

VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	0000000000	
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMMMMM	MMMMMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSS	LLL	III	0000000000	
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSS	LLL	III	000	000
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	
VVV		VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	0000000000	

[illegible]

(2)	76
(3)	166
(4)	254
(5)	314

DECLARATIONS

PRV\$SETPRIV - Set or Clear privilege bit
PRV\$PRIVBIT - Return privilege bit number
PRV\$KEYWORD - Translate privilege bit # to keyword

```
0000 1 .TITLE SETPRIV - PRIVILEGE RELATED PROCEDURES
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24
0000 25 *****
0000 26
0000 27
0000 28 ++
0000 29 FACILITY: System Subroutine
0000 30
0000 31 ABSTRACT:
0000 32
0000 33 This routine takes an ascii privilege name and sets or
0000 34 clears the specified bit in the supplied privilege vector.
0000 35
0000 36 ENVIRONMENT: mode of caller
0000 37
0000 38 AUTHOR: Henry M. Levy , CREATION DATE: 10-January-1978
0000 39
0000 40 MODIFIED BY:
0000 41
0000 42 V03-004 RSH0019 R.Scott Hanna 21-May-1983
0000 43 Add SECURITY privilege
0000 44
0000 45 V03-003 PRB0173 Paul Beck 26-APR-1983 20:43:31
0000 46 Add TMPJNL and PRMJNL privileges.
0000 47
0000 48 V03-002 LMP0083 L. Mark Pilant, 28-Feb-1983 9:18
0000 49 Add the new non-discretionary privileges.
0000 50
0000 51 V001 MLJ0085 Martin L. Jack 31-Mar-1982
0000 52 Change PSECT to _LIB$CODE.
0000 53
0000 54 V005 TMH0005 Tim Halvorsen 27-Oct-1981
0000 55 Fix checking for ALL keyword to happen before table lookup.
0000 56
0000 57 V004 TMH0004 Tim Halvorsen 26-Oct-1981
```


0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :
0000 72 :
0000 73 :
0000 74 :--

Add routine to return bit number associated with privilege keyword. Add routine to translate a privilege bit number into a privilege keyword string.
Add "ALL" to list of privileges.

V02-003 GWF0096 Gary Fowler 29-Jul-1981
Added SYSLCK privilege

V02-002 CNH0018 Chris Hume 12-Oct-1979 12:30
Added PRVSV_BYPASS and brought various privilege bit tables up to date. (VMSLIB]STARDEF.MDL is already up to date.)
(UAF]UAFMAIN.B32 02.02, [CLIUTL]SHOPROCES.MAR 01.03, RUNDET.MAR 01.02)

V02-001 GWF0001 Gary Fowler 30-May-1979
Added PFNMAP, SHMEM, and SYSPRV privileges

```
0000 76 .SBTTL DECLARATIONS
0000 77
00000000 78 .PSECT _LIB$CODE,PIC,USR,CON,REL,LCL,SHR,EXE,RD,NOWRT,NOVEC,2
0000 79
0000 80 :
0000 81 : INCLUDE FILES:
0000 82 :
0000 83
0000 84 $PRVDEF ; define privilege bits
0000 85
0000 86 :
0000 87 : MACROS:
0000 88 :
0000 89 :
0000 90 :
0000 91 : Macro to define entry to privilege name table.
0000 92 :
0000 93
0000 94 .MACRO PRIVDEF BIT,MIN,NAME
0000 95 .BYTE MIN
0000 96 .BYTE PRV$V_'BIT
0000 97 .IF NB NAME
0000 98 .ASCIC /NAME/
0000 99 .IFF
0000 100 .ASCIC /BIT/
0000 101 .ENDC
0000 102 .ENDM PRIVDEF
0000 103
0000 104 :
0000 105 : EQUATED SYMBOLS:
0000 106 :
00000004 0000 107 NAMDSC = 4 ; offset to string descriptor
00000008 0000 108 PRIVEC = 8 ; offset to privilege vector address
0000 109
00000002 0000 110 PRV$_INVNAM == 2 ; privilege name invalid
00000004 0000 111 PRV$_NOTUNQ == 4 ; privilege name not unique
0000 112
0000 113
0000 114 :
0000 115 : OWN STORAGE:
0000 116 :
0000 117
4C 4C 41 0000 118 ALLPRIV:
0000 119 .ASCII 'ALL' ; SETS/CLEARs ALL PRIVILEGES
0003 120
0003 121 PRV$AB_NAMES::
0003 122 PRIVDEF CMKRNL,3 ; MAY CHANGE MODE TO KERNEL
000C 123 PRIVDEF CMEXEC,3 ; MAY CHANGE MODE TO EXEC
0015 124 : ***** THE PRECEEDING TWO BITS MUST BE ADJACENT
0015 125 : ***** THE FOLLOWING TWO BITS MUST BE ADJACENT
0015 126 PRIVDEF SYSNAM,4 ; MAY INSERT IN SYSTEM LOGICAL NAME TABLE
001E 127 PRIVDEF GRPNAM,4 ; MAY INSERT IN GROUP LOGICAL NAME TABLE
0027 128 : ***** THE PRECEEDING TWO BITS MUST BE ADJACENT
0027 129 PRIVDEF ALLSPOOL,4 ; MAY ALLOCATE SPOOLED DEVICE
0032 130 PRIVDEF DETACH,3 ; MAY CREATE DETACHED PROCESSES
003B 131 PRIVDEF DIAGNOSE,3 ; MAY DIAGNOSE DEVICES
0046 132 PRIVDEF LOG_IO,3 ; MAY DO LOGICAL I/O
```


