIC_KEYS                                    ; 2056
                      00000000'    00    9F 0001F               PUSHAB  UIC_STATES                                  ;
                             DC    A2    9F 00025               PUSHAB  TPARSE_BLOCK                                ;
         00000000G    00           03    FB 00028               CALLS   #3, LIB$TPARSE                              ;
                      29           50    E9 0002F               BLBC    R0, 2$                                      ;
                      50     02    A2    3C 00032               MOVZWL  CONVERTED_UIC+2, R0                         ; 2059
            3FFF      8F           50    B1 00036               CMPW    R0, #16383                                  ;
                                   1E    1A 0003B               BGTRU   2$                                          ;
              0E             0C    AC    E8 0003D               BLBS    ALLOW_WILD, 1$                              ; 2061
            FFFF      8F           62    B1 00041               CMPW    CONVERTED_UIC, #65535                       ; 2063
                                   13    13 00046               BEQL    2$                                          ;
            3FFF      8F           50    B1 00048               CMPW    R0, #16383                                  ; 2064
                                   0C    13 0004D               BEQL    2$                                          ;
              04      BC           50    B0 0004F 1$:           MOVW    R0, @RETGRP                                 ; 2071
              08      BC           62    B0 00053               MOVW    CONVERTED_UIC, @RETMEM                      ; 2072
                      50           01    D0 00057               MOVL    #1, R0                                      ; 2073
                                         04 0005A               RET                                                 ;
                                   50    D4 0005B 2$:           CLRL    R0                                          ; 2074
                                         04 0005D               RET                                                 ;

; Routine Size:  94 bytes,    Routine Base:  $CODE$ + 0E9A
```

```
; 1999      2075  1 %sbttl 'parse_wild - parse wildcarded user specification'
; 2000      2076  1 global routine parse_wild (desc_addr,nulldefault) =
; 2001      2077  2 begin
; 2002      2078  2
; 2003      2079  2 !++
; 2004      2080  2 !
; 2005      2081  2 ! FUNCTIONAL DESCRIPTION:
; 2006      2082  2 !
; 2007      2083  2 !       Parse one of six methods by which User Authorization File
; 2008      2084  2 !       records may be specified.  This routine defines boolean
; 2009      2085  2 !       variables for input to the WILD_USER routine.
; 2010      2086  2 !
; 2011      2087  2 ! INPUTS:
; 2012      2088  2 !
; 2013      2089  2 !       DESCADDR        Address of the token string descriptor
; 2014      2090  2 !       NULLDEFAULT     TRUE => A null user specification defaults to *
; 2015      2091  2 !
; 2016      2092  2 ! IMPLICIT INPUTS:
; 2017      2093  2 !
; 2018      2094  2 !       The parsing variables NEXTTOKEN, TOKENLEN, and TOKENPTR are modified.
; 2019      2095  2 !
; 2020      2096  2 ! OUTPUTS:
; 2021      2097  2 !
; 2022      2098  2 !       none
; 2023      2099  2 !
; 2024      2100  2 ! IMPLICIT OUTPUTS:
; 2025      2101  2 !
; 2026      2102  2 !       The following boolean variables are decided:
; 2027      2103  2 !
; 2028      2104  2 !       UIC_FLAG - UIC form (instead of username)
; 2029      2105  2 !       GRP_WILD - Group wild card (implies UIC_FLAG)
; 2030      2106  2 !       MEM_WILD - Member wild card (implies UIC_FLAG)
; 2031      2107  2 !       STR_WILD - all users alphabetically (implies NOT UIC_FLAG)
; 2032      2108  2 !
; 2033      2109  2 !       RECBUF - The appropriate keys are initialized
; 2034      2110  2 !
; 2035      2111  2 ! ROUTINE VALUE:
; 2036      2112  2 !
; 2037      2113  2 !       If the syntax does not follow one of the six methods the routine
; 2038      2114  2 !       returns FALSE.  The outputs are meaningful only when the value
; 2039      2115  2 !       returned is TRUE.
; 2040      2116  2 !
; 2041      2117  2 !--
; 2042      2118  2
; 2043      2119  2
; 2044      2120  2 ! Identify the next token in the command buffer.
; 2045      2121  2 !
; 2046      2122  2
; 2047      2123  2 if not cli$present (.desc_addr) or
; 2048      2124  2    not cli$get_value (.desc_addr, tokendsc) or
; 2049      2125  2    .tokenlen eql 0
; 2050      2126  2 then
; 2051      2127  2     if not .nulldefault
; 2052      2128  2     then
; 2053      2129  2         return LIB$SIGNAL(UAF$_NOUSERSPEC)
; 2054      2130  2     else
; 2055      2131  3         begin
```

```
; 2056          2132  3             uic_flag = false;
; 2057          2133  3             grp_wild = false;
; 2058          2134  3             mem_wild = false;
; 2059          2135  3             str_wild = true;
; 2060          2136  3             match_token = '*';
; 2061          2137  3             match_tokenlen = 1;
; 2062          2138  3             return true;
; 2063          2139  3             end
; 2064          2140  2     else
; 2065          2141  3         begin
; 2066          2142  3         tokenlen = .tokenlen;
; 2067          2143  3         tokenptr = .tokenptr;
; 2068          2144  2         end;
; 2069          2145  2
; 2070          2146  2
; 2071          2147  2     !
; 2072          2148  2     ! Decide whether a UIC or a Username form was used.
; 2073          2149  2     !
; 2074          2150  2
; 2075          2151  2     if .(.tokenptr)<0,8> eql '['
; 2076          2152  2     then
; 2077          2153  3         begin
; 2078          2154  3
; 2079          2155  3         if not parse_wild_uic ( .tokenlen, .tokenptr )
; 2080          2156  3         then
; 2081          2157  3             return LIB$SIGNAL(UAF$_INVUSERSPEC, 2, .tokenlen, .tokenptr);
; 2082          2158  3
; 2083          2159  3         end
; 2084          2160  2     else
; 2085          2161  3         begin
; 2086          2162  3         uic_flag = false;
; 2087          2163  3         grp_wild = false;
; 2088          2164  3         mem_wild = false;
; 2089          2165  3
; 2090          2166  3         !
; 2091          2167  3         ! Decide whether or not the Username is wildcarded.
; 2092          2168  3         !
; 2093          2169  3
; 2094          2170  3         if not ch$fail (ch$find_ch (.tokenlen, .tokenptr, '*'))
; 2095          2171  3                or
; 2096          2172  3            not ch$fail (ch$find_ch (.tokenlen, .tokenptr, '%'))
; 2097          2173  3     !          OR
; 2098          2174  3     !       .by_account
; 2099          2175  3         then
; 2100          2176  4             begin
; 2101          2177  4             match_tokenlen = .tokenlen;
; 2102          2178  4             ch$move (.tokenlen, .tokenptr, match_token);
; 2103          2179  4             str_wild = true;
; 2104          2180  4             end
; 2105          2181  3         else
; 2106          2182  4             begin
; 2107          2183  4             str_wild = false;
; 2108          2184  4             ch$copy (.tokenlen, .tokenptr, %char (' '),
; 2109          2185  4                             uaf$s_username, recbuf[uaf$t_username]);
; 2110          2186  3             end;
; 2111          2187  2         end;
; 2112          2188  2
```

```
; 2113          2189  2 true
; 2114          2190  1 end;



                                                        0FFC 00000          .ENTRY   PARSE_WILD, Save R2,R3,R4,R5,R6,R7,R8,R9,-   ; 2076
                                                                                     R10,R11                                      :
                                        5B 00000000G   00 9E 00002          MOVAB    UIC_FLAG, R11                                :
                                        5A 00000000G   00 9E 00009          MOVAB    LIB$SIGNAL, R10                              :
                                        59 00000000G   00 9E 00010          MOVAB    STR_WILD, R9                                 :
                                        58 00000000G   00 9E 00017          MOVAB    TOKENLEN, R8                                 :
                                                  04   AC DD 0001E          PUSHL    DESC_ADDR                                    ; 2123
                             00000000G  00             01 FB 00021          CALLS    #1, CLI$PRESENT                              :
                                        17             50 E9 00028          BLBC     R0, 1$                                       :
                                           00000000G   00 9F 0002B          PUSHAB   TOKENDSC                                     ; 2124
                                                  04   AC DD 00031          PUSHL    DESC_ADDR                                    :
                             00000000G  00             02 FB 00034          CALLS    #2, CLI$GET_VALUE                            :
                                        04             50 E9 0003B          BLBC     R0, 1$                                       :
                                                       68 B5 0003E          TSTW     TOKENLEN                                     ; 2125
                                                       2F 12 00040          BNEQ     3$                                           :
                                        0A        08   AC E8 00042 1$:      BLBS     NULLDEFAULT, 2$                              ; 2127
                                           00000000G   8F DD 00046          PUSHL    #UAF$_NOUSERSPEC                             ; 2129
                                        6A             01 FB 0004C          CALLS    #1, LIB$SIGNAL                               :
                                                          04 0004F          RET                                                   ; 2131
                                                       6B D4 00050 2$:      CLRL     UIC_FLAG                                     ; 2132
                                           00000000G   00 D4 00052          CLRL     GRP_WILD                                     ; 2133
                                           00000000G   00 D4 00058          CLRL     MEM_WILD                                     ; 2134
                                        69             01 D0 0005E          MOVL     #1, STR_WILD                                 ; 2135
                             00000000G  00             2A D0 00061          MOVL     #42, MATCH_TOKEN                             ; 2136
                             00000000G  00             01 D0 00068          MOVL     #1, MATCH_TOKENLEN                           ; 2137
                                                       79 11 0006F          BRB      9$                                           ; 2138
                                        57             68 3C 00071 3$:      MOVZWL   TOKENLEN, R7                                 ; 2155
                                        56 00000000G   00 D0 00074          MOVL     TOKENPTR, R6                                 ; 2151
                                    5B  8F             66 91 0007B          CMPB     (R6), #91                                    :
                                                       23 12 0007F          BNEQ     4$                                           :
                                                       56 DD 00081          PUSHL    R6                                           ; 2155
                                                       57 DD 00083          PUSHL    R7                                           :
                             00000000V  00             02 FB 00085          CALLS    #2, PARSE_WILD_UIC                           :
                                        5B             50 E8 0008C          BLBS     R0, 9$                                       :
                                           00000000G   00 DD 0008F          PUSHL    TOKENPTR                                     ; 2157
                                        7E             68 3C 00095          MOVZWL   TOKENLEN, -(SP)                              :
                                                       02 DD 00098          PUSHL    #2                                           :
                                           00000000G   8F DD 0009A          PUSHL    #UAF$_INVUSERSPEC                            :
                                        6A             04 FB 000A0          CALLS    #4, LIB$SIGNAL                               :
                                                          04 000A3          RET                                                   :
                                                       6B D4 000A4 4$:      CLRL     UIC_FLAG                                     ; 2162
                                           00000000G   00 D4 000A6          CLRL     GRP_WILD                                     ; 2163
                                           00000000G   00 D4 000AC          CLRL     MEM_WILD                                     ; 2164
                          66            57             2A 3A 000B2          LOCC     #42, R7, (R6)                                ; 2170
                                                       02 12 000B6          BNEQ     5$                                           :
                                                       51 D4 000B8          CLRL     R1                                           :
                                                       51 D5 000BA 5$:      TSTL     R1                                           :
                                                       0C 12 000BC          BNEQ     7$                                           :
                          66            57             25 3A 000BE          LOCC     #37, R7, (R6)                                ; 2172
                                                       02 12 000C2          BNEQ     6$                                           :
```

