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

```
VV      VV    AAAAAA    LL            IIIIII    DDDDDDDD      AAAAAA    TTTTTTTTTT  EEEEEEEEEE
VV      VV    AAAAAA    LL            IIIIII    DDDDDDDD      AAAAAA    TTTTTTTTTT  EEEEEEEEEE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EEEEEEEE
VV      VV   AA    AA   LL              II      DD      DD   AA    AA      TT      EEEEEEEE
VV      VV   AAAAAAAAAA LL              II      DD      DD   AAAAAAAAAA      TT      EE
VV      VV   AAAAAAAAAA LL              II      DD      DD   AAAAAAAAAA      TT      EE
  VV  VV     AA    AA   LL              II      DD      DD   AA    AA      TT      EE              ....
  VV  VV     AA    AA   LL              II      DD      DD   AA    AA      TT      EE              ....
    VV       AA    AA   LLLLLLLLLL    IIIIII    DDDDDDDD    AA    AA      TT      EEEEEEEEEE      ....
    VV       AA    AA   LLLLLLLLLL    IIIIII    DDDDDDDD    AA    AA      TT      EEEEEEEEEE      ....

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

```
;  1          0001  0 Module VALIDATE_USER
;  2          0002  0    (%title 'Validate username and password against UAF'
;  3          0003  0     ident = 'V04-000',
;  4          0004  0     language(Bliss32),
;  5          0005  0     addressing_mode(external=GENERAL)) =
;  6          0006  1 begin
;  7          0007  1
;  8          0008  1 !**************************************************************************
;  9          0009  1 !*                                                                        *
; 10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
; 11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
; 12          0012  1 !*  ALL RIGHTS RESERVED.                                                   *
; 13          0013  1 !*                                                                        *
; 14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
; 16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 19          0019  1 !*  TRANSFERRED.                                                           *
; 20          0020  1 !*                                                                        *
; 21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 23          0023  1 !*  CORPORATION.                                                           *
; 24          0024  1 !*                                                                        *
; 25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
; 27          0027  1 !*                                                                        *
; 28          0028  1 !*                                                                        *
; 29          0029  1 !**************************************************************************
; 30          0030  1
; 31          0031  1 !++
; 32          0032  1 ! FACILITY: System Subprogram
; 33          0033  1 !
; 34          0034  1 ! ABSTRACT:
; 35          0035  1 !
; 36          0036  1 !       Lookup the supplied username in the authorization file.  Validate
; 37          0037  1 !       the password, and return the record if so, else return the
; 38          0038  1 !       default values.
; 39          0039  1 !
; 40          0040  1 ! ENVIRONMENT:
; 41          0041  1 !
; 42          0042  1 ! AUTHOR: Henry M. Levy , CREATION DATE: 10-Sep-1977
; 43          0043  1 !
; 44          0044  1 ! MODIFIED BY:
; 45          0045  1 !
; 46          0046  1 !       V03-008 JRL0030         John R. Lawson, Jr.     20-Jul-1984 14:21
; 47          0047  1 !               Repaired sloppiness in LGI$SEARCHUSER
; 48          0048  1 !
; 49          0049  1 !       V03-007 MHB0157         Mark Bramhall            7-May-1984
; 50          0050  1 !               Guard against no global buffers when opening SYSUAF.
; 51          0051  1 !
; 52          0052  1 !       V03-006 PCG0001         Peter George            31-Jan-1984  19:28
; 53          0053  1 !               Add password number argument to lgi$check_pass.
; 54          0054  1 !               Add access type argument to lgi$searchuser.
; 55          0055  1 !
; 56          0056  1 !       V03-005 ACG0385         Andrew C. Goldstein,    27-Dec-1983  11:54
; 57          0057  1 !               Rewrite in BLISS; add support for PURDY_V
```

```
;    58          0058  1 !                 encryption to support usernames longer than 12 chars.
;    59          0059  1 !
;    60          0060  1 !       V03-004 GAS0165         Gerry Smith              3-Aug-1983
;    61          0061  1 !               Add $PSLDEF, to get psl$c_exec.
;    62          0062  1 !
;    63          0063  1 !       V03-003 GAS0164         Gerry Smith             30-Jul-1983
;    64          0064  1 !               Change the FAB[FAB$B_DSBMSK] kludge to the new
;    65          0065  1 !               logical name disable method, fab$v_lnm_mode.
;    66          0066  1 !
;    67          0067  1 !       V03-002 TMH0002         Tim Halvorsen            07-Feb-1983
;    68          0068  1 !               Make password validation return success if the user
;    69          0069  1 !               password is null.  This allows specific access control
;    70          0070  1 !               strings to be matched with accounts with null passwords.
;    71          0071  1 !
;    72          0072  1 !       V03-001 GAS0057         Gerry Smith              22-Mar-1982
;    73          0073  1 !               Fix FAB to disable all but system logical name
;    74          0074  1 !               translation during open of SYSUAF.
;    75          0075  1 !
;    76          0076  1 !       V02-002 HRJ0031         Herb Jacobs              03-Nov-1981
;    77          0077  1 !               Add a validate password entry point.
;    78          0078  1 !
;    79          0079  1 !       V001    TMH0001         Tim Halvorsen   09-Mar-1981
;    80          0080  1 !               Enable all authorized privileges before accessing SYSUAF
;    81          0081  1 !               file and restore current privileges on exit in case
;    82          0082  1 !               the caller doesn't always run with all authorized
;    83          0083  1 !               privileges enabled (i.e. LOGIN).
;    84          0084  1 !--
;    85          0085  1
;    86          0086  1
;    87          0087  1 FORWARD ROUTINE
;    88          0088  1         LGI$VALIDATE,                   ! validate username and password against UAF
;    89          0089  1         LGI$SEARCHUSER,                 ! read user's UAF record
;    90          0090  1         LGI$CHECK_PASS;                 ! check password against UAF record
;    91          0091  1
;    92          0092  1 LIBRARY
;    93          0093  1         'SYS$LIBRARY:LIB';
;    94          0094  1
;    95          0095  1
;    96          0096  1 PSECT   CODE    = _LIB$CODE (EXECUTE, PIC, SHARE, ALIGN (0)),
;    97          0097  1         PLIT    = _LIB$CODE (EXECUTE, PIC, SHARE, ALIGN (0));
```