004F	133	PRIVDEF GROUP,3	: MAY AFFECT OTHER PROCESSES IN SAME GROUP
0057	134	PRIVDEF NOACNT,3,<ACNT>	: MAY SUPPRESS ACCOUNTING MESSAGE
005E	135	PRIVDEF PRMCEB,4	: MAY CREATE PERMANENT COMMON EVENT CLUSTERS
0067	136	PRIVDEF PRMMBX,4	: MAY CREATE PERMANENT MAILBOX
0070	137	PRIVDEF PSWAPM,3	: MAY CHANGE PROCESS SWAP MODE
0079	138	PRIVDEF SETPRI,3,<ALTPRI>	: MAY SET ANY PRIORITY VALUE
0082	139	PRIVDEF SETPRV,3	: MAY SET ANY PRIVILEGE BITS
008B	140	PRIVDEF TMPMBX,3	: MAY CREATE TEMPORARY MAILBOX
0094	141	PRIVDEF WORLD,3	: MAY AFFECT OTHER PROCESSES IN THE WORLD
009C	142	PRIVDEF OPER,3	: OPERATOR PRIVILEGE
00A3	143	PRIVDEF EXQUOTA,3	: MAY EXCEED QUOTAS
00AD	144	PRIVDEF NETMBX,3	: MAY CREATE NETWORK DEVICE
00B6	145	PRIVDEF VOLPRO,3	: MAY OVERRIDE VOLUME PROTECTION
00BF	146	PRIVDEF PHY_IO,3	: MAY DO PHYSICAL I/O
00C8	147	PRIVDEF BUGCHK,3	: MAY MAKE BUG CHECK ERROR LOG ENTRIES
00D1	148	PRIVDEF PRMGBL,4	: MAY CREATE PERMANENT GLOBAL SECTIONS
00DA	149	PRIVDEF SYSGBL,4	: MAY CREATE SYSTEM WIDE GLOBAL SECTIONS
00E3	150	PRIVDEF MOUNT,3	: MAY EXECUTE MOUNT VOLUME QIO
00EB	151	PRIVDEF PFNMAP,3	: MAY MAP TO SPECIFIC PHYSICAL PAGES
00F4	152	PRIVDEF SHMEM,3	: MAY CREATE/DELETE OBJECTS IN SHARED MEMORY
00FC	153	PRIVDEF SYSPRV,4	: MAY ACCESS OBJECTS VIA SYSTEM PROTECTION
0105	154	PRIVDEF BYPASS,3	: BYPASSES UIC CHECKING
010E	155	PRIVDEF SYSLCK,4	: SYSLCK privilege
0117	156	PRIVDEF SHARE,3	: MAY ASSIGN CHANNEL TO NON-SHARED DEVICE
011F	157	PRIVDEF UPGRADE,3	: May upgrade classification
0129	158	PRIVDEF DOWNGRADE,3	: May downgrade classification
0135	159	PRIVDEF GRPPRV,4	: Group access via system protection field
013E	160	PRIVDEF READALL,3	: Read access to everything
0148	161	PRIVDEF TMPJNL,4	: May create temporary journal
0151	162	PRIVDEF PRMJNL,4	: May create permanent journal
015A	163	PRIVDEF SECURITY,3	: May perform security functions
00 00	0165	.BYTE 0,0	: terminate table

```

0167 166 .SBTTL PRV$SETPRIV - Set or Clear privilege bit
0167 167 :++
0167 168 :
0167 169 : FUNCTIONAL DESCRIPTION:
0167 170 :
0167 171 : Set or clear privilege bit in specified privilege vector
0167 172 : depending on ascii privilege name.
0167 173 :
0167 174 : CALLING SEQUENCE:
0167 175 :
0167 176 : CALLS/CALLG
0167 177 :
0167 178 : INPUTS:
0167 179 :
0167 180 : NAMDSC(AP) = address of a string descriptor for the ascii
0167 181 : privilege name
0167 182 : PRIVEC = address of the privilege vector in which to clear
0167 183 : or set bits
0167 184 :
0167 185 : IMPLICIT INPUTS:
0167 186 :
0167 187 : none
0167 188 :
0167 189 : OUTPUTS:
0167 190 :
0167 191 : none
0167 192 :
0167 193 : IMPLICIT OUTPUTS:
0167 194 :
0167 195 : none
0167 196 :
0167 197 : ROUTINE VALUE:
0167 198 :
0167 199 : R0 =
0167 200 : LOW BIT SET -> success
0167 201 : LOW BIT CLEAR -> failure with:
0167 202 : PRV$_NOTUNQ -> privilege name is ambiguous
0167 203 : PRV$_INVNAM -> invalid privilege name
0167 204 :
0167 205 : SIDE EFFECTS:
0167 206 :
0167 207 : none
0167 208 :--
0167 209 :
0167 210 PRV$SETPRIV::
0167 211 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
54 04 AC 07FC 0169 212 MOVL NAMDSC(AP),R4 : get name string descriptor
54 5A 64 3C 016D 213 MOVZWL (R4),R10 : get string length
54 04 A4 DO 0170 214 MOVL 4(R4),R4 : get string address
55 FE8B CF DE 0174 215 MOVAL PRV$AB_NAMES,R5 : get address of name table
56 01 DO 0179 216 MOVL #1,R6 : assume 'NO' not specified
5A 02 B1 017C 217 CMPW #2,R10 : check at least 2 long
OF 14 017F 218 BGTR 10$ : not enough to check for 'NO'
64 4F4E 8F B1 0181 219 CMPW #^A/NO/,(R4) : see if bit to be turned off
08 12 0186 220 BNEQ 10$ : branch if not
56 D4 0188 221 CLRL R6 : else remember to turn off bit
54 02 C0 018A 222 ADDL #2,R4 : point string over 'NO'

