```
EEEEEEEEEE  XX      XX    AAAAAA    MM      MM  PPPPPPPP    LL          EEEEEEEEEE    SSSSSSSS
EEEEEEEEEE  XX      XX    AAAAAA    MM      MM  PPPPPPPP    LL          EEEEEEEEEE    SSSSSSSS
EEEEEEEEEE  XX      XX    AAAAAA    MM      MM  PPPPPPPP    LL          EEEEEEEEEE    SSSSSSSS
EE          XX      XX  AA      AA  MMMM  MMMM  PP      PP  LL          EE          SS
EE          XX      XX  AA      AA  MMMM  MMMM  PP      PP  LL          EE          SS
EE          XX      XX  AA      AA  MMMM  MMMM  PP      PP  LL          EE          SS
EE            XX  XX    AA      AA  MM  MM  MM  PP      PP  LL          EE          SS
EE            XX  XX    AA      AA  MM  MM  MM  PP      PP  LL          EE          SS
EE            XX  XX    AA      AA  MM  MM  MM  PP      PP  LL          EE          SS
EEEEEEEE        XX      AA      AA  MM      MM  PPPPPPPP    LL          EEEEEEEE      SSSSSS
EEEEEEEE        XX      AA      AA  MM      MM  PPPPPPPP    LL          EEEEEEEE      SSSSSS
EEEEEEEE        XX      AA      AA  MM      MM  PPPPPPPP    LL          EEEEEEEE      SSSSSS
EE            XX  XX    AAAAAAAAAA  MM      MM  PP          LL          EE                  SS
EE            XX  XX    AAAAAAAAAA  MM      MM  PP          LL          EE                  SS
EE            XX  XX    AAAAAAAAAA  MM      MM  PP          LL          EE                  SS
EE          XX      XX  AA      AA  MM      MM  PP          LL          EE                  SS
EE          XX      XX  AA      AA  MM      MM  PP          LL          EE                  SS
EE          XX      XX  AA      AA  MM      MM  PP          LL          EE                  SS
EEEEEEEEEE  XX      XX  AA      AA  MM      MM  PP          LLLLLLLLLL  EEEEEEEEEE  SSSSSSSS
EEEEEEEEEE  XX      XX  AA      AA  MM      MM  PP          LLLLLLLLLL  EEEEEEEEEE  SSSSSSSS
EEEEEEEEEE  XX      XX  AA      AA  MM      MM  PP          LLLLLLLLLL  EEEEEEEEEE  SSSSSSSS
```

```
**FILE**ID**DOD_ERAPAT

DDDDDDDD      OOOOOO    DDDDDDDD                EEEEEEEEEE  RRRRRRRR      AAAAAA    PPPPPPPP      AAAAAA    TTTTTTTTTT
DDDDDDDD      OOOOOO    DDDDDDDD                EEEEEEEEEE  RRRRRRRR      AAAAAA    PPPPPPPP      AAAAAA    TTTTTTTTTT
DD      DD  OO      OO  DD      DD              EE          RR      RR  AA      AA  PP      PP  AA      AA      TT
DD      DD  OO      OO  DD      DD              EE          RR      RR  AA      AA  PP      PP  AA      AA      TT
DD      DD  OO      OO  DD      DD              EE          RR      RR  AA      AA  PP      PP  AA      AA      TT
DD      DD  OO      OO  DD      DD              EE          RR      RR  AA      AA  PP      PP  AA      AA      TT
DD      DD  OO      OO  DD      DD              EEEEEEEE    RRRRRRRR    AA      AA  PPPPPPPP    AA      AA      TT
DD      DD  OO      OO  DD      DD              EEEEEEEE    RRRRRRRR    AA      AA  PPPPPPPP    AA      AA      TT
DD      DD  OO      OO  DD      DD              EE          RR  RR      AAAAAAAAAA  PP          AAAAAAAAAA      TT
DD      DD  OO      OO  DD      DD              EE          RR  RR      AAAAAAAAAA  PP          AAAAAAAAAA      TT
DD      DD  OO      OO  DD      DD              EE          RR    RR    AA      AA  PP          AA      AA      TT          ....
DD      DD  OO      OO  DD      DD              EE          RR    RR    AA      AA  PP          AA      AA      TT          ....
DDDDDDDD      OOOOOO    DDDDDDDD    ----------  EEEEEEEEEE  RR      RR  AA      AA  PP          AA      AA      TT          ....
DDDDDDDD      OOOOOO    DDDDDDDD    ----------  EEEEEEEEEE  RR      RR  AA      AA  PP          AA      AA      TT          ....

MM      MM    AAAAAA    RRRRRRRR
MM      MM    AAAAAA    RRRRRRRR
MMMM  MMMM  AA      AA  RR      RR
MMMM  MMMM  AA      AA  RR      RR
MM  MM  MM  AA      AA  RR      RR
MM  MM  MM  AA      AA  RR      RR
MM      MM  AA      AA  RRRRRRRR
MM      MM  AA      AA  RRRRRRRR
MM      MM  AAAAAAAAAA  RR  RR
MM      MM  AAAAAAAAAA  RR  RR
MM      MM  AA      AA  RR    RR
MM      MM  AA      AA  RR    RR
MM      MM  AA      AA  RR      RR
MM      MM  AA      AA  RR      RR
```

```
DRM
;++
; F
; C
; I
; O
;--
; D
10$
```

```
        .title  DOD_ERAPAT - Generate DoD security erase patterns
        .ident  'V04-000'

;
;****************************************************************************
;*                                                                          *
;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;*  ALL RIGHTS RESERVED.                                                    *
;*                                                                          *
;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;*  TRANSFERRED.                                                            *
;*                                                                          *
;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;*  CORPORATION.                                                            *
;*                                                                          *
;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;*                                                                          *
;*                                                                          *
;****************************************************************************
;

;++
; Facility:
;
;       VMS Executive
;
; Abstract:
;
;       This routine generates security erase  patterns which are used by user