```
                                                   51  D4 000C4       CLRL    R1                                              ;
                                                   51  D5 000C6 6$:   TSTL    R1                                              ;
                                                   14  13 000C8       BEQL    8$                                              ;
                               00000000G  00       57  D0 000CA 7$:   MOVL    R7, MATCH_TOKENLEN                              ; 2177
               00000000G  00              66       57  28 000D1       MOVC3   R7, (R6), MATCH_TOKEN                           ; 2178
                                          69       01  D0 000D9       MOVL    #1, STR_WILD                                    ; 2179
                                                   0C  11 000DC       BRB     9$                                              ; 2170
                                                   69  D4 000DE 8$:   CLRL    STR_WILD                                        ; 2183
          20            20                66       57  2C 000E0       MOVC5   R7, (R6), #32, #32, RECBUF+4                    ; 2185
                                  00000000G 00             000E5                                                              ;
                                          50       01  D0 000EA 9$:   MOVL    #1, R0                                          ; 2190
                                                       04 000ED       RET                                                     ;

; Routine Size:  238 bytes,    Routine Base:  $CODE$ + 0EF8
```

```
1 %sbttl 'parse_wild_uic - parse wildcarded UIC'
1 global routine parse_wild_uic ( strlen, strptr ) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Parse the uic methods by which User Authorization File
2 !       records may be specified.  This routine defines boolean
2 !       variables for input to the WILD_USER routine.
2 !
2 ! INPUTS:
2 !
2 !       STRLEN  - Length of UIC string
2 !       STRPTR  - Address of UIC string
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !
2 ! OUTPUTS:
2 !
2 !       True  - success
2 !       False - failure
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       The following boolean variables are decided:
2 !
2 !       UIC_FLAG - UIC form (instead of username)
2 !       GRP_WILD - Group wild card (implies UIC_FLAG)
2 !       MEM_WILD - Member wild card (implies UIC_FLAG)
2 !       STR_WILD - all users alphabetically (implies NOT UIC_FLAG)
2 !
2 !       RECBUF - The appropriate keys are initialized
2 !
2 ! ROUTINE VALUE:
2 !
2 !       If the syntax does not follow one of the six methods the routine
2 !       returns FALSE.  The outputs are meaningful only when the value
2 !       returned is TRUE.
2 !
2 !--
2
2 uic_flag = true;
2 str_wild = false;
2
2 !
2 ! Crank UIC through tparse
2 !
2 tparse_block [tpa$l_stringcnt] = .strlen;
2 tparse_block [tpa$l_stringptr] = .strptr;
2
2 if not lib$tparse (tparse_block, uic_states, uic_keys)
2 then
2     return false ;
2
```

```
; 2173          2248  2 !
; 2174          2249  2 ! Decide whether or not the Group subfield is wildcarded.
; 2175          2250  2 !
; 2176          2251  2 if .converted_uic<16,16> eql uic$k_wild_group
; 2177          2252  2 then
; 2178          2253  3     begin
; 2179          2254  3     grp_wild = true;
; 2180          2255  3     recbuf[uaf$w_grp] = 0;
; 2181          2256  3     end
; 2182          2257  3
; 2183          2258  2 else
; 2184          2259  3     begin
; 2185          2260  3     grp_wild = false;
; 2186          2261  3     recbuf[uaf$w_grp] = .converted_uic<16,16>;
; 2187          2262  2     end;
; 2188          2263  2
; 2189          2264  2 !
; 2190          2265  2 ! Decide whether or not the Member subfield is wildcarded.
; 2191          2266  2 !
; 2192          2267  2
; 2193          2268  2 if .converted_uic<0,16> eql uic$k_wild_member
; 2194          2269  2 then
; 2195          2270  3     begin
; 2196          2271  3     mem_wild = true;
; 2197          2272  3
; 2198          2273  3     !
; 2199          2274  3     ! Whether it is wildcarded or not this subfield is initialized
; 2200          2275  3     ! for the wildcard processor.
; 2201          2276  3     !
; 2202          2277  3
; 2203          2278  3     recbuf[uaf$w_mem] = 0;
; 2204          2279  3     end
; 2205          2280  2 else
; 2206          2281  3     begin
; 2207          2282  3     mem_wild = false;
; 2208          2283  3     recbuf[uaf$w_mem] = .converted_uic<0,16>;
; 2209          2284  2     end;
; 2210          2285  2
; 2211          2286  2 return true ;
; 2212          2287  1 end ;



                                      003C 00000          .ENTRY  PARSE_WILD_UIC, Save R2,R3,R4,R5                  ; 2192
                  55 00000000G    00    9E 00002          MOVAB   MEM_WILD, R5                                      ;
                  54 00000000G    00    9E 00009          MOVAB   GRP_WILD, R4                                      ;
                  53 00000000G    00    9E 00010          MOVAB   RECBUF+38, R3                                     ;
                  52 00000000'    00    9E 00017          MOVAB   CONVERTED_UIC, R2                                 ;
         00000000G    00              01    D0 0001E          MOVL    #1, UIC_FLAG                                      ; 2235
                     00000000G    00    D4 00025          CLRL    STR_WILD                                          ; 2236
            E4    A2          04    AC    7D 0002B          MOVQ    STRLEN, TPARSE_BLOCK+8                            ; 2241
                     00000000'    00    9F 00030          PUSHAB  UIC_KEYS                                          ; 2244
                     00000000'    00    9F 00036          PUSHAB  UIC_STATES                                        ;
                            DC    A2    9F 0003C          PUSHAB  TPARSE_BLOCK                                      ;
         00000000G    00              03    FB 0003F          CALLS   #3, LIB$TPARSE                                    ;
```

```
                         33              50 E9 00046           BLBC    R0, 5$                                                   ;
                         50       02     A2 3C 00049           MOVZWL  CONVERTED_UIC+2, R0                                      ; 2251
                  3FFF   8F              50 B1 0004D           CMPW    R0, #16383                                               ;
                                         07 12 00052           BNEQ    1$                                                       ;
                         64              01 D0 00054           MOVL    #1, GRP_WILD                                             ; 2254
                                         63 B4 00057           CLRW    RECBUF+38                                                ; 2255
                                         05 11 00059           BRB     2$                                                       ; 2251
                                         64 D4 0005B 1$:       CLRL    GRP_WILD                                                 ; 2260
                         63              50 B0 0005D           MOVW    R0, RECBUF+38                                            ; 2261
                         50              62 3C 00060 2$:       MOVZWL  CONVERTED_UIC, R0                                        ; 2268
                  FFFF   8F              50 B1 00063           CMPW    R0, #65535                                               ;
                                         08 12 00068           BNEQ    3$                                                       ;
                         65              01 D0 0006A           MOVL    #1, MEM_WILD                                             ; 2271
                                  FE     A3 B4 0006D           CLRW    RECBUF+36                                                ; 2278
                                         06 11 00070           BRB     4$                                                       ; 2268
                                         65 D4 00072 3$:       CLRL    MEM_WILD                                                 ; 2282
                    FE   A3              50 B0 00074           MOVW    R0, RECBUF+36                                            ; 2283
                         50              01 D0 00078 4$:       MOVL    #1, R0                                                   ; 2286
                                            04 0007B           RET                                                              ;
                                         50 D4 0007C 5$:       CLRL    R0                                                       ; 2287
                                            04 0007E           RET                                                              ;

; Routine Size:  127 bytes,    Routine Base:  $CODE$ + 0FE6
```

```
; 2214          2288  1 %sbttl 'getstring - get string from input'
; 2215          2289  1 global routine getstring (addr, maxsize, type) =
; 2216          2290  2 begin
; 2217          2291  2
; 2218          2292  2 !++
; 2219          2293  2 !
; 2220          2294  2 ! FUNCTIONAL DESCRIPTION:
; 2221          2295  2 !
; 2222          2296  2 !       This routine reads a string variable and stores it as
; 2223          2297  2 !       a blank filled or counted string in the field supplied.
; 2224          2298  2 !       The first character is checked to see if it is a quote
; 2225          2299  2 !       or double quote and if so, a different terminator
; 2226          2300  2 !       set is used so that embedded blanks may be contained
; 2227          2301  2 !       in the string.
; 2228          2302  2 !
; 2229          2303  2 ! INPUTS:
; 2230          2304  2 !
; 2231          2305  2 !       ADDR - address to store string
; 2232          2306  2 !       MAXSIZE - size of field in which string will be stored.
; 2233          2307  2 !               If this is a counted string, the maximum input
; 2234          2308  2 !               acceptable is MAXSIZE - 1
; 2235          2309  2 !       TYPE - string type, either COUNTED_STRING or FILLED_STRING
; 2236          2310  2 !
; 2237          2311  2 ! IMPLICIT INPUTS:
; 2238          2312  2 !
; 2239          2313  2 !       none
; 2240          2314  2 !
; 2241          2315  2 ! OUTPUTS:
; 2242          2316  2 !
; 2243          2317  2 !       none
; 2244          2318  2 !
; 2245          2319  2 ! IMPLICIT OUTPUTS:
; 2246          2320  2 !
; 2247          2321  2 !       none
; 2248          2322  2 !
; 2249          2323  2 ! ROUTINE VALUE:
; 2250          2324  2 !
; 2251          2325  2 !       true -> string inserted successfully
; 2252          2326  2 !       false -> string too long
; 2253          2327  2 !
; 2254          2328  2 ! SIDE EFFECTS:
; 2255          2329  2 !
; 2256          2330  2 !       none
; 2257          2331  2 !--
; 2258          2332  2
; 2259          2333  2
; 2260          2334  2 if .type eql counted_string
; 2261          2335  2 then
; 2262          2336  2     maxsize = .maxsize - 1;
; 2263          2337  2
; 2264          2338  2 if .tokenlen gtr .maxsize
; 2265          2339  2 then
; 2266          2340  2     return
; 2267          2341  2         LIB$SIGNAL(UAF$_INVSTR, 2, .tokenlen, .tokenptr);
; 2268          2342  2
; 2269          2343  2 if .type eql counted_string
; 2270          2344  2 then
```

```
; 2271          2345  3       begin
; 2272          2346  3       (.addr)<0,8> = .tokenlen;
; 2273          2347  3       ch$copy (.tokenlen, .tokenptr, %char (blank),
; 2274          2348  3                   .maxsize-1, .addr+1);
; 2275          2349  3       end
; 2276          2350  2   else
; 2277          2351  2       ch$copy (.tokenlen, .tokenptr, %char (blank),
; 2278          2352  2                   .maxsize, .addr);
; 2279          2353  2   return true;
; 2280          2354  1   end;