```



```
00  FE6B 5A 02 C2 018D 223      SUBL    #2,R10      ; decrease length by 2
      CF 03 2D 0190 224 10$:  CMPC5    #3,W^ALLPRIV,#0,-    ; CHECK IF "ALL" KEYWORD
      64 5A 12 0196 225      BRB      30$      ;
      13 12 0198 226      BNEQ      30$      ; BRANCH IF NOT
      50 08 AC D0 019A 227      MOVL    PRIVEC(AP),R0      ; GET ADDRESS OF PRIVILEGE MASK
      08 56 E9 019E 228      BLBC     R6,20$      ; BRANCH IF WE MUST CLEAR ALL PRIVS
      80 01 CE 01A1 229      MNEGL    #1,(R0)+      ; SET ALL PRIVILEGES
      80 01 CE 01A4 230      MNEGL    #1,(R0)+
      28 11 01A7 231      BRB      90$      ; EXIT WITH SUCCESS
      60 7C 01A9 232 20$:  CLRQ     (R0)      ; CLEAR ALL PRIVILEGES
      24 11 01AB 233      BRB      90$      ; EXIT WITH SUCCESS
      01AD 234
      57 85 9A 01AD 235 30$:  MOVZBL   (R5)+,R7      ; get min string length
      23 13 01B0 236      BEQL      80$      ; continue if more in table
      58 85 9A 01B2 237      MOVZBL   (R5)+,R8      ; get privilege bit number
      59 85 9A 01B5 238      MOVZBL   (R5)+,R9      ; get cstring length
      65 64 5A 29 01B8 239      CMPC3    R10,(R4),(R5)    ; see if string match
      05 13 01BC 240      BEQL      40$      ; branch if strings match
      55 59 C0 01BE 241      ADDL      R9,R5      ; skip of string in table
      EA 11 01C1 242      BRB      30$      ; continue search
      01C3 243 40$:
      50 04 D0 01C3 244      MOVL      #PRV$_NOTUNQ,R0    ; assume not unique
      57 5A B1 01C6 245      CMPW      R10,R7      ; see if enough was specified
      09 19 01C9 246      BLSS      99$      ; return if not enough
      08 BC 01 58 56 F0 01CB 247      INSV     R6,R8,#1,@PRIVEC(AP) ; set or clear bit
      50 01 D0 01D1 248 90$:  MOVL      #1,R0      ; note success
      04 01D4 249 99$:  RET      ; return to caller
      01D5 250
      50 02 D0 01D5 251 80$:  MOVL      #PRV$_INVNAM,R0    ; table done, name not found
      04 01D8 252      RET
```



```
01D9 254 .SBTTL PRV$PRIVBIT - Return privilege bit number
01D9 255 :++
01D9 256 :
01D9 257 : FUNCTIONAL DESCRIPTION:
01D9 258 :
01D9 259 : Lookup the specified privilege keyword in the list of