```
;   99          0098  1 GLOBAL ROUTINE LGI$VALIDATE (P_USERNAME, P_PASSWORD, P_UAFRECORD) =
;  100          0099  1
;  101          0100  1 !++
;  102          0101  1 !
;  103          0102  1 ! FUNCTIONAL DESCRIPTION:
;  104          0103  1 !
;  105          0104  1 !       This routine reads the UAF record for the specified user and
;  106          0105  1 !       checks the specified password against it.
;  107          0106  1 !
;  108          0107  1 ! CALLING SEQUENCE:
;  109          0108  1 !       LGI$VALIDATE (P_USERNAME, P_PASSWORD, P_UAFRECORD)
;  110          0109  1 !
;  111          0110  1 ! INPUT PARAMETERS:
;  112          0111  1 !       P_USERNAME: address of string descriptor of username
;  113          0112  1 !       P_PASSWORD: address of string descriptor of password
;  114          0113  1 !
;  115          0114  1 ! IMPLICIT INPUTS:
;  116          0115  1 !       NONE
;  117          0116  1 !
;  118          0117  1 ! OUTPUT PARAMETERS:
;  119          0118  1 !       P_UAFRECORD: address of descriptor of buffer into which to read UAF record
;  120          0119  1 !
;  121          0120  1 ! IMPLICIT OUTPUTS:
;  122          0121  1 !       NONE
;  123          0122  1 !
;  124          0123  1 ! ROUTINE VALUE:
;  125          0124  1 !       1 if successful
;  126          0125  1 !       -2 if invalid username
;  127          0126  1 !       -4 if invalid password
;  128          0127  1 !       RMS$_xxx if errors reading the UAF
;  129          0128  1 !
;  130          0129  1 ! SIDE EFFECTS:
;  131          0130  1 !       NONE
;  132          0131  1 !
;  133          0132  1 !--
;  134          0133  1
;  135          0134  2 BEGIN
;  136          0135  2
;  137          0136  2 MAP
;  138          0137  2         P_UAFRECORD     : REF $BBLOCK;
;  139          0138  2
;  140          0139  2 LOCAL
;  141          0140  2         STATUS;                         ! routine return value
;  142          0141  2
;  143          0142  2
;  144          0143  2 ! Use common subroutines to read the UAF record, and if that succeeds,
;  145          0144  2 ! to check the password.
;  146          0145  2 !
;  147          0146  2
;  148          0147  2 STATUS = LGI$SEARCHUSER (.P_USERNAME, .P_PASSWORD, .P_UAFRECORD);
;  149          0148  2 IF .STATUS
;  150          0149  2 THEN LGI$CHECK_PASS (.P_PASSWORD, .P_UAFRECORD[DSC$A_POINTER])
;  151          0150  2 ELSE .STATUS
;  152          0151  2
;  153          0152  1 END;                                    ! End of routine LGI$VALIDATE
```

```
                                                        .TITLE  VALIDATE_USER Validate username and password ag
;                                                               ainst UAF
                                                        .IDENT  \V04-000\


                                                        .PSECT  _LIB$CODE,NOWRT,  SHR,  PIC,0

                                                0004 00000      .ENTRY  LGI$VALIDATE, Save R2                   ; 0098
                        52      0C      AC      D0 00002        MOVL    P_UAFRECORD, R2                         ; C147
                                        52      DD 00006        PUSHL   R2                                      ;
                        7E      04      AC      7D 00008        MOVQ    P_USERNAME, -(SP)                       ;
              0000V     CF              03      FB 0000C        CALLS   #3, LGI$SEARCHUSER                      ;
                        0B              50      E9 00011        BLBC    STATUS, 1$                              ; 0148
                                04      A2      DD 00014        PUSHL   4(R2)                                   ; 0149
                                08      AC      DD 00017        PUSHL   P_PASSWORD                              ;
              0000V     CF              02      FB 0001A        CALLS   #2, LGI$CHECK_PASS                      ;
                                        04 0001F 1$:            RET                                             ; 0152

; Routine Size:  32 bytes,    Routine Base:  _LIB$CODE + 0000
```

```
;  155          0153  1
;  156          0154  1 global routine LGI$SEARCHUSER
;  157          0155  1    (P_USERNAME, P_PASSWORD, P_UAFRECORD, P_FAB, P_RAB, PASS_LC) =
;  158          0156  2 begin
;  159          0157  2
;  160          0158  2 !++
;  161          0159  2 !
;  162          0160  2 ! FUNCTIONAL DESCRIPTION:
;  163          0161  2 !
;  164          0162  2 !       This routine reads the UAF record for the specified user.
;  165          0163  2 !
;  166          0164  2 ! CALLING SEQUENCE:
;  167          0165  2 !
;  168          0166  2 !    LGI$SEARCHUSER(P_USERNAME, P_PASSWORD, P_UAFRECORD, P_FAB, P_RAB, PASS_LC)
;  169          0167  2 !
;  170          0168  2 ! INPUT PARAMETERS:
;  171          0169  2 !
;  172          0170  2 !       P_USERNAME: address of string descriptor of username
;  173          0171  2 !       P_PASSWORD: address of string descriptor of password (not used)
;  174          0172  2 !       P_FAB:      address of FAB to use when opening the UAF
;  175          0173  2 !       P_RAB:      address of RAB to use when connecting to the UAF
;  176          0174  2 !       PASS_LC:    the presence (regardless of value) of this parameter
;  177          0175  2 !                       indicates that no case conversion is to be done.
;  178          0176  2 !
;  179          0177  2 ! IMPLICIT INPUTS:
;  180          0178  2 !
;  181          0179  2 !       NONE
;  182          0180  2 !
;  183          0181  2 ! OUTPUT PARAMETERS:
;  184          0182  2 !
;  185          0183  2 !       P_UAFRECORD: address of descriptor of buffer into which to read UAF record
;  186          0184  2 !
;  187          0185  2 ! IMPLICIT OUTPUTS:
;  188          0186  2 !
;  189          0187  2 !       NONE
;  190          0188  2 !
;  191          0189  2 ! ROUTINE VALUE:
;  192          0190  2 !
;  193          0191  2 !       1 if successful
;  194          0192  2 !       -2 if invalid username
;  195          0193  2 !       RMS$_xxx if errors reading the UAF
;  196          0194  2 !
;  197          0195  2 ! SIDE EFFECTS:
;  198          0196  2 !
;  199          0197  2 !       NONE
;  200          0198  2 !
;  201          0199  2 !--
;  202          0200  2
;  203          0201  2    builtin
;  204          0202  2
;  205          0203  2       NULLPARAMETER;
;  206          0204  2
;  207          0205  2    map
;  208          0206  2
;  209          0207  2       P_USERNAME: ref $BBLOCK,
;  210          0208  2       P_UAFRECORD: ref $BBLOCK,
;  211          0209  2       P_FAB: ref $BBLOCK,
```

```
;  212          210  2        P_RAB: ref $BBLOCK;
;  213          0211  2
;  214          0212  2    label
;  215          0213  2
;  216          0214  2        READ_UAF;
;  217          0215  2
;  218          0216  2    local
;  219          0217  2
;  220          0218  2        STATUS,                                 ! Routine status value
;  221          0219  2        PREV_PRIVS: vector[2],                  ! Saved process privileges
;  222          0220  2        USERNAME_LENGTH,                        ! Length of username
;  223          0221  2        USER_NAME: vector[UAF$S_USERNAME, byte], ! Buffer for username
;  224          0222  2        FAB: ref $BBLOCK,                       ! Pointer to FAB for the UAF
;  225          0223  2        RAB: ref $BBLOCK,                       ! Pointer to RAB for the UAF
;  226          0224  2        LOCAL_FAB: $FAB_DECL preset([FAB$W_IFI] = 0),     ! FAB to open UAF
;  227          0225  2        LOCAL_RAB: $RAB_DECL preset([RAB$W_ISI] = 0);     ! RAB to read record
;  228          0226  2
;  229          0227  2    bind
;  230          0228  2
;  231          0229  2        SYSPRV = uplit(1^($BITPOSITION(PRV$V_SYSPRV)), 0),
;  232          0230  2        DEFAULT_USER      = uplit byte('DEFAULT');
;  233          0231  2
;  234          0232  2 !
;  235          0233  2 !       Prepare to access UAF through FAB and RAB
;  236          0234  2 !
;  237          0235  2
;  238          0236  2 READ_UAF:
;  239          0237  3    begin
;  240          0238  3
;  241          0239  3 !
;  242          0240  3 !       Which FAB and RAB should we use?
;  243          0241  3 !
;  244          0242  3
;  245          0243  3        if NULLPARAMETER(4) then
;  246          0244  4           begin
;  247          0245  4              FAB = LOCAL_FAB;
;  248          0246  4              RAB = LOCAL_RAB;
;  249          0247  4           end
;  250          0248  3        else
;  251          0249  4           begin
;  252          0250  4              FAB = .P_FAB;
;  253          0251  4              RAB = .P_RAB;
;  254          0252  3           end;
;  255          0253  3
;  256          0254  3 !
;  257          0255  3 !       Is the UAF already open?
;  258          0256  3 !
;  259          0257  3
;  260          0258  3        if .FAB[FAB$W_IFI] eql 0 then
;  261          0259  4           begin
;  262          0260  4
;  263        P 0261  4              $FAB_INIT(fab = .FAB,
;  264        P 0262  4                        fnm = 'SYSUAF',
;  265        P 0263  4                        dnm = 'SYS$SYSTEM:.DAT',
;  266        P 0264  4                        fac = (GET, UPD),
;  267          0265  4                        shr = (GET, PUT, UPD, DEL));
;  268          0266  4
```

```
;  269          0267  4              RAB[RAB$W_ISI] = 0;
;  270          0268  4              FAB[FAB$V_LNM_MODE] = PSL$C_EXEC;
;  271          0269  4
;  272        P 0270  4              $SETPRV(enbflg = 1,
;  273        P 0271  4                      prvadr = SYSPRV,
;  274          0272  4                      prvprv = PREV_PRIVS);
;  275          0273  4
;  276          0274  4 !
;  277          0275  4 ! Open the UAF. If errors indicate problems with file sharing, try opening
;  278          0276  4 ! it non-shared.
;  279          0277  4 !
;  280          0278  4
;  281          0279  4              STATUS = $OPEN(fab = .FAB);
;  282          0280  4
;  283          0281  4              if .STATUS eql RMS$_SNE
;  284          0282  4              or .STATUS eql RMS$_SPE
;  285          0283  4              or .STATUS eql RMS$_DME then
;  286          0284  5                 begin
;  287          0285  5                    FAB[FAB$B_SHR] = 0;
;  288          0286  5                    STATUS = $OPEN(fab = .FAB);
;  289          0287  4                 end;
;  290          0288  4
;  291          0289  4              if not .STATUS then
;  292          0290  4                 leave READ_UAF;
;  293          0291  4
;  294          0292  3           end;
;  295          0293  3
;  296          0294  3 !
;  297          0295  3 !       Is the current RAB connected to the UAF?
;  298          0296  3 !
;  299          0297  3
;  300          0298  3        if .RAB[RAB$W_ISI] eql 0 then
;  301          0299  4           begin
;  302          0300  4
;  303        P 0301  4              $RAB_INIT(rab = .RAB,
;  304        P 0302  4                        fab = .FAB,
;  305        P 0303  4                        rac = KEY,
;  306          0304  4                        rop = (NLK, WAT));
;  307          0305  4
;  308          0306  4              STATUS = $CONNECT(rab = .RAB);
;  309          0307  4
;  310          0308  4              if .STATUS eql RMS$_CRMP then
;  311          0309  5                 begin
;  312          0310  5                    FAB[FAB$W_GBC] = 0;
;  313          0311  5                    STATUS = $CONNECT(rab = .RAB);
;  314          0312  4                 end;
;  315          0313  4
;  316          0314  4              if not .STATUS then
;  317          0315  4                 leave READ_UAF;
;  318          0316  4
;  319          0317  3           end;
;  320          0318  3
;  321          0319  3 !
;  322          0320  3 !       Attach key and buffer to RAB
;  323          0321  3 !
;  324          0322  3
;  325          0323  3        RAB[RAB$W_USZ] = .P_UAFRECORD[DSC$W_LENGTH];
```

```
;  326          0324  3        RAB[RAB$L_UBF] = .P_UAFRECORD[DSC$A_POINTER];
;  327          0325  3        RAB[RAB$B_KSZ] = UAF$S_USERNAME;
;  328          0326  3        RAB[RAB$L_KBF] = USER_NAME;
;  329          0327  3
;  330          0328  3    !
;  331          0329  3    !       Uppercase and blank fill the username string
;  332          0330  3    !
;  333          0331  3
;  334          0332  3        USERNAME_LENGTH = .P_USERNAME[DSC$W_LENGTH];
;  335          0333  3
;  336          0334  3        incr J to .USERNAME_LENGTH-1 do
;  337          0335  4           begin
;  338          0336  4
;  339          0337  4               local
;  340          0338  4                  C: byte;
;  341          0339  4
;  342          0340  4               C = .vector[.P_USERNAME[DSC$A_POINTER], .J;, byte];
;  343          0341  4
;  344          0342  4               if .C eql ' ' then
;  345          0343  4                  exitloop (USERNAME_LENGTH = .J);
;  346          0344  4
;  347          0345  4               if not NULLPARAMETER(6)
;  348          0346  4               and .C gequ 'a'
;  349          0347  4               and .C lequ 'z' then
;  350          0348  4                  USER_NAME[.J] = .C - 32
;  351          0349  4               else
;  352          0350  4                  USER_NAME[.J] = .C;
;  353          0351  4
;  354          0352  3           end;
;  355          0353  3
;  356          0354  3        ch$fill(' ', UAF$S_USERNAME-.USERNAME_LENGTH, USER_NAME[.USERNAME_LENGTH]);
;  357          0355  3
;  358          0356  3    !
;  359          0357  3    !       Read the UAF record. If not present, read the default record instead.
;  360          0358  3    !
;  361          0359  3
;  362          0360  3        STATUS = $GET(rab = .RAB);
;  363          0361  3
;  364          0362  3        if .STATUS eql RMS$_RNF then
;  365          0363  4           begin
;  366          0364  4              USERNAME_LENGTH = 7;
;  367          0365  4              ch$copy(7, DEFAULT_USER, ' ', UAF$S_USERNAME, USER_NAME);
;  368          0366  4              STATUS = $GET(rab = .RAB);
;  369          0367  4              if .STATUS then
;  370          0368  4                 STATUS = -2;
;  371          0369  3           end;
;  372          0370  3
;  373          0371  3    !
;  374          0372  3    ! Check for username = DEFAULT, and return failure if so.
;  375          0373  3    !
;  376          0374  3
;  377          0375  3        if .STATUS
;  378          0376  3        and ch$eql(.USERNAME_LENGTH, USER_NAME, 7, DEFAULT_USER, ' ') then
;  379          0377  3           STATUS = -2,
;  380          0378  3
;  381          0379  2    end;                                    ! end of block READ_UAF
;  382          0380  2
```

```
;  383          0381  2 !
;  384          0382  2 ! Close the UAF (if necessary) and restore the caller's privileges.
;  385          0383  2 !
;  386          0384  2
;  387          0385  2     if NULLPARAMETER(4) then
;  388          0386  3        begin
;  389          0387  3           $DISCONNECT(rab = .RAB);
;  390          0388  3           $CLOSE(fab = .FAB);
;  391          0389  2        end;
;  392          0390  2
;  393          0391  2     $SETPRV(enbflg = 0, prvadr = uplit(0, 0));
;  394          0392  2     $SETPRV(enbflg = 1, prvadr = PREV_PRIVS);
;  395          0393  2
;  396          0394  2     return .STATUS
;  397          0395  2
;  398          0396  1 end;                                    ! End of routine LGI$SEARCHUSER


                                                    00# 00020 P.AAA:  .BYTE   0[2]
                                                0000  00022         .WORD   0
                                                    00# 00024 P.AAB:  .BYTE   0[2]
                                                0000  00026         .WORD   0
                                  00000000  10000000  00028 P.AAC:  .LONG   268435456, 0
                              54  4C  55  41  46  45  44  00030 P.AAD:  .ASCII  \DEFAULT\
                                  46  41  55  53  59  53  00037 P.AAE:  .ASCII  \SYSUAF\
54  41  44  2E  3A  4D  45  54  53  59  53  24  53  59  53  0003D P.AAF:  .ASCII  \SYS$SYSTEM:.DAT\
                                  00000000  00000000  0004C P.AAG:  .LONG   0, 0

                                                        SYSPRV=          P.AAC
                                                        DEFAULT_USER=    P.AAD
                                                                .EXTRN  SYS$SETPRV, SYS$OPEN
                                                                .EXTRN  SYS$CONNECT, SYS$GET
                                                                .EXTRN  SYS$DISCONNECT, SYS$CLOSE

                                                    0FFC 00000          .ENTRY  LGI$SEARCHUSER, Save R2,R3,R4,R5,R6,R7,R8,- ; 0154
                                                                                R9,R10,R11
                          5B 00000000G  00  9E 00002        MOVAB   SYS$SETPRV, R11                             ;
                          5A        D0  AF  9E 00009        MOVAB   DEFAULT_USER, R10                           ;
                          5E      FF44  CE  9E 0000D        MOVAB   -188(SP), SP                                ;
0050  8F        00    F0  AA            04  2C 00012        MOVC5   #4, P.AAA, #0, #80, LOCAL_FAB               ; 0224
                              44        AE      0001A
0044  8F        00    F4  AA            04  2C 0001C        MOVC5   #4, P.AAB, #0, #68, LOCAL_RAB               ; 0225
                                        6E      00024
                          04            6C  91 00025        CMPB    (AP), #4                                    ; 0243
                                        05  1F 00028        BLSSU   1$                                          ;
                              10        AC  D5 0002A        TSTL    16(AP)                                      ;
                                        09  12 0002D        BNEQ    2$                                          ;
                          57  44        AE  9E 0002F 1$:    MOVAB   LOCAL_FAB, FAB                              ; 0245
                          58            6E  9E 00033        MOVAB   LOCAL_RAB, RAB                              ; 0246
                                        04  11 00036        BRB     3$                                          ; 0243
                          57  10        AC  7D 00038 2$:    MOVQ    P_FAB, FAB                                  ; 0250
                              02        A7  B5 0003C 3$:    TSTW    2(FAB)                                      ; 0258
                                        76  12 0003F        BNEQ    6$                                          ;
0050  8F        00        6E            00  2C 00041        MOVC5   #0, (SP), #0, #80, (FAB)                    ; 0265
                                        67      00048
                          67      5003  8F  B0 00049        MOVW    #20483, (FAB)                               ;
```

```
                      16  A7     0F0A  8F  B0 0004E       MOVW    #3850, 22(FAB)                      :
                      1F  A7           02  90 00054       MOVB    #2, 31(FAB)                         :
                      2C  A7       07  AA  9E 00058       MOVAB   P.AAE, 44(FAB)                      :
                      30  A7       0D  AA  9E 0005D       MOVAB   P.AAF, 48(FAB)                      :
                      34  A7     0F06  8F  B0 00062       MOVW    #3846, 52(FAB)                      :
                                   02  A8  B4 00068       CLRW    2(RAB)                              : 0267
   4A   A7        02          00       01  F0 0006B       INSV    #1, #0, #2, 74(FAB)                 : 0268
                                   F8  AD  9F 00071       PUSHAB  PREV_PRIVS                          : 0272
                                       7E  D4 00074       CLRL    -(SP)                               :
                                   F8  AA  9F 00076       PUSHAB  SYSPRV                              :
                                       01  DD 00079       PUSHL   #1                                  :
                              6B       04  FB 0007B       CALLS   #4, SYS$SETPRV                      :
                                       57  DD 0007E       PUSHL   FAB                                 : 0279
              00000000G  00            01  FB 00080       CALLS   #1, SYS$OPEN                        :
                         59            50  D0 00087       MOVL    R0, STATUS                          :
              0001879C   8F            59  D1 0008A       CMPL    STATUS, #100252                     : 0281
                                       12  13 00091       BEQL    4$                                  :
              000187A4   8F            59  D1 00093       CMPL    STATUS, #100260                     : 0282
                                       09  13 0009A       BEQL    4$                                  :
              000184D4   8F            59  D1 0009C       CMPL    STATUS, #99540                      : 0283
                                       0F  12 000A3       BNEQ    5$                                  :
                                   17  A7  94 000A5 4$:   CLRB    23(FAB)                             : 0285
                                       57  DD 000A8       PUSHL   FAB                                 : 0286
              00000000G  00            01  FB 000AA       CALLS   #1, SYS$OPEN                        :
                         59            50  D0 000B1       MOVL    R0, STATUS                          :
                         46            59  E9 000B4 5$:   BLBC    STATUS, 7$                          : 0289
                                   02  A8  B5 000B7 6$:   TSTW    2(RAB)                              : 0298
                                       47  12 000BA       BNEQ    8$                                  :
 0044   8F        00          6E       00  2C 000BC       MOVC5   #0, (SP), #0, #68, (RAB)            : 0304
                                       68     000C3                                               :
                         68      4401  8F  B0 000C4       MOVW    #17409, (RAB)                       :
                      04  A8 00120000  8F  D0 000C9       MOVL    #1179648, 4(RAB)                    :
                      1E  A8           01  90 000D1       MOVB    #1, 30(RAB)                         :
                      3C  A8           57  D0 000D5       MOVL    FAB, 60(RAB)                        :
                                       58  DD 000D9       PUSHL   RAB                                 : 0306
              00000000G  00            01  FB 000DB       CALLS   #1, SYS$CONNECT                     :
                         59            50  D0 000E2       MOVL    R0, STATUS                          :
              0001C14C   8F            59  D1 000E5       CMPL    STATUS, #115020                     : 0308
                                       0F  12 000EC       BNEQ    7$                                  :
                                   48  A7  B4 000EE       CLRW    72(FAB)                             : 0310
                                       58  DD 000F1       PUSHL   RAB                                 : 0311
              00000000G  00            01  FB 000F3       CALLS   #1, SYS$CONNECT                     :
                         59            50  D0 000FA       MOVL    R0, STATUS                          :
                         03            59  E8 000FD 7$:   BLBS    STATUS, 8$                          : 0314
                                 00A8  31 00100           BRW     15$                                 :
                         50        0C  AC  D0 00103 8$:   MOVL    P_UAFRECORD, R0                     : 0323
                      20  A8           60  B0 00107       MOVW    (R0), 32(RAB)                       :
                      24  A8       04  A0  D0 0010B       MOVL    4(R0), 36(RAB)                      : 0324
                      34  A8           20  90 00110       MOVB    #32, 52(RAB)                        : 0325
                      30  A8       D8  AD  9E 00114       MOVAB   USER_NAME, 48(RAB)                  : 0326
                         56        04  BC  3C 00119       MOVZWL  @P_USERNAME, USERNAME_LENGTH        : 0332
                         50            56  D0 0011D       MOVL    USERNAME_LENGTH, R0                 : 0334
                         51        04  AC  D0 00120       MOVL    P_USERNAME, R1                      : 0340
                         53            01  CE 00124       MNEGL   #1, J                               :
                                       32  11 00127       BRB     12$                                 :
                         52    04 B143  90 00129 9$:      MOVB    @4(R1)[J], C                        :
                         20            52  91 0012E       CMPB    C, #32                              : 0342
```

```
                                      05  12 00131         BNEQ    10$                                          ;
                             56       53  D0 00133         MOVL    J, USERNAME_LENGTH                           ; 0343
                                      27  11 00136         BRB     13$                                          ;
                             06       6C  91 00138  10$:   CMPB    (AP), #6                                     ; 0345
                                      19  1F 0013B         BLSSU   11$                                          ;
                                  18  AC  D5 0013D         TSTL    24(AP)                                       ;
                                      14  13 00140         BEQL    11$                                          ;
                        61   8F       52  91 00142         CMPB    C, #97                                       ; 0346
                                      0E  1F 00146         BLSSU   11$                                          ;
                        7A   8F       52  91 00148         CMPB    C, #122                                      ; 0347
                                      08  1A 0014C         BGTRU   11$                                          ;
         D8 AD43             52       20  83 0014E         SUBB3   #32, C, USER_NAME[J]                         ; 0348
                                      05  11 00154         BRB     12$                                          ;
                        D8 AD43       52  90 00156  11$:   MOVB    C, USER_NAME[J]                              ; 0350
                   CA        53       50  F2 0015B  12$:   AOBLSS  R0, J, 9$                                    ; 0334
                   50        20       56  C3 0015F  13$:   SUBL3   USERNAME_LENGTH, #32, R0                     ; 0354
     50            20        6E       00  2C 00163         MOVC5   #0, (SP), #32, R0, USER_NAME-                ;
                             D8 AD46         00168                 [USERNAME_LENGTH]                            ;
                                      58  DD 0016B         PUSHL   RAB                                          ; 0360
                 00000000G   00       01  FB 0016D         CALLS   #1, SYS$GET                                  ;
                             59       50  D0 00174         MOVL    R0, STATUS                                   ;
                 000182B2    8F       59  D1 00177         CMPL    STATUS, #98994                               ; 0362
                                      1C  12 0017E         BNEQ    14$                                          ;
                             56       07  D0 00180         MOVL    #7, USERNAME_LENGTH                          ; 0364
     20            20        6A       07  2C 00183         MOVC5   #7, DEFAULT_USER, #32, #32, USER_NAME        ; 0365
                               D8     AD     00188                                                              ;
                                      58  DD 0018A         PUSHL   RAB                                          ; 0366
                 00000000G   00       01  FB 0018C         CALLS   #1, SYS$GET                                  ;
                             59       50  D0 00193         MOVL    R0, STATUS                                   ;
                             12       59  E9 00196         BLBC    STATUS, 15$                                  ; 0367
                             59       02  CE 00199         MNEGL   #2, STATUS                                   ; 0368
                             0C       59  E9 0019C  14$:   BLBC    STATUS, 15$                                  ; 0375
     07            20   D8   AD       56  2D 0019F         CMPC5   USERNAME_LENGTH, USER_NAME, #32, #7, -       ; 0376
                                      6A     001A5                 DEFAULT_USER                                 ;
                                      03  12 001A6         BNEQ    15$                                          ;
                             59       02  CE 001A8         MNEGL   #2, STATUS                                   ; 0377
                             04       6C  91 001AB  15$:   CMPB    (AP), #4                                     ; 0385
                                      05  1F 001AE         BLSSU   16$                                          ;
                                  10  AC  D5 001B0         TSTL    16(AP)                                       ;
                                      12  12 001B3         BNEQ    17$                                          ;
                                      58  DD 001B5  16$:   PUSHL   RAB                                          ; 0387
                 00000000G   00       01  FB 001B7         CALLS   #1, SYS$DISCONNECT                           ;
                                      57  DD 001BE         PUSHL   FAB                                          ; 0388
                 00000000G   00       01  FB 001C0         CALLS   #1, SYS$CLOSE                                ;
                                      7E  7C 001C7  17$:   CLRQ    -(SP)                                        ; 0391
                                  1C  AA  9F 001C9         PUSHAB  P.AAG                                        ;
                                      7E  D4 001CC         CLRL    -(SP)                                        ;
                             6B       04  FB 001CE         CALLS   #4, SYS$SETPRV                               ;
                                      7E  7C 001D1         CLRQ    -(SP)                                        ; 0392
                                  F8  AD  9F 001D3         PUSHAB  PREV_PRIVS                                   ;
                                      01  DD 001D6         PUSHL   #1                                           ;
                             6B       04  FB 001D8         CALLS   #4, SYS$SETPRV                               ;
                             50       59  D0 001DB         MOVL    STATUS, R0                                   ; 0394
                                          04 001DE         RET                                                  ; 0396

; Routine Size:  479 bytes,    Routine Base:  _LIB$CODE + 0054
```

; 399 0397 1

```
;  401     0398  1 GLOBAL ROUTINE LGI$CHECK_PASS (P_PASSWORD, UAF_RECORD, PWD_NUMBER) =
;  402     0399  1
;  403     0400  1 !++
;  404     0401  1 !
;  405     0402  1 ! FUNCTIONAL DESCRIPTION:
;  406     0403  1 !
;  407     0404  1 !       This routine reads the UAF record for the specified user.
;  408     0405  1 !
;  409     0406  1 ! CALLING SEQUENCE:
;  410     0407  1 !       LGI$CHECK_PASS (P_PASSWORD, UAF_RECORD)
;  411     0408  1 !
;  412     0409  1 ! INPUT PARAMETERS:
;  413     0410  1 !       P_PASSWORD: address of string descriptor of password
;  414     0411  1 !       UAF_RECORD: address of UAF record
;  415     0412  1 !       PWD_NUMBER: password number, 0 - primary, 1 - secondary
;  416     0413  1 !
;  417     0414  1 ! IMPLICIT INPUTS:
;  418     0415  1 !       NONE
;  419     0416  1 !
;  420     0417  1 ! OUTPUT PARAMETERS:
;  421     0418  1 !       NONE
;  422     0419  1 !
;  423     0420  1 ! IMPLICIT OUTPUTS:
;  424     0421  1 !       NONE
;  425     0422  1 !
;  426     0423  1 ! ROUTINE VALUE:
;  427     0424  1 !       1 if successful
;  428     0425  1 !       -4 if invalid password
;  429     0426  1 !
;  430     0427  1 ! SIDE EFFECTS:
;  431     0428  1 !       NONE
;  432     0429  1 !
;  433     0430  1 !--
;  434     0431  1
;  435     0432  2 BEGIN
;  436     0433  2
;  437     0434  2 MAP
;  438     0435  2         P_PASSWORD      : REF $BBLOCK,
;  439     0436  2         UAF_RECORD      : REF $BBLOCK;
;  440     0437  2
;  441     0438  2 LITERAL
;  442     0439  2         MAX_PASSWORD    = 80;           ! maximum password accepted
;  443     0440  2
;  444     0441  2 LOCAL
;  445     0442  2         PASSWORD_LENGTH,                ! length of password string
;  446     0443  2         PASSWORD        : VECTOR [MAX_PASSWORD, BYTE], ! password string buffer
;  447     0444  2         PASSWORD_DESC   : VECTOR [2],   ! password string descriptor
;  448     0445  2         USERNAME_DESC   : VECTOR [2],   ! username string descriptor
;  449     0446  2         ENCRYPT_BUF     : VECTOR [2],   ! buffer for encrypted password
;  450     0447  2         ENCRYPT_DESC    : VECTOR [2];   ! descriptor for above
;  451     0448  2
;  452     0449  2 EXTERNAL ROUTINE
;  453     0450  2         LGI$HPWD;                       ! encrypt password
;  454     0451  2
;  455     0452  2 $ASSUME ($BYTEOFFSET (uaf$q_pwd2), EQL, $BYTEOFFSET (uaf$q_pwd)+8);
;  456     0453  2 $ASSUME ($BYTEOFFSET (uaf$b_encrypt2), EQL, $BYTEOFFSET (uaf$b_encrypt)+1);
;  457     0454  2
```

```
;  458          0455  2 ! Upcase and length check the password string.
;  459          0456  2 !
;  460          0457  2
;  461          0458  2 PASSWORD_LENGTH = MINU (.P_PASSWORD[DSC$W_LENGTH], MAX_PASSWORD);
;  462          0459  2 INCR J FROM 0 TO .PASSWORD_LENGTH
;  463          0460  2 DO
;  464          0461  3     BEGIN
;  465          0462  3     LOCAL C : BYTE;
;  466          0463  3     C = .VECTOR [.P_PASSWORD[DSC$A_POINTER], .J; ,BYTE];
;  467          0464  3     IF .C GEQU 'a'
;  468          0465  3     AND .C LEQU 'z'
;  469          0466  3     THEN PASSWORD[.J] = .C - 32
;  470          0467  3     ELSE PASSWORD[.J] = .C;
;  471          0468  2     END;
;  472          0469  2
;  473          0470  2 ! Set up username and password string descriptors. Note the encryption
;  474          0471  2 ! algorithm determines the format of username used - the old PURDY
;  475          0472  2 ! algorithm uses a 12 character blank-filled username, while the new
;  476          0473  2 ! version uses a variable trimmed username.
;  477          0474  2 !
;  478          0475  2
;  479          0476  2 ENCRYPT_DESC[0] = 8;
;  480          0477  2 ENCRYPT_DESC[1] = ENCRYPT_BUF;
;  481          0478  2
;  482          0479  2 PASSWORD_DESC[0] = .PASSWORD_LENGTH;
;  483          0480  2 PASSWORD_DESC[1] = PASSWORD;
;  484          0481  2
;  485          0482  2 USERNAME_DESC[0] = 12;
;  486          0483  2 USERNAME_DESC[1] = UAF_RECORD[UAF$T_USERNAME];
;  487          0484  2
;  488          0485  2 IF .VECTOR [UAF_RECORD[UAF$B_ENCRYPT], .PWD_NUMBER; ,BYTE] NEQ UAF$C_PURDY
;  489          0486  2 THEN
;  490          0487  3     BEGIN
;  491          0488  3     BUILTIN LOCC, R0;
;  492          0489  3     LOCC (%REF (' '), %REF (UAF$S_USERNAME), UAF_RECORD[UAF$T_USERNAME]);
;  493          0490  3     USERNAME_DESC[0] = UAF$S_USERNAME - .R0;
;  494          0491  2     END;
;  495          0492  2
;  496          0493  2 ! Finally encrypt the password and check it.
;  497          0494  2 !
;  498          0495  2
;  499          0496  2 LGI$HPWD (ENCRYPT_DESC,
;  500          0497  2           PASSWORD_DESC,
;  501          0498  2           .VECTOR [UAF_RECORD[UAF$B_ENCRYPT], .PWD_NUMBER; ,BYTE],
;  502          0499  2           .UAF_RECORD[UAF$W_SALT],
;  503          0500  2           USERNAME_DESC);
;  504          0501  2
;  505          0502  2 IF CH$EQL (8, ENCRYPT_BUF,
;  506          0503  2            UAF$S_PWD, UAF_RECORD[UAF$Q_PWD] + (.PWD_NUMBER * 8))
;  507          0504  2 THEN 1
;  508          0505  2 ELSE -4
;  509          0506  2
;  510          0507  1 END;                                    ! End of routine LGI$CHECK_PASS


                                                                .EXTRN  LGI$HPWD
```

```
                                   000C 00000       .ENTRY  LGI$CHECK_PASS, Save R2,R3                    : 0398
                5E       90    AE    9E 00002       MOVAB   -112(SP), SP                                  :
                50       04    BC    3C 00006       MOVZWL  @P_PASSWORD, R0                               : 0458
          0050  8F             50    B1 0000A       CMPW    R0, #80                                       :
                               04    1B 0000F       BLEQU   1$                                            :
                50       50    8F    9A 00011       MOVZBL  #80, R0                                       :
                53             50    D0 00015 1$:   MOVL    R0, PASSWORD_LENGTH                           :
                51       04    AC    D0 00018       MOVL    P_PASSWORD, R1                                : 0463
                50             01    CE 0001C       MNEGL   #1, J                                         :
                               1E    11 0001F       BRB     4$                                            :
                52       04 B140     90 00021 2$:   MOVB    @4(R1)[J], C                                  :
             61 8F             52    91 00026       CMPB    C, #97                                        : 0464
                               0E    1F 0002A       BLSSU   3$                                            :
             7A 8F             52    91 0002C       CMPB    C, #122                                       : 0465
                               08    1A 00030       BGTRU   3$                                            :
     20 AE40    52             20    83 00032       SUBB3   #32, C, PASSWORD[J]                           : 0466
                               05    11 00038       BRB     4$                                            :
           20 AE40             52    90 0003A 3$:   MOVB    C, PASSWORD[J]                                : 0467
        DE      50             53    F3 0003F 4$:   AOBLEQ  PASSWORD_LENGTH, J, 2$                        : 0459
                6E             08    D0 00043       MOVL    #8, ENCRYPT_DESC                              : 0476
             04 AE       08    AE    9E 00046       MOVAB   ENCRYPT_BUF, ENCRYPT_DESC+4                   : 0477
             18 AE             53    D0 0004B       MOVL    PASSWORD_LENGTH, PASSWORD_DESC                : 0479
             1C AE       20    AE    9E 0004F       MOVAB   PASSWORD, PASSWORD_DESC+4                     : 0480
             10 AE             0C    D0 00054       MOVL    #12, USERNAME_DESC                            : 0482
                53       08    AC    D0 00058       MOVL    UAF_RECORD, R3                                : 0483
             14 AE       04    A3    9E 0005C       MOVAB   4(R3), USERNAME_DESC+4                        :
                52       0C    AC    D0 00061       MOVL    PWD_NUMBER, R2                                : 0485
                01     0168 C243     91 00065       CMPB    360(R2)[R3], #1                               :
                               0A    13 0006B       BEQL    5$                                            :
     04   A3    20             20    3A 0006D       LOCC    #32, #32, 4(R3)                               : 0489
     10   AE    20             50    C3 00072       SUBL3   R0, #32, USERNAME_DESC                        : 0490
                         10    AE    9F 00077 5$:   PUSHAB  USERNAME_DESC                                 : 0496
                7E     0166    C3    3C 0007A       MOVZWL  358(R3), -(SP)                                : 0499
                7E     0168 C243     9A 0007F       MOVZBL  360(R2)[R3], -(SP)                            : 0498
                         24    AE    9F 00085       PUSHAB  PASSWORD_DESC                                 : 0496
                         10    AE    9F 00088       PUSHAB  ENCRYPT_DESC                                  :
      00000000G 00             05    FB 0008B       CALLS   #5, LGI$HPWD                                  :
                       0154 C342     7F 00092       PUSHAQ  340(R3)[R2]                                   : 0502
        9E   0C AE             08    29 00097       CMPC3   #8, ENCRYPT_BUF, @(SP)+                       :
                               04    12 0009C       BNEQ    6$                                            :
                50             01    D0 0009E       MOVL    #1, R0                                        :
                                     04 000A1       RET                                                   :
                50             04    CE 000A2 6$:   MNEGL   #4, R0                                        : 0505
                                     04 000A5       RET                                                   : 0507
```

```
; Routine Size:  166 bytes,    Routine Base:  _LIB$CODE + 0233

;  511          0508  1
;  512          0509  1 end
;  513          0510  1
;  514          0511  0 eludom
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _LIB$CODE                        729  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)


;                                 Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1          18619       101         0        1000        00:01.3



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

; Size:         677 code + 52 data bytes
; Run Time:        00:11.9
; Elapsed Time:    00:54.0
; Lines/CPU Min:    2576
; Lexemes/CPU-Min: 37689
; Memory Used:  207 pages
; Compilation Complete
```

DEFINE
MAR

LOGINCMD
LIS

ACTONE
LIS

MESSAGES
LIS

ACTCHR
LIS

VALIDATE
LIS

ACTIF
LIS

PROTCLI
LIS

CRFMACRO5
MAR

MACRO

MACRO32
MAP