                                                       00FC 00000          .ENTRY  GETSTRING, Save R2,R3,R4,R5,R6,R7           ; 2289
                                       57 00000000G  00  9E 00002          MOVAB   TOKENLEN, R7                                ;
                                       56 00000000G  00  9E 00009          MOVAB   TOKENPTR, R6                                ;
                                                     54  D4 00010          CLRL    R4                                          ; 2334
                                       01       0C   AC  D1 00012          CMPL    TYPE, #1                                    ;
                                                     05  12 00016          BNEQ    1$                                          ;
                                                     54  D6 00018          INCL    R4                                          ;
                                                08   AC  D7 0001A          DECL    MAXSIZE                                     ; 2336
                                       50            67  3C 0001D 1$:      MOVZWL  TOKENLEN, R0                                ; 2338
                                  08   AC            50  D1 00020          CMPL    R0, MAXSIZE                                 ;
                                                     14  15 00024          BLEQ    2$                                          ;
                                                     66  DD 00026          PUSHL   TOKENPTR                                    ; 2341
                                                     50  DD 00028          PUSHL   R0                                          ;
                                                     02  DD 0002A          PUSHL   #2                                          ;
                                          00000000G  8F  DD 0002C          PUSHL   #UAF$_INVSTR                                ;
                            00000000G  00            04  FB 00032          CALLS   #4, LIB$SIGNAL                              ;
                                                         04 00039          RET                                                 ;
                                       50       04   AC  D0 0003A 2$:      MOVL    ADDR, R0                                    ; 2346
                                       52            67  3C 0003E          MOVZWL  TOKENLEN, R2                                ;
                                       53            66  D0 00041          MOVL    TOKENPTR, R3                                ; 2347
                                       11            54  E9 00044          BLBC    R4, 3$                                      ;
                                       60            52  90 00047          MOVB    R2, (R0)                                    ; 2346
                         51       08   AC            01  C3 0004A          SUBL3   #1, MAXSIZE, R1                             ; 2348
           51            20            63            52  2C 0004F          MOVC5   R2, (R3), #32, R1, 1(R0)                    ;
                                                01   A0     00054                                                              ;
                                                     07  11 00056          BRB     4$                                          ; 2343
      08   AC            20            63            52  2C 00058 3$:      MOVC5   R2, (R3), #32, MAXSIZE, (R0)                ; 2352
                                                     60     0005E                                                              ;
                                       50            01  D0 0005F 4$:      MOVL    #1, R0                                      ; 2353
                                                         04 00062          RET                                                 ; 2354

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 1065
```

```
1 %sbttl 'getval - get numeric value from user'
1 global routine getval (addr, size) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       Routine to return binary value for next ASCII decimal
2 !       number in the input stream.
2 !
2 ! INPUTS:
2 !
2 !       ADDR - address to return converted value
2 !       SIZE - size in bits of field in which to return value
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       NEXTTOKEN - contains address of delimiter preceeding the number
2 !
2 ! OUTPUTS:
2 !
2 !       field described by ADDR and SIZE receives binary value
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> value converted successfully
2 !       false -> non-numeric character encounted before next
2 !               delimiter found
2 !
2 ! SIDE EFFECTS:
2 !
2 !       If error is encountered, an error message will be printed.
2 !
2 !--
2
2 local
2       value;                                  ! for value returned
2                                               ! by CVTNUM routine
2
2 !
2 ! Check for no value supplied
2 !
2
2 if .tokenlen eql 0
2 then
2     return LIB$SIGNAL(UAF$_NOARG, 2, .tokenlen, .tokenptr);
2
2 !
2 ! Convert to decimal. Report error if any non-numerics encountered.
2 !
2
2 if not cvtnum (.tokenlen, .tokenptr, 10, value)
```

```
; 2339          2412  2 then
; 2340          2413  2     return LIB$SIGNAL(UAF$_BADVALUE, 2, .tokenlen, .tokenptr);
; 2341          2414  2
; 2342          2415  2 !
; 2343          2416  2 ! Check that the value supplied is within the size of
; 2344          2417  2 ! the field to be stored.
; 2345          2418  2 !
; 2346          2419  2
; 2347          2420  2 if .value<.size - 1, 32 - .size + 1> neq 0
; 2348          2421  2 then
; 2349          2422  2     return LIB$SIGNAL(UAF$_VALTOOBIG, 2, .tokenlen, .tokenptr);
; 2350          2423  2
; 2351          2424  2 !
; 2352          2425  2 ! Finally, store the value in field specified .
; 2353          2426  2 !
; 2354          2427  2
; 2355          2428  2 (.addr)<0, .size> = .value;
; 2356          2429  2
; 2357          2430  2 return true;
; 2358          2431  1 end;



                                                       001C 00000         .ENTRY  GETVAL, Save R2,R3,R4                 ; 2356
                                       54 00000000G  00  9E 00002         MOVAB   TOKENLEN, R4                          ;
                                       53 00000000G  00  9E 00009         MOVAB   TOKENPTR, R3                          ;
                                       5E            04  C2 00010         SUBL2   #4, SP                                ;
                                       50            64  3C 00013         MOVZWL  TOKENLEN, R0                          ; 2403
                                                     0E  12 00016         BNEQ    1$                                    ;
                                                     63  DD 00018         PUSHL   TOKENPTR                              ; 2405
                                                     50  DD 0001A         PUSHL   R0                                    ;
                                                     02  DD 0001C         PUSHL   #2                                    ;
                                          00000000G  8F  DD 0001E         PUSHL   #UAF$_NOARG                           ;
                                                     40  11 00024         BRB     3$                                    ;
                                                     5E  DD 00026 1$:     PUSHL   SP                                    ; 2411
                                                     0A  DD 00028         PUSHL   #10                                   ;
                                                     63  DD 0002A         PUSHL   TOKENPTR                              ;
                                       7E            64  3C 0002C         MOVZWL  TOKENLEN, -(SP)                       ;
                            00000000V  00            04  FB 0002F         CALLS   #4, CVTNUM                            ;
                                       0F            50  E8 00036         BLBS    R0, 2$                                ;
                                                     63  DD 00039         PUSHL   TOKENPTR                              ; 2413
                                       7E            64  3C 0003B         MOVZWL  TOKENLEN, -(SP)                       ;
                                                     02  DD 0003E         PUSHL   #2                                    ;
                                          00000000G  8F  DD 00040         PUSHL   #UAF$_BADVALUE                        ;
                                                     1E  11 00046         BRB     3$                                    ;
                         52       08   AC            01  C3 00048 2$:     SUBL3   #1, SIZE, R2                          ; 2420
                         50            21       08   AC  C3 0004D         SUBL3   SIZE, #33, R0                         ;
           51            6E            50            52  EF 00052         EXTZV   R2, R0, VALUE, R1                     ;
                                                     15  13 00057         BEQL    4$                                    ;
                                                     63  DD 00059         PUSHL   TOKENPTR                              ; 2422
                                       7E            64  3C 0005B         MOVZWL  TOKENLEN, -(SP)                       ;
                                                     02  DD 0005E         PUSHL   #2                                    ;
                                          00000000G  8F  DD 00060         PUSHL   #UAF$_VALTOOBIG                       ;
                            00000000G  00            04  FB 00066 3$:     CALLS   #4, LIB$SIGNAL                        ;
                                                         04 0006D         RET                                           ;
```

```
     04    BC        08    AC               00          6E  F0 0006E 4$:     INSV     VALUE, #0, SIZE, @ADDR                          ; 2428
                                            50          01  D0 00075         MOVL     #1, R0                                          ; 2430
                                                            04 00078         RET                                                      ; 2431

; Routine Size:  121 bytes,    Routine Base:  $CODE$ + 10C8
```

```
1 %sbttl 'cvtnum - convert decimal ascii to binary'
1 routine cvtnum (size, adr, radix, valadr) =
2 begin
2
2 !++
2 !
2 ! FUNCTIONAL DESCIRPTION:
2 !
2 !       Routine to convert ascii digits to binary.
2 !
2 ! INPUTS:
2 !
2 !       SIZE - number of digits in input buffer
2 !       ADR - address of the buffer containing ascii digits
2 !       RADIX - radix of number to be converted
2 !       VALADR - address of longword to receive converted value
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !       none
2 !
2 ! OUTPUTS:
2 !
2 !       longword pointed to by VALADR receives converted value
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       none
2 !
2 ! ROUTINE VALUE:
2 !
2 !       true -> value successfully converted
2 !       false -> non-numeric encountered before end of string
2 !
2 ! SIDE EFFECTS:
2 !
2 !       none
2 !--
2
2 bind
2       sum = .valadr;                          ! address ref scaler
2
2
2 sum = 0;
2
2 !
2 ! Loop thorugh buffer.  Stop on end of string or non-decimal digit.
2 !
2
3 incr i from .adr to  (.adr + .size - 1)
2 do
3     begin
3     local
3         digit,
3         pointer;
3
3     digit = .(.i)<0,8>;
```

```
; 2417          2489  3
; 2418          2490  3      !
; 2419          2491  3      ! Check validity of digit depending on radix
; 2420          2492  3      !
; 2421          2493  3
; 2422          2494  3
; 2423          2495  3      if ch$fail (pointer = ch$find_ch (.radix, numbers, .digit))
; 2424          2496  3      then
; 2425          2497  3          return false
; 2426          2498  3      else
; 2427          2499  4          sum = (.sum*.radix) +  (.pointer - numbers)
; 2428          2500  2      end;
; 2429          2501  2
; 2430          2502  2 return true;
; 2431          2503  1 end;




                                                         003C 00000 CVTNUM: .WORD    Save R2,R3,R4,R5                             ; 2433
                                        55 00000000'   00  9E 00002         MOVAB    NUMBERS, R5                                  ;
                                                  10   BC  D4 00009         CLRL     @VALADR                                      ; 2475
                          54       08   AC        04   AC  C1 0000C         ADDL3    SIZE, ADR, R4                                ; 2481
                          53       08   AC             01  C3 00012         SUBL3    #1, ADR, I                                   ; 2495
                                                       21  11 00017         BRB      3$                                           ;
                                        50             63  9A 00019 1$:     MOVZBL   (I), DIGIT                                   ; 2488
                          65       0C   AC             50  3A 0001C         LOCC     DIGIT, RADIX, NUMBERS                        ; 2495
                                                       02  12 00021         BNEQ     2$                                           ;
                                                       51  D4 00023         CLRL     R1                                           ;
                                                       51  D5 00025 2$:     TSTL     POINTER                                      ;
                                                       19  13 00027         BEQL     4$                                           ;
                          52       10   BC        0C   AC  C5 00029         MULL3    RADIX, @VALADR, R2                           ; 2499
                                        50             65  9E 0002F         MOVAB    NUMBERS, R0                                  ;
                                        51             50  C2 00032         SUBL2    R0, R1                                       ;
                     10   BC            52             51  C1 00035         ADDL3    R1, R2, @VALADR                              ;
                          DB            53             54  F2 0003A 3$:     AOBLSS   R4, I, 1$                                    ; 2495
                                        50             01  D0 0003E         MOVL     #1, R0                                       ; 2502
                                                           04 00041         RET                                                   ;
                                                       50  D4 00042 4$:     CLRL     R0                                           ; 2503
                                                           04 00044         RET                                                   ;

; Routine Size:  69 bytes,    Routine Base:  $CODE$ + 1141
```

```
  1 %sbttl 'PASSWORD MACROS -- for N-ary passwords'
  1
  1 !++
  1 !
  1 ! FUNCTIONAL DESCRIPTION:
  1 !
  1 !       These macros and the routines that follow allow for
  1 !       the use of any number of passwords. At the time of
  1 !       this writing, only two (primary and secondary) pass-
  1 !       words were in effect. When a change is made to the
  1 !       number of passwords, it will be necessary, only, to
  1 !       update the argument list macros below.
  1 !
  1 !--
  1
  1 macro
  1
  1     $NARY_LIST   = 'PRIMARY', '', 'SECONDARY', '2' %,
  1     $N_LIST      =            '',              '2' %;
  1
  1 !
  1 !       The remainder of the text will need no modification for N-ary passwords
  1 !
  1
  1 macro $PASSWORD(n) = %name('PASSWORD_',n) %;
  1 macro $PRESENT(n) = %name('PRESENT_',n) %;
  1
M 1 macro OWN_PASSWORD[n] =
M 1     own $PASSWORD(n): block[DSC$K_D_BLN, byte]
  1         preset([DSC$B_CLASS] = DSC$K_CLASS_D); %;
  1
  1 OWN_PASSWORD($N_LIST);
```

```
; 2466          2536 1 %sbttl 'UAF$GENERATE -- Generate random passwords'
; 2467          2537 1 routine UAF$GENERATE =
; 2468          2538 2 begin
; 2469          2539 2
; 2470          2540 2 !++
; 2471          2541 2 !
; 2472          2542 2 ! FUNCTIONAL DESCRIPTION:
; 2473          2543 2 !
; 2474          2544 2 !       Decide which passwords to generate
; 2475          2545 2 !
; 2476          2546 2 ! INPUTS:
; 2477          2547 2 !
; 2478          2548 2 !       none
; 2479          2549 2 !
; 2480          2550 2 ! IMPLICIT INPUTS:
; 2481          2551 2 !
; 2482          2552 2 !       CLI$GET_VALUE(%ascid'GENERATE_PASSWORD')
; 2483          2553 2 !
; 2484          2554 2 ! OUTPUTS:
; 2485          2555 2 !
; 2486          2556 2 !       none
; 2487          2557 2 !
; 2488          2558 2 ! IMPLICIT OUTPUTS:
; 2489          2559 2 !
; 2490          2560 2 !       Places new values in $PASSWORD(n)'s
; 2491          2561 2 !
; 2492          2562 2 ! ROUTINE VALUE:
; 2493          2563 2 !
; 2494          2564 2 !       none
; 2495          2565 2 !
; 2496          2566 2 ! SIDE EFFECTS:
; 2497          2567 2 !
; 2498          2568 2 !       none
; 2499          2569 2 !
; 2500          2570 2 !--
; 2501          2571 2
; 2502          2572 2     own WHICH: block[DSC$K_D_BLN, byte] preset([DSC$B_CLASS] = DSC$K_CLASS_D);
; 2503          2573 2
; 2504        M 2574 2     macro ACTIVE_PASSWORD(n) =
; 2505          2575 2        ch$neq(%name('UAF$S_PWD',n), RECBUF[%name('UAF$Q_PWD',n)], 0, 0, 0) %;
; 2506        M 2576 2     macro WHICH_EQL[THIS] =
; 2507        M 2577 2        ch$eql( .WHICH[DSC$W_LENGTH], .WHICH[DSC$A_POINTER],
; 2508          2578 2                  %charcount(THIS), uplit(%ascii %string(THIS)) ) %;
; 2509        M 2579 2     macro GENERATE_NEW_PASSWORD[THIS, n] =
; 2510        M 2580 2        if (ACTIVE_PASSWORD(n) and (FALSE or WHICH_EQL('CURRENT')))
; 2511        M 2581 2        or WHICH_EQL(THIS, 'BOTH', 'ALL') then
; 2512        M 2582 2           AUTHORIZE_GENERATE($PASSWORD(n), uplit(%ascii THIS), %charcount(THIS))
; 2513        M 2583 2        else
; 2514          2584 2           $PASSWORD(n)[DSC$W_LENGTH] = 0; %;
; 2515          2585 2
; 2516          2586 2     CLI$GET_VALUE(%ascid'GENERATE_PASSWORD', WHICH);
; 2517          2587 2     GENERATE_NEW_PASSWORD($NARY_LIST);
; 2518          2588 2
; 2519          2589 2     return TRUE;
; 2520          2590 2
; 2521          2591 1 end;
```

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

4F  57  53  53  41  50  5F  45  54  41  52  45  4E  45  47  004D0 P.AFH:  .ASCII  \GENERATE_PASSWORD\<0><0><0>                ;
                                        00  00  00  44  52  004DF                                                             :
                                                  010E0011  004E4 P.AFG:  .LONG   17694737                                    :
                                                  00000000' 004E8         .ADDRESS P.AFH                                      :
                            00  54  4E  45  52  52  55  43  004EC P.AFI:  .ASCII  \CURRENT\<0>                                :
                            00  59  52  41  4D  49  52  50  004F4 P.AFJ:  .ASCII  \PRIMARY\<0>                                :
                                            48  54  4F  42  004FC P.AFK:  .ASCII  \BOTH\                                      :
                                            00  4C  4C  41  00500 P.AFL:  .ASCII  \ALL\<0>                                    :
                            00  59  52  41  4D  49  52  50  00504 P.AFM:  .ASCII  \PRIMARY\<0>                                :
                            00  54  4E  45  52  52  55  43  0050C P.AFN:  .ASCII  \CURRENT\<0>                                :
            00  00  00  59  52  41  44  4E  4F  43  45  53  00514 P.AFO:  .ASCII  \SECONDARY\<0><0><0>                        :
                                            48  54  4F  42  00520 P.AFP:  .ASCII  \BOTH\                                      :
                                            00  4C  4C  41  00524 P.AFQ:  .ASCII  \ALL\<0>                                    :
            00  00  00  59  52  41  44  4E  4F  43  45  53  00528 P.AFR:  .ASCII  \SECONDARY\<0><0><0>                        :

                                                                          .PSECT  $OWN$,NOEXE,2

                                                        00# 0010C PASSWORD_:
                                                                          .BYTE   0[3]                                        ;
                                                        02  0010F         .BYTE   2                                           :
                                                            00110         .BLKB   4
                                                        00# 00114 PASSWORD_2:
                                                                          .BYTE   0[3]                                        ;
                                                        02  00117         .BYTE   2                                           :
                                                            00118         .BLKB   4
                                                        00# 0011C WHICH:  .BYTE   0[3]                                        ;
                                                        02  0011F         .BYTE   2                                           :
                                                            00120         .BLKB   4


                                                                          .PSECT  $CODE$,NOWRT,2

                                                       007C 00000 UAF$GENERATE:
                                                                          .WORD   Save R2,R3,R4,R5,R6                         ; 2537
                                        56 00000000V  00  9E  00002       MOVAB   AUTHORIZE_GENERATE, R6                      :
                                        55 00000000'  00  9E  00009       MOVAB   P.AFG, R5                                   :
                                        54 00000000'  00  9E  00010       MOVAB   WHICH, R4                                   :
                                                      54  DD  00017       PUSHL   R4                                          ; 2586
                                                      55  DD  00019       PUSHL   R5                                          :
                              00000000G 00            02  FB  0001B       CALLS   #2, CLI$GET_VALUE                           :
            00             00 00000000G 00            08  2D  00022       CMPC5   #8, RECBUF+340, #0, #0, @#^X00000000        ; 2587
                                           00000000   9F      0002B                                                           :
                                                      0D  13  00030       BEQL    1$                                          :
                                        50      04    A4  D0  00032       MOVL    WHICH+4, R0                                 :
            07             00           60            64  2D  00036       CMPC5   WHICH, (R0), #0, #7, P.AFI                  :
                                                08    A5      0003B                                                           :
                                                      27  13  0003D       BEQL    2$                                          :
                                        50      04    A4  D0  0003F 1$:   MOVL    WHICH+4, R0                                 :
            07             00           60            64  2D  00043       CMPC5   WHICH, (R0), #0, #7, P.AFJ                  :
                                                10    A5      00048                                                           :
                                                      1A  13  0004A       BEQL    2$                                          :
                                        50      04    A4  D0  0004C       MOVL    WHICH+4, R0                                 :
```

```
        04          00            60           64  2D 00050          CMPC5   WHICH, (R0), #0, #4, P.AFK                      ;
                                        18     A5     00055                                                                  ;
                                               0D  13 00057          BEQL    2$                                              ;
                                  50    04     A4  D0 00059          MOVL    WHICH+4, R0                                     ;
        03          00            60           64  2D 0005D          CMPC5   WHICH, (R0), #0, #3, P.AFL                      ;
                                        1C     A5     00062                                                                  ;
                                               0D  12 00064          BNEQ    3$                                              ;
                                               07  DD 00066 2$:      PUSHL   #7                                              ;
                                        20     A5  9F 00068          PUSHAB  P.AFM                                           ;
                                        F0     A4  9F 0006B          PUSHAB  PASSWORD                                        ;
                                  66           03  FB 0006E          CALLS   #3, AUTHORIZE_GENERATE                          ;
                                               03  11 00071          BRB     4$                                              ;
                                        F0     A4  B4 00073 3$:      CLRW    PASSWORD                                        ;
        00          00 00000000G  00           08  2D 00076 4$:      CMPC5   #8, RECBUF+348, #0, #0, @#^X00000000            ;
                                  00000000     9F     0007F                                                                  ;
                                               0D  13 00084          BEQL    5$                                              ;
                                  50    04     A4  D0 00086          MOVL    WHICH+4, R0                                     ;
        07          00            60           64  2D 0008A          CMPC5   WHICH, (R0), #0, #7, P.AFN                      ;
                                        28     A5     0008F                                                                  ;
                                               27  13 00091          BEQL    6$                                              ;
                                  50    04     A4  D0 00093 5$:      MOVL    WHICH+4, R0                                     ;
        09          00            60           64  2D 00097          CMPC5   WHICH, (R0), #0, #9, P.AFO                      ;
                                        30     A5     0009C                                                                  ;
                                               1A  13 0009E          BEQL    6$                                              ;
                                  50    04     A4  D0 000A0          MOVL    WHICH+4, R0                                     ;
        04          00            60           64  2D 000A4          CMPC5   WHICH, (R0), #0, #4, P.AFP                      ;
                                        3C     A5     000A9                                                                  ;
                                               0D  13 000AB          BEQL    6$                                              ;
                                  50    04     A4  D0 000AD          MOVL    WHICH+4, R0                                     ;
        03          00            60           64  2D 000B1          CMPC5   WHICH, (R0), #0, #3, P.AFQ                      ;
                                        40     A5     000B6                                                                  ;
                                               0D  12 000B8          BNEQ    7$                                              ;
                                               09  DD 000BA 6$:      PUSHL   #9                                              ;
                                        44     A5  9F 000BC          PUSHAB  P.AFR                                           ;
                                        F8     A4  9F 000BF          PUSHAB  PASSWORD_2                                      ;
                                  66           03  FB 000C2          CALLS   #3, AUTHORIZE_GENERATE                          ;
                                               03  11 000C5          BRB     8$                                              ;
                                        F8     A4  B4 000C7 7$:      CLRW    PASSWORD_2                                      ;
                                  50           01  D0 000CA 8$:      MOVL    #1, R0                                          ; 2589
                                                   04 000CD          RET                                                     ; 2591

; Routine Size:  206 bytes,    Routine Base:  $CODE$ + 1186
```

```
; 2523          2592  1 %sbttl 'GETPASSWORD - get user password'
; 2524          2593  1 routine GETPASSWORD =
; 2525          2594  2 begin
; 2526          2595  2
; 2527          2596  2 !++
; 2528          2597  2 !
; 2529          2598  2 ! FUNCTIONAL DESCRIPTION:
; 2530          2599  2 !
; 2531          2600  2 !       Action routine to process password.
; 2532          2601  2 !
; 2533          2602  2 ! INPUTS:
; 2534          2603  2 !
; 2535          2604  2 !       none
; 2536          2605  2 !
; 2537          2606  2 ! IMPLICIT INPUTS:
; 2538          2607  2 !
; 2539          2608  2 !       none
; 2540          2609  2 !
; 2541          2610  2 ! OUTPUTS:
; 2542          2611  2 !
; 2543          2612  2 !       none
; 2544          2613  2 !
; 2545          2614  2 ! IMPLICIT OUTPUTS:
; 2546          2615  2 !
; 2547          2616  2 !       sets hashed password field in recbuf
; 2548          2617  2 !
; 2549          2618  2 ! ROUTINE VALUE:
; 2550          2619  2 !
; 2551          2620  2 !       true -> success
; 2552          2621  2 !       false -> failure
; 2553          2622  2 !
; 2554          2623  2 ! SIDE EFFECTS:
; 2555          2624  2 !
; 2556          2625  2 !       none
; 2557          2626  2 !--
; 2558          2627  2
; 2559          2628  2     own
; 2560          2629  2
; 2561          2630  2         USER_DSC: $bblock[8],
; 2562          2631  2         REC_ENCRYPT_DSC: $bblock[8];
; 2563          2632  2
; 2564        M 2633  2     macro LOCAL_PRESENT[n] =
; 2565          2634  2         local $PRESENT(n): initial(TRUE); %;
; 2566          2635  2
; 2567          2636  2     LOCAL_PRESENT($N_LIST);
; 2568          2637  2
; 2569          2638  2 !
; 2570          2639  2 !       What are the passwords ???
; 2571          2640  2 !
; 2572          2641  2
; 2573          2642  2     if CLI$PRESENT(SD_PASSWORD) eql CLI$_NEGATED then
; 2574          2643  3         begin
; 2575          2644  3             macro BLANK_PASSWORD[n] = $PRESENT(n) = FALSE; %;
; 2576          2645  3             BLANK_PASSWORD($N_LIST);
; 2577          2646  3         end
; 2578          2647  2     else if CLI$PRESENT(SD_PASSWORD) then
; 2579          2648  3         begin
```

```
; 2580      M 2649  3              macro GET_PASSWORD[n] =
; 2581      M 2650  3                  $PASSWORD(n)[DSC$W_LENGTH] = 0;
; 2582      M 2651  3                  $PASSWORD(n)[DSC$A_POINTER] = 0;
; 2583        2652  3                  $PRESENT(n) = CLI$GET_VALUE(SD_PASSWORD, $PASSWORD(n)); %;
; 2584        2653  3              GET_PASSWORD($N_LIST);
; 2585        2654  2            end;
; 2586        2655  2
; 2587        2656  2  !
; 2588        2657  2  !           When is it being changed ???
; 2589        2658  2  !
; 2590        2659  2
; 2591        2660  2          $GETTIM(timadr=TIME_BUF);
; 2592        2661  2
; 2593        2662  2  !
; 2594        2663  2  !           If this is a new password ...
; 2595        2664  2  !
; 2596        2665  2
; 2597        2666  2          if .RECBUF[UAF$W_SALT] eql 0 then
; 2598        2667  2            RECBUF[UAF$W_SALT] = .TIME_BUF<3*8,16>;
; 2599        2668  2
; 2600        2669  2  !
; 2601        2670  2  !           Find the length of the username
; 2602        2671  2  !
; 2603        2672  2
; 2604        2673  3          begin
; 2605        2674  3
; 2606        2675  3            builtin LOCC;
; 2607        2676  3            builtin R0;
; 2608        2677  3            LOCC(%ref(%char(BLANK)), %ref(UAF$S_USERNAME), RECBUF[UAF$T_USERNAME]);
; 2609        2678  3            USER_DSC[DSC$W_LENGTH] = UAF$S_USERNAME - .R0;
; 2610        2679  3            USER_DSC[DSC$A_POINTER] = RECBUF[UAF$T_USERNAME];
; 2611        2680  3
; 2612        2681  2          end;
; 2613        2682  2
; 2614        2683  2  !
; 2615        2684  2  !           Check the new passwords for syntax
; 2616        2685  2  !
; 2617        2686  2
; 2618        2687  3          begin
; 2619        2688  3
; 2620      M 2689  3            macro CHECK_PASSWORD[n] =
; 2621      M 2690  3
; 2622      M 2691  3              if .$PASSWORD(n)[DSC$W_LENGTH] gtru 31 then
; 2623      M 2692  3                return LIB$SIGNAL(UAF$_PWDSYNTAX);
; 2624      M 2693  3              incr COUNTER to .$PASSWORD(n)[DSC$W_LENGTH]-1 do
; 2625      M 2694  3                if ch$fail( ch$find_ch(.SYMBOL_STR<0,8>, SYMBOL_STR+1,
; 2626      M 2695  3                            .vector[.$PASSWORD(n)[DSC$A_POINTER],
; 2627      M 2696  3                            .COUNTER;, byte]) )
; 2628      M 2697  3                then
; 2629        2698  3                  return LIB$SIGNAL(UAF$_PWDSYNTAX); %;
; 2630        2699  3
; 2631        2700  3            CHECK_PASSWORD($N_LIST);
; 2632        2701  3
; 2633        2702  2          end;
; 2634        2703  2
; 2635        2704  2  !
; 2636        2705  2  !           Modify the passwords
```

```
; 2637          2706 2 !
; 2638          2707 2
; 2639          2708 3     begin
; 2640          2709 3
; 2641        M 2710 3        macro MODIFY_PASSWORD[n] =
; 2642        M 2711 3
; 2643        M 2712 3           if not .$PRESENT(n) then
; 2644        M 2713 3              begin ! Blank out the current values
; 2645        M 2714 3
; 2646        M 2715 3                 ch$fill(0, %name('UAF$S_PWD',n), RECBUF[%name('UAF$Q_PWD',n)]);
; 2647        M 2716 3                 ch$fill(0, %name('UAF$S_PWD',n,'_DATE'),
; 2648        M 2717 3                            RECBUF[%name('UAF$Q_PWD',n,'_DATE')]);
; 2649        M 2718 3
; 2650        M 2719 3                 RECBUF[%name('UAF$V_PWD',n,'_EXPIRED')] = FALSE;
; 2651        M 2720 3                 PWD_FLAG = FALSE;
; 2652        M 2721 3
; 2653        M 2722 3              end
; 2654        M 2723 3           else if .$PASSWORD(n)[DSC$W_LENGTH] neq 0 then
; 2655        M 2724 3              begin ! Encrypt the new values
; 2656        M 2725 3
; 2657        M 2726 3                 RECBUF[%name('UAF$B_ENCRYPT',n)] = ENCRYPT;
; 2658        M 2727 3                 REC_ENCRYPT_DSC[DSC$W_LENGTH] = %name('UAF$S_PWD',n);
; 2659        M 2728 3                 REC_ENCRYPT_DSC[DSC$A_POINTER] = RECBUF[%name('UAF$Q_PWD',n)];
; 2660        M 2729 3                 PWD_FLAG = FALSE;
; 2661        M 2730 3
; 2662        M 2731 3                 LGI$HPWD(REC_ENCRYPT_DSC, $PASSWORD(n),
; 2663        M 2732 3                    .RECBUF[%name('UAF$B_ENCRYPT',n)], .RECBUF[UAF$W_SALT],
; 2664        M 2733 3                    user_dsc);
; 2665        M 2734 3
; 2666        M 2735 3                 ch$move( %name('UAF$S_PWD',n,'_DATE'), TIME_BUF,
; 2667        M 2736 3                          RECBUF[%name('UAF$Q_PWD',n,'_DATE')]);
; 2668        M 2737 3
; 2669          2738 3              end; %;
; 2670          2739 3
; 2671          2740 3        MODIFY_PASSWORD($N_LIST);
; 2672          2741 3
; 2673          2742 2     end;
; 2674          2743 2
; 2675          2744 2 !
; 2676          2745 2 !       That's all folks ...
; 2677          2746 2 !
; 2678          2747 2
; 2679          2748 2     return TRUE;
; 2680          2749 2
; 2681          2750 1 end;


                                                                   .PSECT  $OWN$,NOEXE,2

                                                   00124 USER_DSC:
                                                                   .BLKB   8
                                                   0012C REC_ENCRYPT_DSC:
                                                                   .BLKB   8

                                                                   .EXTRN  SYS$GETTIM
```

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

                                                         OFFC 00000 GETPASSWORD:
                                                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 2593
                                        5B 00000000G   00  9E 00002        MOVAB   PWD_FLAG, R11
                                        5A 00000000G   00  9E 00009        MOVAB   SYMBOL_STR, R10
                                        59 00000000G   00  9E 00010        MOVAB   TIME_BUF, R9
                                        58 00000000G   00  9E 00017        MOVAB   RECBUF+358, R8
                                        57 00000000'   00  9E 0001E        MOVAB   PASSWORD_, R7
                                        54             01  D0 00025        MOVL    #1, PRESENT_                                ; 2594
                                        56             01  D0 00028        MOVL    #1, PRESENT_2
                                           00000000'   00  9F 0002B        PUSHAB  SD_PASSWORD                                 ; 2642
                             00000000G  00             01  FB 00031        CALLS   #1, CLI$PRESENT
                             00000000G  8F             50  D1 00038        CMPL    R0, #CLI$_NEGATED
                                                       06  12 0003F        BNEQ    1$
                                                       54  D4 00041        CLRL    PRESENT_                                    ; 2645
                                                       56  D4 00043        CLRL    PRESENT_2
                                                       40  11 00045        BRB     2$                                          ; 2642
                                           00000000'   00  9F 00047 1$:    PUSHAB  SD_PASSWORD                                 ; 2647
                             00000000G  00             01  FB 0004D        CALLS   #1, CLI$PRESENT
                                        30             50  E9 00054        BLBC    R0, 2$
                                                       67  B4 00057        CLRW    PASSWORD_                                   ; 2653
                                                  04   A7  D4 00059        CLRL    PASSWORD_+4
                                                       57  DD 0005C        PUSHL   R7
                                           00000000'   00  9F 0005E        PUSHAB  SD_PASSWORD
                             00000000G  00             02  FB 00064        CALLS   #2, CLI$GET_VALUE
                                        54             50  D0 0006B        MOVL    R0, PRESENT_
                                                  08   A7  B4 0006E        CLRW    PASSWORD_2
                                                  0C   A7  D4 00071        CLRL    PASSWORD_2+4
                                                  08   A7  9F 00074        PUSHAB  PASSWORD_2
                                           00000000'   00  9F 00077        PUSHAB  SD_PASSWORD
                             00000000G  00             02  FB 0007D        CALLS   #2, CLI$GET_VALUE
                                        56             50  D0 00084        MOVL    R0, PRESENT_2
                                                       59  DD 00087 2$:    PUSHL   R9                                          ; 2660
                             00000000G  00             01  FB 00089        CALLS   #1, SYS$GETTIM
                                                       68  B5 00090        TSTW    RECBUF+358                                  ; 2666
                                                       04  12 00092        BNEQ    3$
                                        68        03   A9  B0 00094        MOVW    TIME_BUF+3, RECBUF+358                      ; 2667
                   FE9E   C8            20             20  3A 00098 3$:    LOCC    #32, #32, RECBUF+4                          ; 2677
                     18   A7            20             50  A3 0009E        SUBW3   R0, #32, USER_DSC                           ; 2678
                                    1C  A7      FE9E   C8  9E 000A3        MOVAB   RECBUF+4, USER_DSC+4                        ; 2679
                                        1F             67  B1 000A9        CMPW    PASSWORD_, #31                              ; 2700
                                                       45  1A 000AC        BGTRU   9$
                                        53             67  3C 000AE        MOVZWL  PASSWORD_, R3
                                        52             01  CE 000B1        MNEGL   #1, COUNTER
                                                       15  11 000B4        BRB     6$
                                        51             6A  9A 000B6 4$:    MOVZBL  SYMBOL_STR, R1
                                        50        04   A7  D0 000B9        MOVL    PASSWORD_+4, R0
                     01   AA            51           6240  3A 000BD        LOCC    (COUNTER)[R0], R1, SYMBOL_STR+1
                                                       02  12 000C3        BNEQ    5$
                                                       51  D4 000C5        CLRL    R1
                                                       51  D5 000C7 5$:    TSTL    R1
                                                       28  13 000C9        BEQL    9$
                          E7            52             53  F2 000CB 6$:    AOBLSS  R3, COUNTER, 4$
                                        1F        08   A7  B1 000CF        CMPW    PASSWORD_2, #31
                                                       1E  1A 000D3        BGTRU   9$
                                        53        08   A7  3C 000D5        MOVZWL  PASSWORD_2, R3
```

```
                                                                   B  1
UAFPARSE                                                          16-Sep-1984 02:21:19   VAX-11 Bliss-32 V4.0-742                Page  98
V04-000          GETPASSWORD - get user password                  14-Sep-1984 13:21:23   DISK$VMSMASTER:[UAF.SRC]UAFPARSE.B32;1  (27)
                                         52            01  CE 000D9           MNEGL   #1, COUNTER                                   ;
                                                       23  11 000DC           BRB     10$                                           ;
                                         51            6A  9A 000DE 7$:       MOVZBL  SYMBOL_STR, R1                                ;
                                         50       0C   A7  D0 000E1           MOVL    PASSWORD_2+4, R0                              ;
                     01   AA             51          6240  3A 000E5           LOCC    (COUNTER)[R0], R1, SYMBOL_STR+1               ;
                                                       02  12 000EB           BNEQ    8$                                            ;
                                                       51  D4 000ED           CLRL    R1                                            ;
                                                       51  D5 000EF 8$:       TSTL    R1                                            ;
                                                       0E  12 000F1           BNEQ    10$                                           ;
                                            00000000G  8F  DD 000F3 9$:       PUSHL   #UAF$_PWDSYNTAX                               ;
                             00000000G   00            01  FB 000F9           CALLS   #1, LIB$SIGNAL                                ;
                                                       04     00100           RET                                                   ;
                          D9             52            53  F2 00101 10$:      AOBLSS  R3, COUNTER, 7$                               ;
                                         16            54  E8 00105           BLBS    PRESENT_, 11$                                 ; 2740
           08             00             6E            00  2C 00108           MOVC5   #0, (SP), #0, #8, RECBUF+340                  ;
                                                  EE   A8     0010D                                                                 ;
           08             00             6E            00  2C 0010F           MOVC5   #0, (SP), #0, #8, RECBUF+380                  ;
                                                  16   A8     00114                                                                 ;
                                   6F    A8            02  8A 00116           BICB2   #2, RECBUF+469                                ;
                                                       6B  D4 0011A           CLRL    PWD_FLAG                                      ;
                                                       2F  11 0011C           BRB     12$                                           ;
                                                       67  B5 0011E 11$:      TSTW    PASSWORD_                                     ;
                                                       2B  13 00120           BEQL    12$                                           ;
                                   02    A8       00G  8F  90 00122           MOVB    #ENCRYPT, RECBUF+360                          ;
                                   20    A7            08  B0 00127           MOVW    #8, REC_ENCRYPT_DSC                           ;
                                   24    A7       EE   A8  9E 0012B           MOVAB   RECBUF+340, REC_ENCRYPT_DSC+4                 ;
                                                       6B  D4 00130           CLRL    PWD_FLAG                                      ;
                                                  18   A7  9F 00132           PUSHAB  USER_DSC                                      ;
                                         7E            68  3C 00135           MOVZWL  RECBUF+358, -(SP)                             ;
                                         7E       02   A8  9A 00138           MOVZBL  RECBUF+360, -(SP)                             ;
                                                       57  DD 0013C           PUSHL   R7                                            ;
                                                  20   A7  9F 0013E           PUSHAB  REC_ENCRYPT_DSC                               ;
                             00000000G   00            05  FB 00141           CALLS   #5, LGI$HPWD                                  ;
                     16   A8             69            08  28 00148           MOVC3   #8, TIME_BUF, RECBUF+380                      ;
                                         16            56  E8 0014D 12$:      BLBS    PRESENT_2, 13$                                ;
           08             00             6E            00  2C 00150           MOVC5   #0, (SP), #0, #8, RECBUF+348                  ;
                                                  F6   A8     00155                                                                 ;
           08             00             6E            00  2C 00157           MOVC5   #0, (SP), #0, #8, RECBUF+388                  ;
                                                  1E   A8     0015C                                                                 ;
                                   6F    A8            04  8A 0015E           BICB2   #4, RECBUF+469                                ;
                                                       6B  D4 00162           CLRL    PWD_FLAG                                      ;
                                                       31  11 00164           BRB     14$                                           ;
                                                  08   A7  B5 00166 13$:      TSTW    PASSWORD_2                                    ;
                                                       2C  13 00169           BEQL    14$                                           ;
                                   03    A8       00G  8F  90 0016B           MOVB    #ENCRYPT, RECBUF+361                          ;
                                   20    A7            08  B0 00170           MOVW    #8, REC_ENCRYPT_DSC                           ;
                                   24    A7       F6   A8  9E 00174           MOVAB   RECBUF+348, REC_ENCRYPT_DSC+4                 ;
                                                       6B  D4 00179           CLRL    PWD_FLAG                                      ;
                                                  18   A7  9F 0017B           PUSHAB  USER_DSC                                      ;
                                         7E            68  3C 0017E           MOVZWL  RECBUF+358, -(SP)                             ;
                                         7E       03   A8  9A 00181           MOVZBL  RECBUF+361, -(SP)                             ;
                                                  08   A7  9F 00185           PUSHAB  PASSWORD_2                                    ;
                                                  20   A7  9F 00188           PUSHAB  REC_ENCRYPT_DSC                               ;
                             00000000G   00            05  FB 0018B           CALLS   #5, LGI$HPWD                                  ;
                     1E   A8             69            08  28 00192           MOVC3   #8, TIME_BUF, RECBUF+388                      ;
                                         50            01  D0 00197 14$:      MOVL    #1, R0                                        ; 2748
                                                       04     0019A           RET                                                   ; 2750
```

```
; Routine Size:  411 bytes,    Routine Base:  $CODE$ + 1254
```

```
_$2



Sy
---
LI
PR
SA
SY
SY
SY
SY
SY
SY
SY
SY
SY
SY
```

```
; 2683      2751  1 %sbttl 'AUTHORIZE_GENERATE -- Interface to PL/I GENERATE_PASSWORDS routine'
; 2684      2752  1 routine AUTHORIZE_GENERATE( NEW_PASSWORD: ref block[DSC$K_D_BLN, byte],
; 2685      2753  1                              THIS, TLENGTH ) =
; 2686      2754  2 begin
; 2687      2755  2
; 2688      2756  2 !++
; 2689      2757  2 !
; 2690      2758  2 ! FUNCTIONAL DESCRIPTION:
; 2691      2759  2 !
; 2692      2760  2 !       Interface to PL/I procedure GENERATE_PASSWORD, which
; 2693      2761  2 !       generates random, pronounceable passwords. This routine
; 2694      2762  2 !       must prompt the user for input with NOECHO in effect
; 2695      2763  2 !
; 2696      2764  2 ! INPUTS:
; 2697      2765  2 !
; 2698      2766  2 !       None
; 2699      2767  2 !
; 2700      2768  2 ! IMPLICIT INPUTS:
; 2701      2769  2 !
; 2702      2770  2 !       Input from user terminal.
; 2703      2771  2 !
; 2704      2772  2 ! OUTPUTS:
; 2705      2773  2 !
; 2706      2774  2 !       PASSWORD, by descriptor.
; 2707      2775  2 !
; 2708      2776  2 ! IMPLICIT OUTPUTS:
; 2709      2777  2 !
; 2710      2778  2 !       User prompt.
; 2711      2779  2 !
; 2712      2780  2 ! ROUTINE VALUE:
; 2713      2781  2 !
; 2714      2782  2 !       None
; 2715      2783  2 !
; 2716      2784  2 ! NOTE:
; 2717      2785  2 !
; 2718      2786  2 !       PL/I apparently uses a form of ASCIC to represent its strings,
; 2719      2787  2 !       however, PL/I uses a word for the character count, not a byte.
; 2720      2788  2 !
; 2721      2789  2 !--
; 2722      2790  2
; 2723      2791  2    macro
; 2724      2792  2
; 2725      2793  2       DSC$L_L1 = 24, 0, 32, 0 %,
; 2726      2794  2       DSC$L_U1 = 28, 0, 32, 0 %;
; 2727      2795  2
; 2728      2796  2    literal
; 2729      2797  2
; 2730      2798  2       MAX = 10,
; 2731      2799  2       MIN = 6,
; 2732      2800  2       N_WORDS = 5;
; 2733      2801  2
; 2734      2802  2    own
; 2735      2803  2
; 2736      2804  2       CHANNEL,
; 2737      2805  2       DSC: block[DSC$K_S_BLN, byte] preset([DSC$B_CLASS]=DSC$K_CLASS_S),
; 2738      2806  2       FUNCTION: initial(IO$_READPROMPT or IO$M_CVTLOW or IO$M_NOECHO),
; 2739      2807  2       PLENGTH,
```

```
   2            PROMPT: vector[45, byte] preset([0]=13, [1]=10,
   2               [2]='E', [3]='n', [4]='t', [5]='e', [6]='r', [7]=' ');
   2
   2         bind
   2
   2            P2 = UPLIT(' password: ');
   2 !
   2 !            The password generator requires a complex descriptor
   2 !
   2
 M 2         macro PLI$VARYING_CHARACTER_ARRAY[NAME, LENGTH, WIDTH] =
 M 2
 M 2            own
 M 2
 M 2               %name(NAME): blockvector[LENGTH, (WIDTH+2), byte],
 M 2               %name(NAME,'_DSC'): block[32, byte]
 M 2                  preset([DSC$W_MAXSTRLEN] = WIDTH,
 M 2                         [DSC$B_DTYPE] = 37,
 M 2                         [DSC$B_CLASS] = DSC$K_CLASS_VSA,
 M 2                         [DSC$A_POINTER] = %name(NAME),
 M 2                         [DSC$B_DIMCT] = 1,
 M 2                         [DSC$A_A0] = %name(NAME) - (WIDTH+2),
 M 2                         [DSC$L_S1] = (WIDTH+2),
 M 2                         [DSC$L_L1] = 1,
   2                         [DSC$L_U1] = LENGTH); %;
   2
   2         PLI$VARYING_CHARACTER_ARRAY('PWD', N_WORDS, MAX, 'HYPH', N_WORDS, 2*MAX);
   2
   2 !
   2 !            Build the prompt string
   2 !
   2
   2         ch$move(.TLENGTH, .THIS, PROMPT[8]);
   2         ch$move(11, P2, PROMPT[8+.TLENGTH]);
   2
   2         PLENGTH = 8 + .TLENGTH + 11;
   2
   2 !
   2 !            Open a channel to the user
   2 !
   2
   2         SYS$ASSIGN(%ascid'SYS$COMMAND:', CHANNEL, 0, 0);
   2         NEW_PASSWORD[DSC$W_LENGTH] = 0;
   2
   2 !
   2 !            Generate random passwords ...
   2 !
   2
   2         while .NEW_PASSWORD[DSC$W_LENGTH] eql 0 do
   3            begin
   3
   3               own
   3                  BUFFER: vector[128, byte],
   3                  IOSB: vector[4, word],
   3                  LISTED;
```

```
; 2797          2865  3             LISTED = FALSE;
; 2798          2866  3
; 2799          2867  3             SYS$QIOW(0, .CHANNEL, IO$_WRITEVBLK, 0,0,0, 0, 0, 0, 32, 0,0);
; 2800          2868  3
; 2801          2869  3             GENERATE_PASSWORDS(PWD_DSC, HYPH_DSC,
; 2802          2870  3                 uplit(MIN), uplit(MAX), uplit(N_WORDS));
; 2803          2871  3
; 2804          2872  3             SYS$QIOW(0, .CHANNEL, IO$_WRITEVBLK, 0,0,0, 0, 0, 0, 32, 0,0);
; 2805          2873  3
; 2806          2874  3             incr i to N_WORDS-1 do
; 2807          2875  4                 begin
; 2808          2876  4                     DSC[DSC$W_LENGTH] = .PWD[.i, 0, 0, 16, 0];
; 2809          2877  4                     DSC[DSC$A_POINTER] = PWD[.i, 2, 0, 0, 0];
; 2810          2878  4                     STR$UPCASE(DSC, DSC);
; 2811          2879  3                 end;
; 2812          2880  3
; 2813          2881  3             while not .LISTED do
; 2814          2882  4                 begin
; 2815          2883  4
; 2816          2884  4                     SYS$QIOW(0, .CHANNEL, .FUNCTION, IOSB, 0, 0,
; 2817          2885  4                         BUFFER, 127, 0, 0, PROMPT, .PLENGTH);
; 2818          2886  4
; 2819          2887  4                     if .BUFFER[.IOSB[1]] eql 26 then ! CTRL/Z
; 2820          2888  4                         return LIB$SIGNAL(UAF$_PWDNCH, 2, .TLENGTH, .THIS)
; 2821          2889  4                     else if .IOSB[1] eql 0 then
; 2822          2890  4                         exitloop;
; 2823          2891  4
; 2824          2892  4                     incr i to N_WORDS-1 do
; 2825          2893  4                         LISTED = .LISTED or
; 2826          2894  4                             ch$eql(.IOSB[1], BUFFER,
; 2827          2895  4                                 .PWD[.i, 0, 0, 16, 0], PWD[.i, 2, 0, 0, 0]);
; 2828          2896  4
; 2829          2897  4                     if .LISTED then
; 2830          2898  5                         begin
; 2831          2899  5                             NEW_PASSWORD[DSC$W_LENGTH] = .IOSB[1];
; 2832          2900  5                             LIB$GET_VM(%ref(.IOSB[1]), NEW_PASSWORD[DSC$A_POINTER]);
; 2833          2901  5                             ch$move(.IOSB[1], BUFFER, .NEW_PASSWORD[DSC$A_POINTER]);
; 2834          2902  5                         end
; 2835          2903  4                     else
; 2836          2904  4                         LIB$SIGNAL(UAF$_PWDNOL);
; 2837          2905  4
; 2838          2906  3                 end;
; 2839          2907  3
; 2840          2908  2         end;
; 2841          2909  2
; 2842          2910  2 !
; 2843          2911  2 !       That's all folks ...
; 2844          2912  2 !
; 2845          2913  2
; 2846          2914  2     return SYS$DASSGN(.CHANNEL);
; 2847          2915  2
; 2848          2916  1 end;


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

```
  00  20  3A  64  72  6F  77  73  73  61  70  20  00534 P.AFS:   .ASCII  \ password: \<0>                                     ;
  3A  44  4E  41  4D  4D  4F  43  24  53  59  53  00540 P.AFU:   .ASCII  \SYS$COMMAND:\                                       ;
                                        010E000C  0054C P.AFT:   .LONG   17694732                                             ;
                                        00000000' 00550          .ADDRESS P.AFU                                               ;
                                        00000006  00554 P.AFV:   .LONG   6                                                    ;
                                        0000000A  00558 P.AFW:   .LONG   10                                                   ;
                                        00000005  0055C P.AFX:   .LONG   5                                                    ;

                                                                 .PSECT  $OWN$,NOEXE,2

                                                  00134 CHANNEL:.BLKB    4
                                              00# 00138 DSC:     .BYTE   0[3]                                                 ;
                                              01  0013B          .BYTE   1                                                    ;
                                                  0013C          .BLKB   4
                                        00000177  00140 FUNCTION:
                                                                 .LONG   375                                                  ;
                                                  00144 PLENGTH:.BLKB    4
                  20  72  65  74  6E  45  0A  0D  00148 PROMPT:  .BYTE   13, 10, 69, 110, 116, 101, 114, 32                   ;
                                                  00150          .BLKB   37
                                                  00175          .BLKB   3
                                                  00178 PWD:     .BLKB   60
                                            000A  001B4 PWD_DSC:.WORD    10                                                   ;
                                           0C  25 001B6          .BYTE   37, 12                                               ;
                                        00000000' 001B8          .ADDRESS PWD                                                 ;
                                              00# 001BC          .BYTE   0[3]                                                 ;
                                              01  001BF          .BYTE   1                                                    ;
                                              00# 001C0          .BYTE   0[4]                                                 ;
                                        00000000' 001C4          .ADDRESS PWD-12                                              ;
                    00000005  00000001  0000000C  001C8          .LONG   12, 1, 5                                             ;
                                                  001D4 HYPH:    .BLKB   110
                                                  00242          .BLKB   2
                                            0014  00244 HYPH_DSC:
                                                                 .WORD   20                                                   ;
                                           0C  25 00246          .BYTE   37, 12                                               ;
                                        00000000' 00248          .ADDRESS HYPH                                                ;
                                              00# 0024C          .BYTE   0[3]                                                 ;
                                              01  0024F          .BYTE   1                                                    ;
                                              00# 00250          .BYTE   0[4]                                                 ;
                                        00000000' 00254          .ADDRESS HYPH-22                                             ;
                    00000005  00000001  00000016  00258          .LONG   22, 1, 5                                             ;
                                                  00264 BUFFER: .BLKB    128
                                                  002E4 IOSB:    .BLKB   8
                                                  002EC LISTED: .BLKB    4

                                                        P2=                      P.AFS


                                                                 .PSECT  $CODE$,NOWRT,2

                                            03FC  00000 AUTHORIZE_GENERATE:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                         ; 2752
                         59 00000000G  00  9E 00002          MOVAB   LIB$SIGNAL, R9                                           ;
                         58 00000000G  00  9E 00009          MOVAB   SYS$QIOW, R8                                             ;
                         57 00000000'  00  9E 00010          MOVAB   P2, R7                                                   ;
                         56 00000000'  00  9E 00017          MOVAB   CHANNEL, R6                                              ;
                         5E            04  C2 0001E          SUBL2   #4, SP                                                   ;
      1C    A6      08   BC       0C   AC  28 00021          MOVC3   TLENGTH, @THIS, PROMPT+8                                 ; 2841
```

```
                           50      1C   A6   9E 00028        MOVAB   PROMPT+8, R0                        ; 2842
  0C BC40                  67           0B   28 0002C        MOVC3   #11, P2, @TLENGTH[R0]               ;
  10   A6         0C       AC           13   C1 00032        ADDL3   #19, TLENGTH, PLENGTH               ; 2844
                                        7E   7C 00038        CLRQ    -(SP)                               ; 2850
                                        56   DD 0003A        PUSHL   R6                                  ;
                                   18   A7   9F 0003C        PUSHAB  P.AFT                               ;
             00000000G     00           04   FB 0003F        CALLS   #4, SYS$ASSIGN                      ;
                                   04   BC   B4 00046        CLRW    @NEW_PASSWORD                       ; 2851
                                   04   BC   B5 00049 1$:    TSTW    @NEW_PASSWORD                       ; 2857
                                        03   13 0004C        BEQL    2$                                  ;
                                      0111   31 0004E        BRW     11$                                 ;
                                 01B8   C6   D4 00051 2$:    CLRL    LISTED                              ; 2865
                                        7E   7C 00055        CLRQ    -(SP)                               ; 2867
                                        20   DD 00057        PUSHL   #32                                 ;
                                        7E   7C 00059        CLRQ    -(SP)                               ;
                                        7E   7C 0005B        CLRQ    -(SP)                               ;
                                        7E   7C 0005D        CLRQ    -(SP)                               ;
                                        30   DD 0005F        PUSHL   #48                                 ;
                                        66   DD 00061        PUSHL   CHANNEL                             ;
                                        7E   D4 00063        CLRL    -(SP)                               ;
                           68           0C   FB 00065        CALLS   #12, SYS$QIOW                       ;
                                   28   A7   9F 00068        PUSHAB  P.AFX                               ; 2870
                                   24   A7   9F 0006B        PUSHAB  P.AFW                               ;
                                   20   A7   9F 0006E        PUSHAB  P.AFV                               ;
                                 0110   C6   9F 00071        PUSHAB  HYPH_DSC                            ; 2869
                                 0080   C6   9F 00075        PUSHAB  PWD_DSC                             ;
             00000000G     00           05   FB 00079        CALLS   #5, GENERATE_PASSWORDS              ;
                                        7E   7C 00080        CLRQ    -(SP)                               ; 2872
                                        20   DD 00082        PUSHL   #32                                 ;
                                        7E   7C 00084        CLRQ    -(SP)                               ;
                                        7E   7C 00086        CLRQ    -(SP)                               ;
                                        7E   7C 00088        CLRQ    -(SP)                               ;
                                        30   DD 0008A        PUSHL   #48                                 ;
                                        66   DD 0008C        PUSHL   CHANNEL                             ;
                                        7E   D4 0008E        CLRL    -(SP)                               ;
                           68           0C   FB 00090        CALLS   #12, SYS$QIOW                       ;
                                        52   D4 00093        CLRL    I                                   ; 2874
           50              52           0C   C5 00095 3$:    MULL3   #12, I, R0                          ; 2876
                              44 A640   9F 00099             PUSHAB  PWD[R0]                             ;
                  04       A6           9E   B0 0009D        MOVW    @(SP)+, DSC                         ;
                  08       A6 46 A640   9E 000A1             MOVAB   PWD+2[R0], DSC+4                    ; 2877
                                   04   A6   9F 000A7        PUSHAB  DSC                                 ; 2878
                                   04   A6   9F 000AA        PUSHAB  DSC                                 ;
             00000000G     00           02   FB 000AD        CALLS   #2, STR$UPCASE                      ;
           DD              52           04   F3 000B4        AOBLEQ  #4, I, 3$                           ; 2874
                           8C    01B8   C6   E8 000B8 4$:    BLBS    LISTED, 1$                          ; 2881
                                   10   A6   DD 000BD        PUSHL   PLENGTH                             ; 2885
                                   14   A6   9F 000C0        PUSHAB  PROMPT                              ; 2884
                                        7E   7C 000C3        CLRQ    -(SP)                               ;
                           7E      7F   8F   9A 000C5        MOVZBL  #127, -(SP)                         ;
                                 0130   C6   9F 000C9        PUSHAB  BUFFER                              ;
                                        7E   7C 000CD        CLRQ    -(SP)                               ;
                                 01B0   C6   9F 000CF        PUSHAB  IOSB                                ;
                                   0C   A6   DD 000D3        PUSHL   FUNCTION                            ;
                                        66   DD 000D6        PUSHL   CHANNEL                             ;
                                        7E   D4 000D8        CLRL    -(SP)                               ;
                           68           0C   FB 000DA        CALLS   #12, SYS$QIOW                       ;
```

```
                                         50      01B2  C6  3C 000DD         MOVZWL  IOSB+2, R0                                        ; 2887
                                         1A      0130C640  91 000E2         CMPB    BUFFER[R0], #26                                   ;
                                                       12  12 000E8         BNEQ    5$                                                ;
                                                   08  AC  DD 000EA         PUSHL   THIS                                              ; 2888
                                                   0C  AC  DD 000ED         PUSHL   TLENGTH                                           ;
                                                       02  DD 000F0         PUSHL   #2                                                ;
                                            00000000G  8F  DD 000F2         PUSHL   #UAF$_PWDNCH                                      ;
                                         69            04  FB 000F8         CALLS   #4, LIB$SIGNAL                                    ;
                                                           04 000FB         RET                                                       ;
                                                       50  D5 000FC 5$:     TSTL    R0                                                ; 2889
                                                       03  12 000FE         BNEQ    6$                                                ;
                                                     FF46  31 00100         BRW     1$                                                ;
                                                       54  D4 00103 6$:     CLRL    I                                                 ; 2892
                          50             54            0C  C5 00105 7$:     MULL3   #12, I, R0                                        ; 2895
                                                       55  D4 00109         CLRL    R5                                                ;
                                                  44 A640  9F 0010B         PUSHAB  PWD[R0]                                           ;
            9E            00     0130    C6      01B2  C6  2D 0010F         CMPC5   IOSB+2, BUFFER, #0, @(SP)+, PWD+2[R0]             ;
                                                  46 A640     00118                                                                   ;
                                                       02  12 0011B         BNEQ    8$                                                ;
                                                       55  D6 0011D         INCL    R5                                                ;
                                 01B8    C6            55  C8 0011F 8$:     BISL2   R5, LISTED                                        ;
                          DD             54            04  F3 00124         AOBLEQ  #4, I, 7$                                         ; 2893
                                         29      01B8  C6  E9 00128         BLBC    LISTED, 9$                                        ; 2897
                                         50      01B2  C6  3C 0012D         MOVZWL  IOSB+2, R0                                        ; 2899
                                   04    BC            50  B0 00132         MOVW    R0, @NEW_PASSWORD                                 ;
                                         52        04  AC  D0 00136         MOVL    NEW_PASSWORD, R2                                  ; 2900
                                                   04  A2  9F 0013A         PUSHAB  4(R2)                                             ;
                                   04    AE            50  D0 0013D         MOVL    R0, 4(SP)                                         ;
                                                   04  AE  9F 00141         PUSHAB  4(SP)                                             ;
                            00000000G    00            02  FB 00144         CALLS   #2, LIB$GET_VM                                    ;
                     04   B2     0130    C6      01B2  C6  28 0014B         MOVC3   IOSB+2, BUFFER, @4(R2)                            ; 2901
                                                       09  11 00154         BRB     10$                                               ; 2897
                                            00000000G  8F  DD 00156 9$:     PUSHL   #UAF$_PWDNOL                                      ; 2904
                                         69            01  FB 0015C         CALLS   #1, LIB$SIGNAL                                    ;
                                                     FF56  31 0015F 10$:    BRW     4$                                                ; 2881
                                                       66  DD 00162 11$:    PUSHL   CHANNEL                                           ; 2914
                            00000000G    00            01  FB 00164         CALLS   #1, SYS$DASSGN                                    ;
                                                           04 0016B         RET                                                       ; 2916

; Routine Size:  364 bytes,    Routine Base:  $CODE$ + 13EF


; 2849          2917  1
; 2850          2918  1 end
; 2851          2919  0 eludom



                                                                            .EXTRN  LIB$SIGNAL

;                                      PSECT SUMMARY
;
;      Name                        Bytes                         Attributes
;
;  $PLIT$                                 1376  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                                   752  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;  _LIB$KEY0$                        4  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                     158  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$KEY1$                       19  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  $CODE$                         5467  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]TPAMAC.L32;1                 42        30        71           14        00:00.1
;  _$255$DUA28:[SYSLIB]LIB.L32;1                 18619       125         0         1000        00:02.2





;                                      COMMAND QUALIFIERS

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

; Size:          5467 code + 2309 data bytes
; Run Time:         01:43.8
; Elapsed Time:     01:50.7
; Lines/CPU Min:     1686
; Lexemes/CPU-Min: 26833
; Memory Used:  685 pages
; Compilation Complete
```