01D9 260 : legal keywords and return the privilege bit number
01D9 261 : associated with the keyword.
01D9 262 :
01D9 263 : CALLING SEQUENCE:
01D9 264 :
01D9 265 : CALLS/CALLG
01D9 266 :
01D9 267 : INPUTS:
01D9 268 :
01D9 269 : 4(AP) = address of a string descriptor for the ascii
01D9 270 : privilege name
01D9 271 : 8(AP) = address of a word to receive the privilege bit
01D9 272 : number associated with the keyword.
01D9 273 :
01D9 274 : ROUTINE VALUE:
01D9 275 :
01D9 276 : RO = 1 -> bit number returned in RETNUM
01D9 277 : PRV$_NOTUNQ -> privilege name is ambiguous
01D9 278 : PRV$_INVNAM -> invalid privilege name
01D9 279 :--
01D9 280
01D9 281 PRV$PRIVBIT::
07FC 01D9 282 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
01DB 283
54 04 AC D0 01DB 284 MOVL 4(AP),R4 ; get name string descriptor
5A 64 3C 01DF 285 MOVZWL (R4),R10 ; get string length
54 04 A4 D0 01E2 286 MOVL 4(R4),R4 ; get string address
55 FE19 CF DE 01E6 287 MOVAL PRV$AB_NAMES,R5 ; get address of name table
5A 02 B1 01EB 288 CMPW #2,R10 ; check at least 2 long
0D 14 01EE 289 BGTR 10$ ; not enough to check for 'NO'
64 4F4E 8F B1 01F0 290 CMPW #^A/NO/, (R4) ; see if bit to be turned off
06 12 01F5 291 BNEQ 10$ ; branch if not
54 02 C0 01F7 292 ADDL #2,R4 ; point string over 'NO'
5A 02 C2 01FA 293 SUBL #2,R10 ; decrease length by 2
01FD 294 10$:
57 85 9A 01FD 295 MOVZBL (R5)+,R7 ; get min string length
21 13 0200 296 BEQL 60$ ; continue if more in table
58 85 9A 0202 297 MOVZBL (R5)+,R8 ; get privilege bit number
59 85 9A 0205 298 MOVZBL (R5)+,R9 ; get cstring length
65 64 5A 29 0208 299 CMPC3 R10,(R4),(R5) ; see if string match
05 13 020C 300 BEQL 30$ ; branch if strings match
55 59 C0 020E 301 ADDL R9,R5 ; skip of string in table
EA 11 0211 302 BRB 10$ ; continue search
0213 303 30$:
50 04 D0 0213 304 MOVL #PRV$_NOTUNQ,R0 ; assume not unique
57 5A B1 0216 305 CMPW R10,R7 ; see if enough was specified
07 19 0219 306 BLSS 40$ ; return if not enough
08 BC 58 D0 021B 307 MOVL R8,@8(AP) ; return bit number
50 01 D0 021F 308 MOVL #1,R0 ; note success
04 0222 309 40$: RET ; return to caller
0223 310
```