;       written programs to preclude the unauthorized disclosure of classified
;       information.
;
; Envrionment:
;
;       VAX/VMS, Kernel Mode
;
; Author:
;
;       Michael T. Rhodes,      Creation Date:  October, 1982
;
; Modified By:
;
;       V03-001 JRL0023         John R. Lawson, Jr.     10-Jul-1984 14:23
;               Add interface to the system.
;
;--

        .page
```

```
DRM

;++
; F
;
;
; C
;
;
; I
;
;
; O
;
;--
```

```
        .sbttl  Declarations

        $ERADEF                                 ; Define function codes
        $SSDEF                                  ; Define status codes

;
; Equated symbols:
;

        TYPE    = 4                             ; Offset to TYPE parameter (value)
        COUNT   = 8                             ; Offset to COUNT parameter (value)
        PATADR  = 12                            ; Offset to PATADR parameter (address)

;
; Assumptions:
;

        ASSUME  ERA$K_MINTYPE EQ 1
        ASSUME  ERA$K_MAXTYPE EQ 3

        ASSUME  ERA$K_MEMORY EQ 1
        ASSUME  ERA$K_DISK EQ 2
        ASSUME  ERA$K_TAPE EQ 3

        .page
        .sbttl  Loadable image header and trailer
;++
;
; Loader Information:
;
;       At boot time,  SYSBOOT.EXE  checks  the SYSGEN parameter LOADERAPT
;       (SGN$V_LOADERAPAT);  if it  is  set,  this  image gets loaded from
;       SYS$SYSTEM:ERAPATLOA.EXE.  There must exist, in the image, certain
;       information for the loader; these two PSECT's supply that info.
;
; Linking this Object:
;
;       $ link/notraceback/system=0/header/executable=SYS$SYSTEM:ERAPATLOA -
;               DOD_ERAPAT, SYS$SYSTEM:SYS.STB/selective_search
;
;       The /SYSTEM qualifier guaratees  that  the PSECT's will be ordered
;       alphabetically within the image,  forcing $$$$$$$$ to be first and
;       ________ to be last.
;
;--

;
; This table must appear at the beginning of the image
;

        .psect  $$$$$$$$        page, pic       ; In a system image, the PSECT's
                                                ; are ordered alphabetically
        PRMSW = 1                               ; Flag to indicate loadable code

;
```

```
; This table directs SYSBOOT.EXE for loading the rest of this image
;

        SLVTAB  subtype=DYN$C_NON_PAGED,-          ; Non-paged pool
                end=DOD_ERAPAT$END,-               ; Computed size of image
                sysvecs=VECTORS$,-                 ; Vector into the routine
                prot_w=PRT$C_URKW,-                ; Page protection
                facility=<DoD Security Erase>      ; What is this?

;
; These vectors replace the default ones in SYS.EXE
;

VECTORS$:                                          ; Vector table

        LOADVEC EXE$ERAPAT_VEC,5,,DOD_ERAPAT$

        .long   -1                                 ; Terminated by -1

;
; This label must appear at the end of the image
;

        .psect  ________          byte, pic

DOD_ERAPAT$END::

        .page
        .sbttl  $ERAPAT System Service

;++
; $ERAPAT
;
; Functional Description:
;
;       To preclude the unauthorized disclosure of classified information,
;       the caller iteratively invokes  the  $ERAPAT system service.  Upon
;       each invocation, the user  increments  the iteration count and the
;       service returns  an  erasure  pattern  plus  either  SS$_NORMAL or
;       SS$_NOTRAN  (which  indicates  the  declassification  procedure is
;       complete).
;
; Calling sequence:
;
;       This routine should be called via a CALLS/G to EXE$ERAPAT.
;
; Input:
;
;       TYPE(AP)          Security erase type.  The legal types are
;
;                               1. ERA$K_MEMORY : main memory
;                                                  (volatile r/w semiconductor)
;                               2. ERA$K_DISK   : disk storage
;                               3. ERA$K_TAPE   : tape storage
;
;       COUNT(AP)         Iteration count.  The  service  should  be called
;                       the first time  with  the  value 1, then 2, etc.,
;                       until the  status  SS$_NOTRAN  is  returned.  The
;                       local symbol MAXCOUNT defines how many times this
;                       happens.
;
; Output:
;
;       PATADR(AP)      Address of a longword into which the security
;                       erase pattern is to be written.
;
; Routine value:
;
;       R0 = SS$_ACCVIO         Pattern output area not accessible
;            SS$_BADPARAM       Invalid security type code
;            SS$_NORMAL         Normal successful completion
;            SS$_NOTRAN         Security erase complete
;
;--

        .page
        .sbttl  Data necessary for routine

        .psect  $DATA$  long, pic

;
; Own Storage:
;

COUNTS$:

        .long   1                       ; Main Memory iteration count
        .long   3                       ; Disk Storage iteration count
        .long   2                       ; Tape Storage iteration count

PATTERNS$:                              ; Storage type erasure patterns

        .long   0                       ; Main memory erase pattern
        .long   -1                      ; Disk Storage erase pattern
        .long   ^XDB6DB6DB              ; Tape Storage erase pattern

        .page
        .sbttl  Routine to generate the erase patterns

        .psect  $CODE$  long, pic

;
; Routine to return erase patterns:
;

DOD_ERAPAT$:                            ; $ERAPAT entry point

        pushr   ^M<r1>                  ; Save registers

;
; Check the values of the parameters ...
```