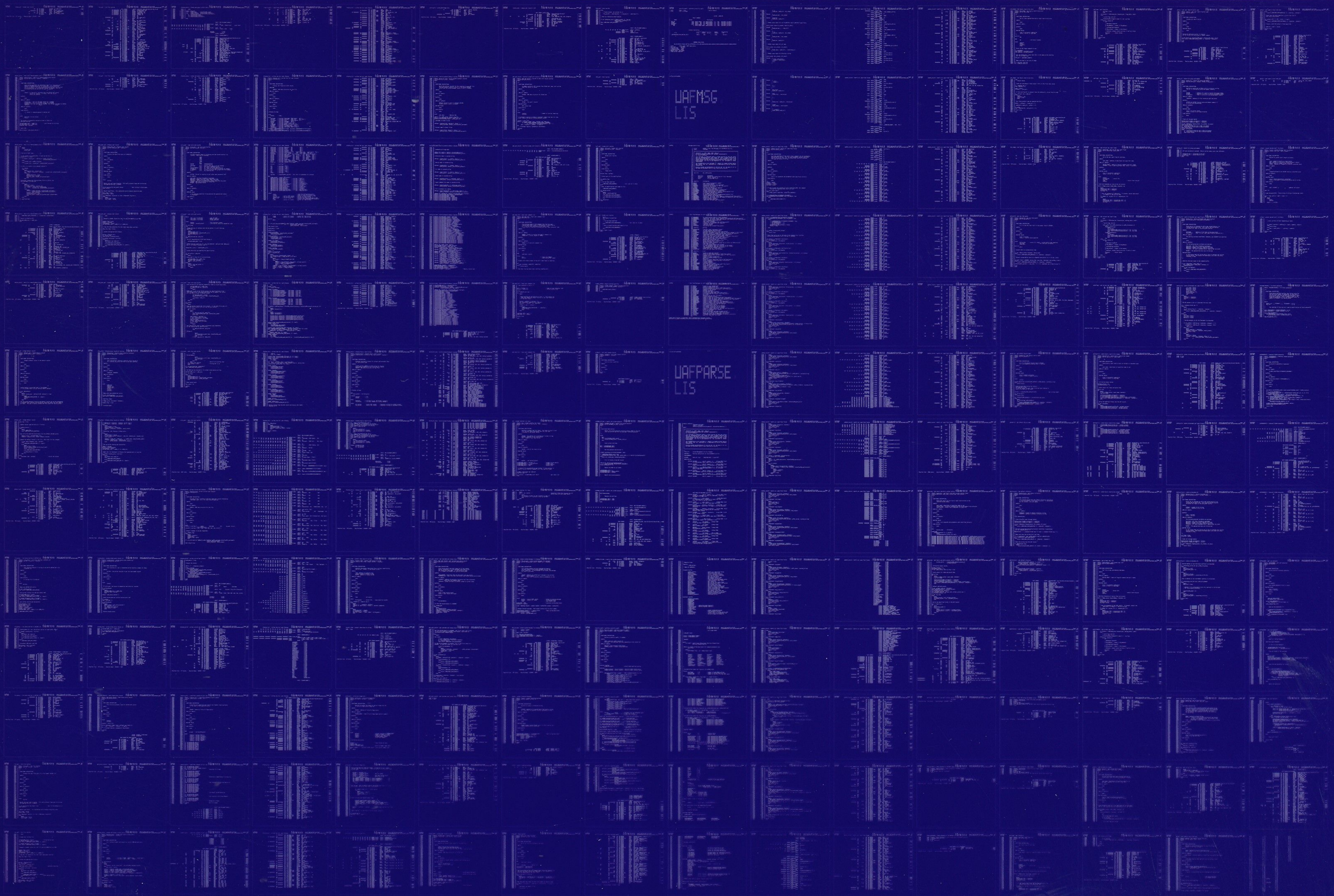
```
_$



Sy
--
LI
PR
SA
SY
SY
SY
SY
SY
SY
SY
SY
SY
```

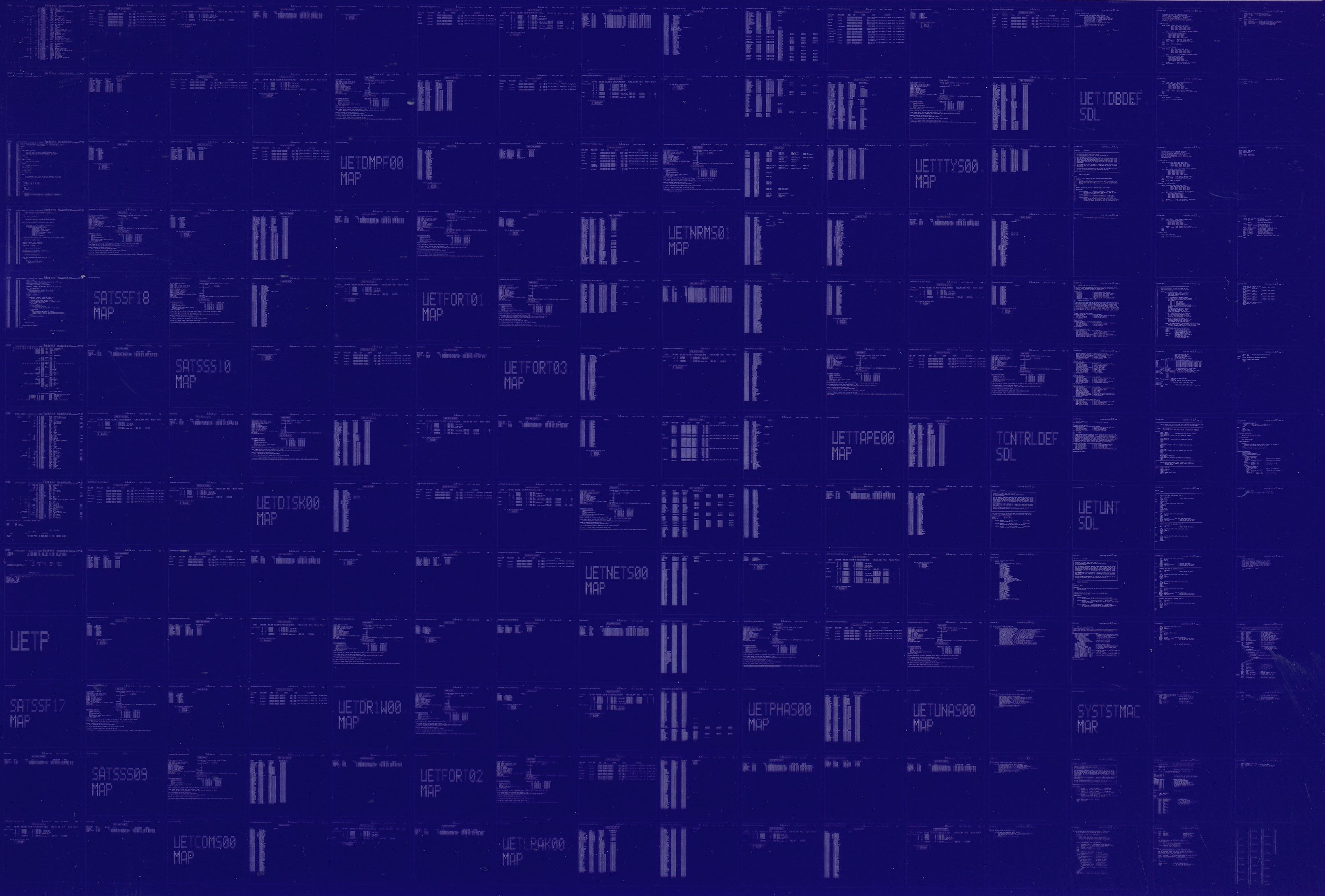
0407 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0408 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



UETIDBDEF
SDL

UETDMPF00
MAP

UETTTYS00
MAP

UETNRMS01
MAP

SATSSF18
MAP

UETFORT01
MAP

SATSSS10
MAP

UETFORT03
MAP

UETTAPE00
MAP

TCNTRLDEF
SDL

UETDISK00
MAP

UETUNT
SDL

UETNETS00
MAP

UETP

SATSSF17
MAP

UETDR1W00
MAP

UETPHAS00
MAP

UETUNAS00
MAP

SYSTSTMAC
MAR

SATSSS09
MAP

UETFORT02
MAP

UETCOMS00
MAP

UETLPAK00
MAP