SETPRIV
V04-000

- PRIVILEGE RELATED PROCEDURES
PRV\$PRIVBIT - Return privilege bit numbe

D 14

16-SEP-1984 02:19:07 VAX/VMS Macro V04-00
5-SEP-1984 04:43:41 [VMSLIB.SRC]SETPRIV.MAR;1

Page 8
(4)

50 02 D0 0223 311 60\$: MOVL #PRV\$_INVNAM,R0 ; table done, name not found
04 0226 312 RET


```

0227 314 .SBTTL PRV$KEYWORD - Translate privilege bit # to keyword
0227 315 :++
0227 316 :
0227 317 : FUNCTIONAL DESCRIPTION:
0227 318 :
0227 319 : Return the keyword string associated with the given privilege
0227 320 : bit number.
0227 321 :
0227 322 : CALLING SEQUENCE:
0227 323 :
0227 324 : CALLS/CALLG
0227 325 :
0227 326 : INPUTS:
0227 327 :
0227 328 : 4(AP) = privilege mask bit number (0-63)
0227 329 : 8(AP) = address of a buffer descriptor to receive the keyword
0227 330 : 12(AP) = address of a word to receive the keyword length
0227 331 :
0227 332 : OUTPUTS:
0227 333 :
0227 334 : R0 = status (error if illegal privilege bit number)
0227 335 :
0227 336 : The keyword is stored into the buffer, and the length is
0227 337 : returned in the user-specified word.
0227 338 :--
0227 339 :
0227 340 PRV$KEYWORD::
0227 341 .WORD ^M<R2,R3,R4,R5>
0229 342
55 FDD6 CF DE 0229 343 MOVAL PRV$AB_NAMES,R5 ; get address of name table
85 04 AC 91 022E 344 10$: TSTB (R5)+ ; end of table?
50 85 9A 0230 345 BEQL 60$ ; if so, illegal bit number
55 50 C0 0232 346 CMPB 4(AP),(R5)+ ; does privilege bit number match?
EE 11 0236 347 BEQL 20$ ; branch if so
50 85 9A 0238 348 MOVZBL (R5)+,R0 ; get length of counted string
55 50 C0 023B 349 ADDL R0,R5 ; skip to next entry in table
EE 11 023E 350 BRB 10$ ; loop until end of table
50 85 9A 0240 351
51 08 AC D0 0240 352 20$: MOVZBL (R5)+,R0 ; get cstring length
61 50 B1 0243 353 MOVL 8(AP),R1 ; get address of buffer descriptor
50 03 15 0247 354 CMPW R0,(R1) ; is buffer too small?
50 61 3C 024A 355 BLEQ 25$ ; branch if ok
04 B1 0C BC 50 B0 024C 356 MOVZWL (R1),R0 ; truncate keyword to size of buffer
65 50 28 024F 357 25$: MOVW R0,@12(AP) ; return length of keyword
50 01 D0 0253 358 MOVC R0,(R5),@4(R1) ; copy keyword into buffer
04 04 025B 359 MOVL #1,R0 ; exit with success
50 0C BC B4 025C 360 RET
50 02 D0 025F 361
04 04 0262 362 60$: CLRW @12(AP) ; return nothing in buffer
0263 363 MOVL #PRV$_INVNAM,R0 ; table done, name not found
0263 364 RET
0263 365
0263 366 .END

```


SETPRIV
Symbol table

- PRIVILEGE RELATED PROCEDURES

F 14

16-SEP-1984 02:19:07
5-SEP-1984 04:43:41

VAX/VMS Macro V04-00
[VMSLIB.SRC]SETPRIV.MAR;1

Page 10
(5)