DRS

$D
$D
$D
$D
$D
$D
$D

```
;
        movzwl  #SS$_BADPARAM, r0       ; Assume bad parameters

        cmpl    COUNT(ap), #0           ; Is count too small?
        bleq    EXIT                    ; Branch if yes

        cmpl    TYPE(ap), #ERA$K_MINTYPE        ; Type code too small?
        blss    EXIT                            ; Branch if yes
        cmpl    TYPE(ap), #ERA$K_MAXTYPE        ; Type code too large?
        bgtr    EXIT                            ; Branch if yes

;
; Use the TYPE as an index into COUNTS$ and PATTERNS$
;

        subl3   #1, TYPE(ap), r1        ; Vectors begin with 0

;
; Signal completion by returning SS$_NOTRAN ...
;

        movzwl  #SS$_NOTRAN, r0         ; Set completion status
        cmpl    COUNT(ap), COUNTS$[r1]  ; Are we done?
        bgtr    EXIT                    ; Yes, return completion status

;
; Is the return address for the pattern writable ???
;

        movzwl  #SS$_ACCVIO, r0         ; Assume access violation
        IFNOWRT #4, @PATADR(ap), EXIT   ; Branch if no write access

;
; Look up the appropriate erase pattern ...
;

        movzwl  #SS$_NORMAL, r0                 ; Assume success at this point
        movl    PATTERNS$[r1], @PATADR(ap)      ; Send back the pattern


;
; That's all folks ...
;
EXIT:   popr    ^M<r1>                  ; Restore registers
        ret                             ; Return

        .END
```

```
DRS

;
; E
;

;
; O
;


SLV

SLV
```

0157 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LPMULT
B32

DRMAST
MAR

ADDRIVER
MAR

TDRIVER
MAR

USSTEST
MAR

GBLSECUFO
MAR

USSDISP
MAR

DOD_ERAPAT
MAR

LBRMAC
MAR

XADRIVER
MAR

LABIOCIN
MAR

DRSLV
MAR

DTE_DF03
MAR

WORKQ
LIS

SCRFT
MAR

EXAMPLES