ALLPRIV	00000000	R	01
NAMDSC	= 00000004		
PRIVEC	= 00000008		
PRV\$AB_NAMES	00000003	RG	01
PRV\$KEYWORD	00000227	RG	01
PRV\$PRIVBIT	000001D9	RG	01
PRV\$SETPRIV	00000167	RG	01
PRV\$V_ALLSPOOL	= 00000004		
PRV\$V_BUGCHK	= 00000017		
PRV\$V_BYPASS	= 0000001D		
PRV\$V_CMEXEC	= 00000001		
PRV\$V_CMKRNL	= 00000000		
PRV\$V_DETACH	= 00000005		
PRV\$V_DIAGNOSE	= 00000006		
PRV\$V_DOWNGRADE	= 00000021		
PRV\$V_EXQUOTA	= 00000013		
PRV\$V_GROUP	= 00000008		
PRV\$V_GRPNAM	= 00000003		
PRV\$V_GRPDRV	= 00000022		
PRV\$V_LOG_IO	= 00000007		
PRV\$V_MOUNT	= 00000011		
PRV\$V_NETMBX	= 00000014		
PRV\$V_NOACNT	= 00000009		
PRV\$V_OPER	= 00000012		
PRV\$V_PFNMAP	= 0000001A		
PRV\$V_PHY_IO	= 00000016		
PRV\$V_PRCB	= 0000000A		
PRV\$V_PRCBL	= 00000018		
PRV\$V_PRCJNL	= 00000025		
PRV\$V_PRCMBX	= 0000000B		
PRV\$V_PSWAPM	= 0000000C		
PRV\$V_READALL	= 00000023		
PRV\$V_SECURITY	= 00000026		
PRV\$V_SETPRI	= 0000000D		
PRV\$V_SETPRV	= 0000000E		
PRV\$V_SHARE	= 0000001F		
PRV\$V_SHMEM	= 0000001B		
PRV\$V_SYSGBL	= 00000019		
PRV\$V_SYSLCK	= 0000001E		
PRV\$V_SYSNAM	= 00000002		
PRV\$V_SYSPRV	= 0000001C		
PRV\$V_TMPJNL	= 00000024		
PRV\$V_TMPMBX	= 0000000F		
PRV\$V_UPGRADE	= 00000020		
PRV\$V_VOLPRO	= 00000015		
PRV\$WORLD	= 00000010		
PRV\$INVNAM	= 00000002	G	
PRV\$NOTUNG	= 00000004	G	

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes												
ABS	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE		
_LIB\$CODE	00000263 (611.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG		

SETPRIV
Psect synopsis

- PRIVILEGE RELATED PROCEDURES

G 14

16-SEP-1984 02:19:07 VAX/VMS Macro V04-00
5-SEP-1984 04:43:41 [VMSLIB.SRC]SETPRIV.MAR;1

Page 11
(5)

\$AB\$\$

00000000 (0.) 02 (2.) NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	29	00:00:00.06	00:00:00.47
Command processing	105	00:00:00.54	00:00:02.42
Pass 1	160	00:00:02.31	00:00:05.26
Symbol table sort	0	00:00:00.08	00:00:00.28
Pass 2	91	00:00:00.96	00:00:01.94
Symbol table output	7	00:00:00.05	00:00:00.05
Psect synopsis output	5	00:00:00.03	00:00:00.23
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	400	00:00:04.05	00:00:10.66

The working set limit was 900 pages.

13208 bytes (26 pages) of virtual memory were used to buffer the intermediate code.

There were 10 pages of symbol table space allocated to hold 85 non-local and 15 local symbols.

366 source lines were read in Pass 1, producing 12 object records in Pass 2.

9 pages of virtual memory were used to define 8 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name

Macros defined

_ \$255\$DUA28:[SYSLIB]STARLET.MLB;2

4

135 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

MACRO/DISA=TRACE/LIS=LIS\$:SETPRIV/OBJ=OBJ\$:SETPRIV MSRC\$:SETPRIV/UPDATE=(ENH\$:SETPRIV)

0437 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY