My Super Sweet 16-Bit Malware
~* MS-DOS *~ Edition
[TSR Remix]

Nika Korchok Wakulich
ic3qu33n

whoami

Twitter: @nikaroxanne
Discord: @ic3qu33n
Mastodon: ic3qu33n@infosec.exchange
Website: https://ic3qu33n.fyi/
GitHub: @nikaroxanne

Security Consultant at Leviathan Security Group
Reverse engineer + artist
I <3 malware, hardware hacking, firmware hacking, skateboarding, learning languages, creating art, writing lil assembly programs, etc.

greetz 2 the following for their assistance/support w this talk:
@0daySimpson, Ben Mason (@suidroot), @Laughing_Mantis,
Richard Johnson (@richinseattle), the Rootsyn Discord (@qkumba, @phlaul and @barbie),
The team at Leviathan
BSidesSF

leviathan security group

```
C:\>_
```

DISCLAIMER:

The views expressed in this presentation are my own and do not reflect the opinions of my past, present or future employers

Viewer Discretion is advised.

What this talk is

[and what this talk is not]

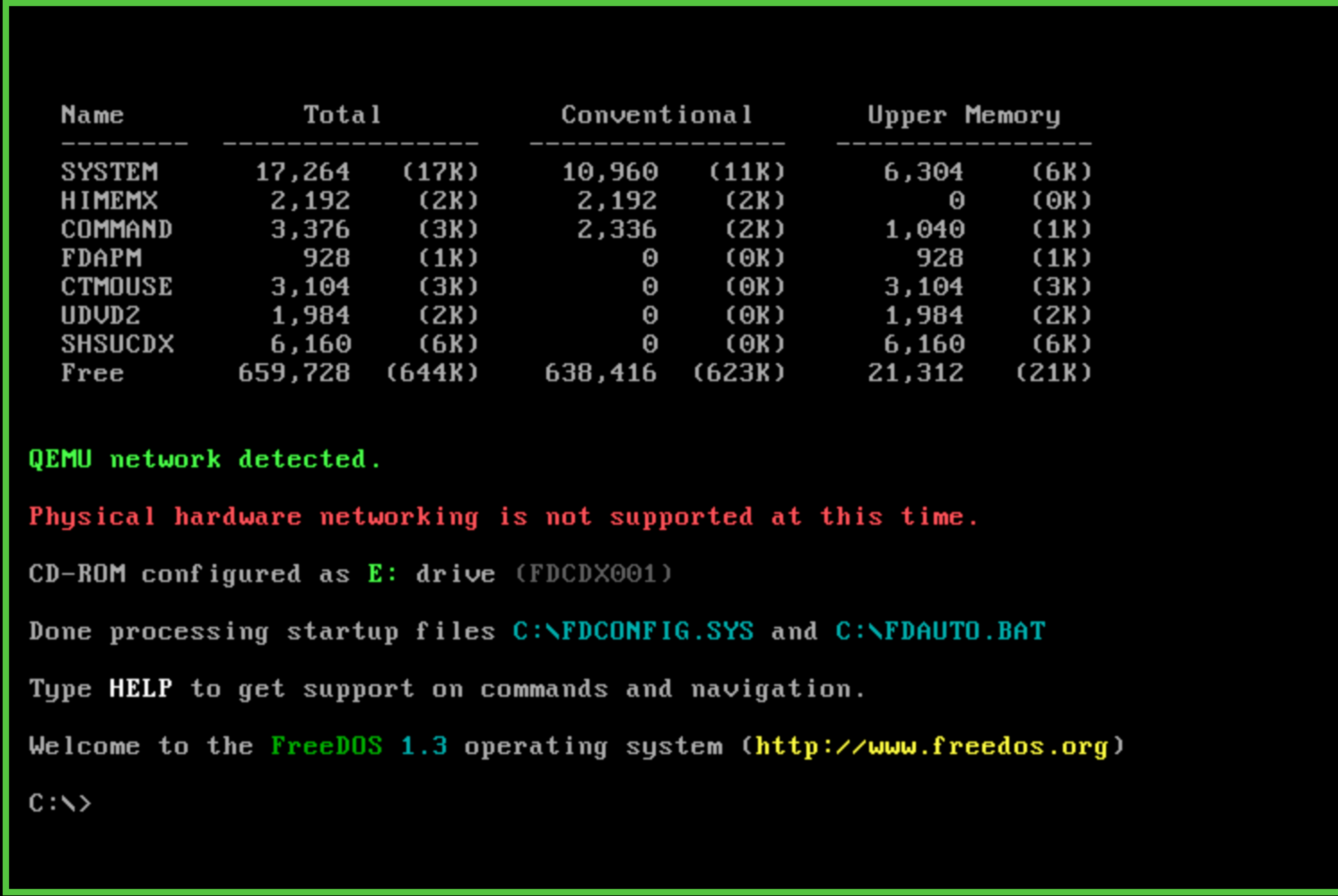
- An introduction to MS-DOS era malware, including an overview of the MS-DOS architecture and unique threat landscape
- A starting point for learning about virus techniques of sophisticated malware of the 1980s and 1990s
- ***Not:*** *A complete and thorough examination of every piece of DOS-era malware; nor is it an in-depth analysis of malware targeting other OSes of that era (Elk Cloner isn't relevant here, though its RAM infection techniques are, to my heart, very dear (deer) (I'll stop))*

Mars Land, by Spanska

(coding a virus can be creative)

Overview

- Introduction
- Motivations [Why would you want to study malware for an EOL OS?]
- Definitions of terms
- An overview of MS-DOS architecture
- Notable interrupts of MS-DOS malware
- TSR Overview
- Overview of stealth/persistence techniques
- Analysis of notable malware samples
- Connections to modern malware
- Additional resources for continued learning
- Q+A



```
Name           Total                Conventional          Upper Memory
--------  ----------------     ----------------     ----------------
SYSTEM      17,264    (17K)      10,960    (11K)       6,304     (6K)
HIMEMX       2,192     (2K)       2,192     (2K)           0     (0K)
COMMAND      3,376     (3K)       2,336     (2K)       1,040     (1K)
FDAPM          928     (1K)           0     (0K)         928     (1K)
CTMOUSE      3,104     (3K)           0     (0K)       3,104     (3K)
UDVD2        1,984     (2K)           0     (0K)       1,984     (2K)
SHSUCDX      6,160     (6K)           0     (0K)       6,160     (6K)
Free       659,728   (644K)     638,416   (623K)      21,312    (21K)

QEMU network detected.

Physical hardware networking is not supported at this time.

CD-ROM configured as E: drive (FDCDX001)

Done processing startup files C:\FDCONFIG.SYS and C:\FDAUTO.BAT

Type HELP to get support on commands and navigation.

Welcome to the FreeDOS 1.3 operating system (http://www.freedos.org)

C:\>
```

Motivations

Choose your own adventure

A. Sharpen your reverse engineering skills - these bins are TINY and packed with puzzles

B. become a demoscene icon (I’d rather code in x86 than Rust and I’m not sorry)

C. It looks pretty …why does it look pretty? Why is it infecting the MBR? Is this malware or is this art? is it both??

D. Reversing 16-bit malware has fun side-quests for everyone:

    A. Hardware hacking

    B. BIOS bb

    C. Graphics programming

    D. Polymorphism

    E. OS development

    F. Binary golf

Definitions

- Virus:
  Fred Cohen (credited as being the “creator” of the term “computer virus” as a way to describe a self-reproducing program, which he used in his 1984 paper “Computer Viruses, Theory and Experiments.”
  Cohen’s definition was thus:

  We define a computer 'virus' as a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself. With the infection property, a virus can spread throughout a computer system or network using the authorizations of every user using it to infect their programs. Every program that gets infected may also act as a virus and thus the infection grows. — Fred Cohen, “Computer Viruses, Theory and Experiments,” 1984

- **Virus** = a self-replicating program that uses a host program to produce those new copies

Definitions

- **Polymorphic virus = a virus that uses a *variable* encryption/decryption routine and a variable key to create an encrypted copy of itself in memory, which is appended to/inserted into a host file [1]**
    - **The encrypted image of the virus payload (and the encryption routine of the virus itself) changes with each iteration, so as to avoid/minimize the presence of known byte patterns used in AV signatures**
- **Bootkit = A bootkit is a type of malware that infects a critical component of the OS boot process to install itself and maintain persistence.**
- **Boot sector infector = the earliest form of bootkit; a BSI is a bootkit that targets storage media that did not have an MBR (Master Boot Record), and only had a boot sector (hence the name! Surprise!)**
    - **BSI's targeted various forms of floppy diskettes, which did not use an MBR**

[1] Page 318-322 "The Giant Black Book of Computer Viruses. Chapter 27: Polymorphic Viruses" Mark Ludwig, 2nd ed., American Eagle Books, 1998.

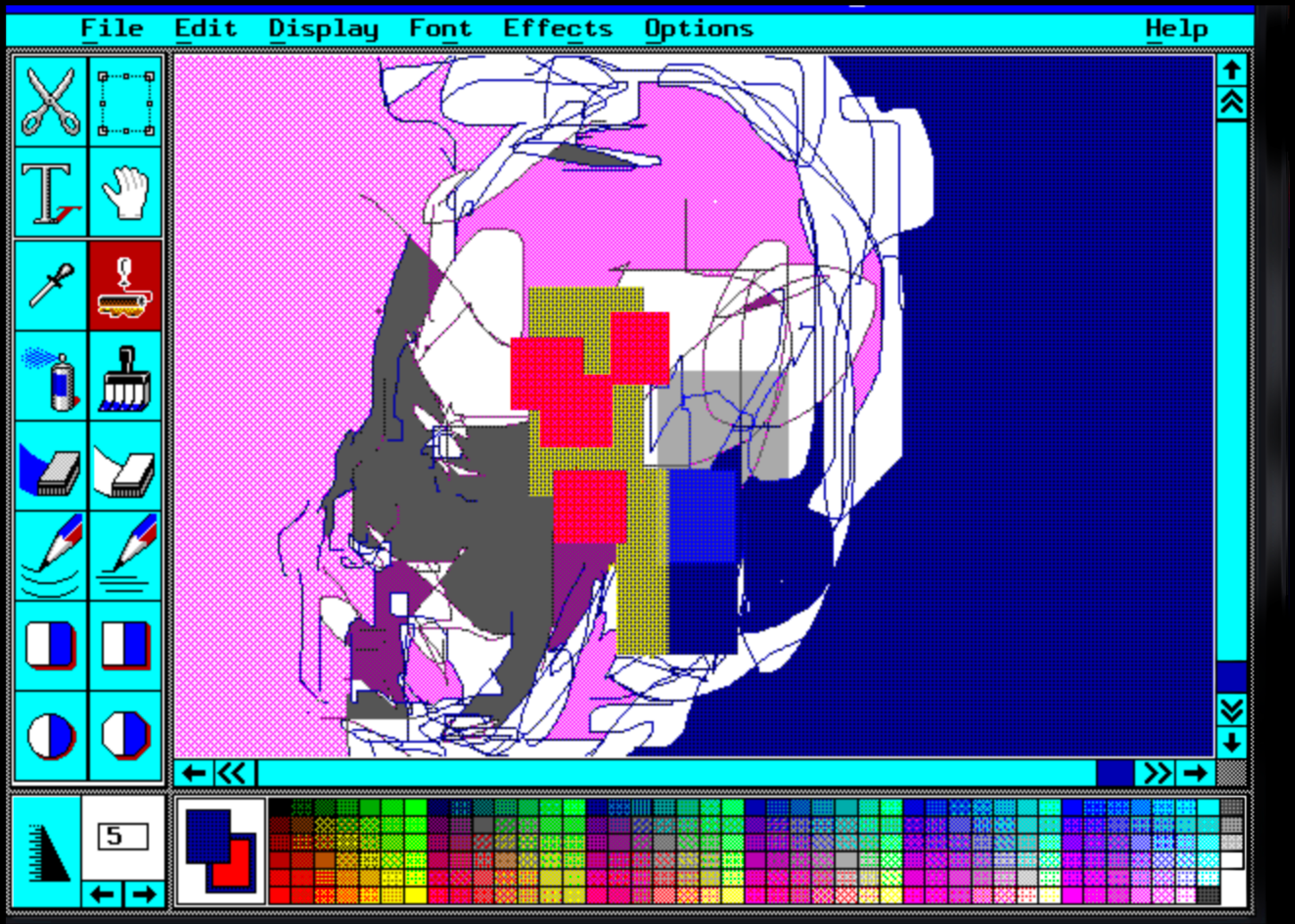
```
My Super Sweet 16-Bit Malware:────────
$MS-DOS Editionddddddddddddddddddddddddd
MS-DOS Edition°" ◆•♀▶¶↑∟ $(,048<@DHLPTX
 ◆•♀▶¶↑∟ $(,048<@DHLPTX\`dhlptx¦ÇäêîÉöÿ
```

A Whirlwind Tour of MS-DOS

The DOS Kernel

- OS v1.0 debuted in 1981, v6.22 1994
- Some features of MS-DOS include:
  - MS-DOS operates in 16-bit real mode
  - Provides device-independent device access to computer resources, using the key programming interface of MS-DOS: ***system functions***
  - Single-task operating system [only one program runs at a time]**
    **TSRs are a partial workaround to the limitations of a single-task OS

“Microsoft MS-DOS Programmer’s Reference,” Microsoft Corporation, 2nd ed.: version 6.0., Microsoft Press, 1993

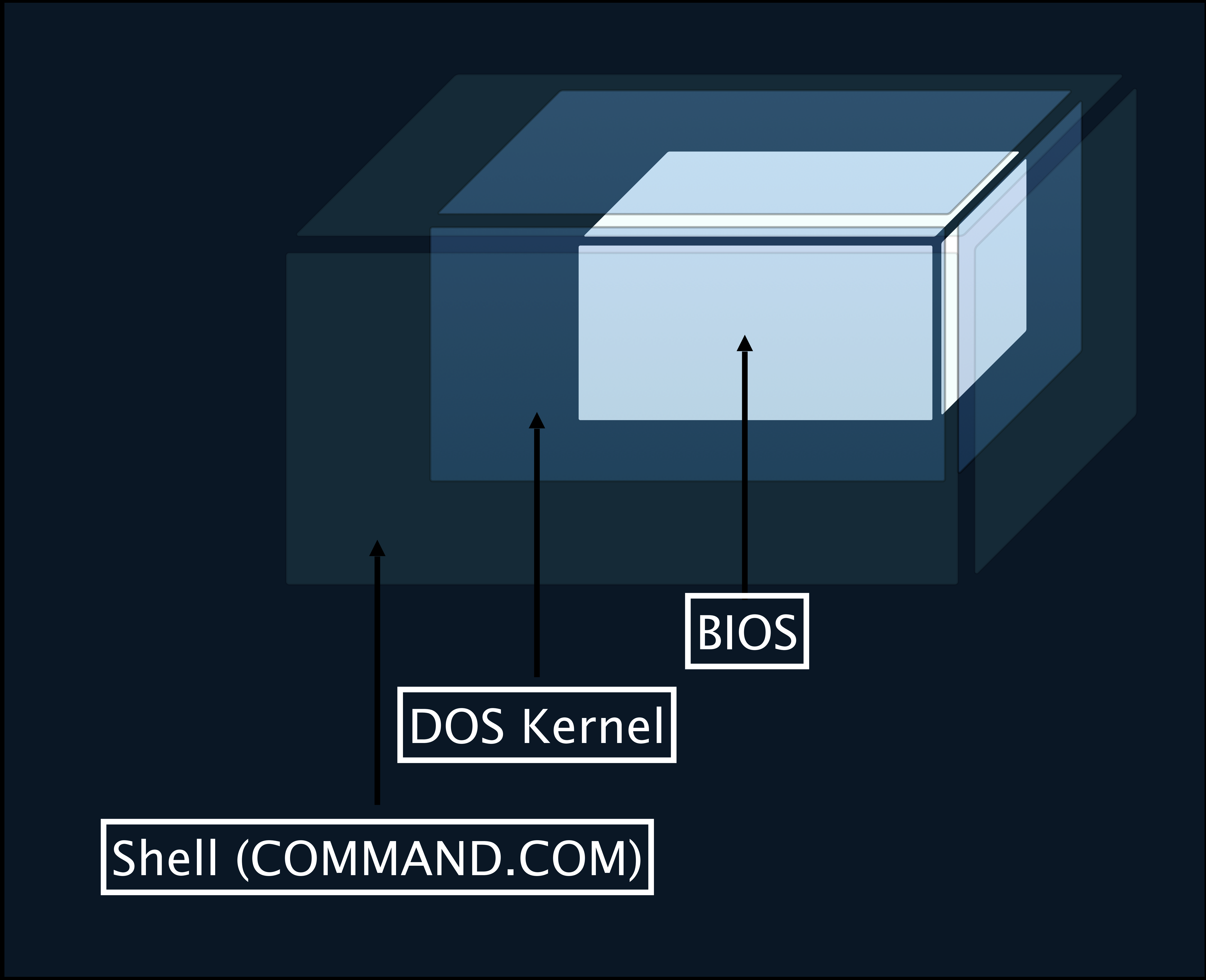


MS Paintbrush, you don’t look a day over 1989 honey xoxo

The DOS Kernel

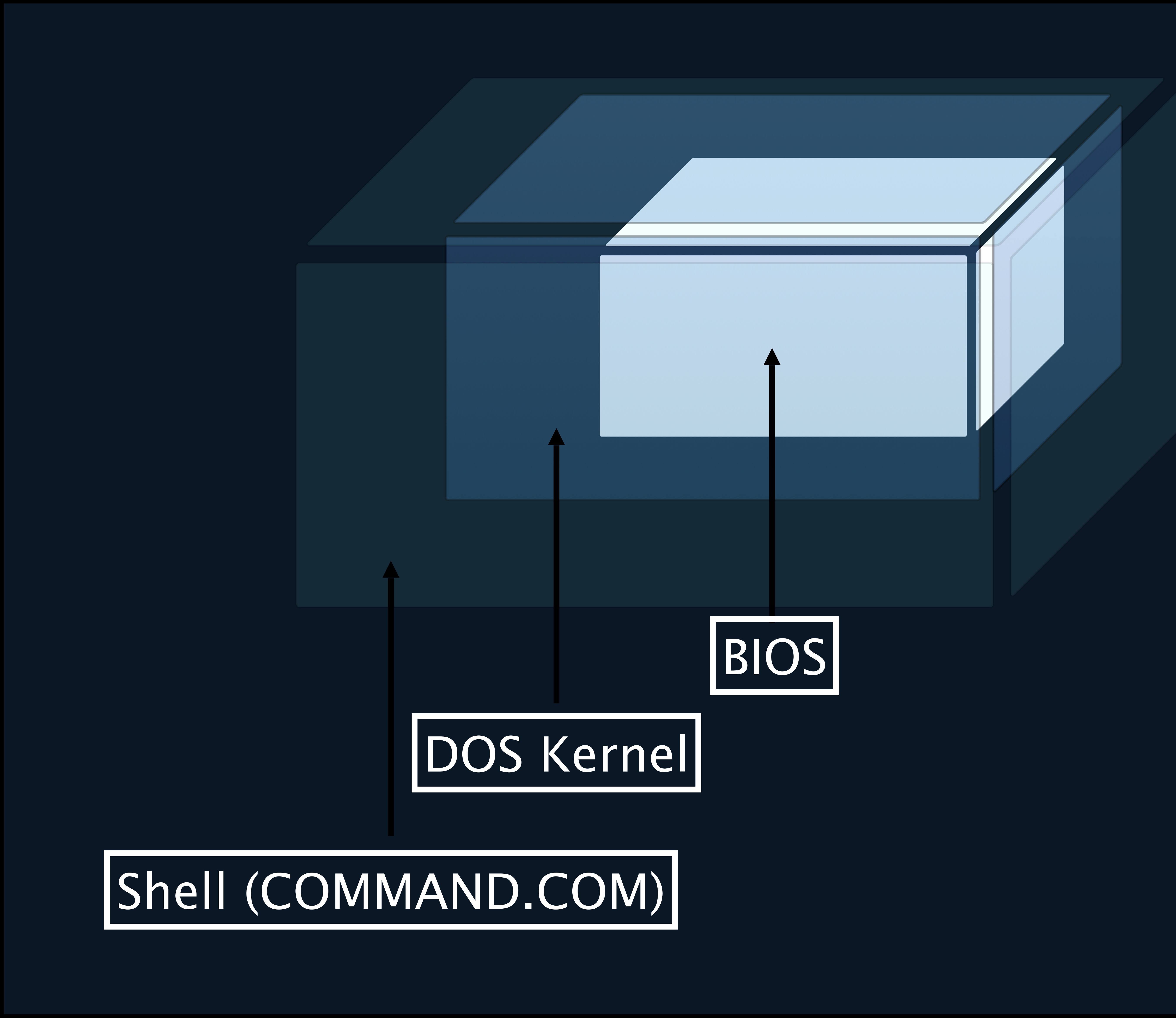
- The MS-DOS operating system is divided into roughly three layers:
- 1. The BIOS (Basic Input/Output System)
- 2. The DOS Kernel
- 3. The command processor (shell) — COMMAND.COM

“Advanced MS-DOS Programming: Section 1 – Programming for MS-DOS,” Ray Duncan, Microsoft Press, 1986



The DOS Kernel

- The DOS kernel provides ***system functions*** that allow a user to perform actions with a provided collection of hardware-independent services
- These system functions include:
  - Memory management
  - Spawning programs
  - Character device I/O
  - File management
- Programs in MS-DOS interact with these system functions by loading registers with function-specific values + transferring control using ***software interrupts***



“Advanced MS-DOS Programming: Section 1 – Programming for MS-DOS,” Ray Duncan, Microsoft Press, 1986

```
My Super Sweet 16-Bit Malware:────────
$MS-DOS Editionddddddddddddddddddddddddd
MS-DOS Edition°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX
 ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx¦ÇäêîÉöÿ
```

A Whirlwind Tour of MS-DOS: Notable Interrupts for Malware

Notable Interrupts for MS-DOS Malware

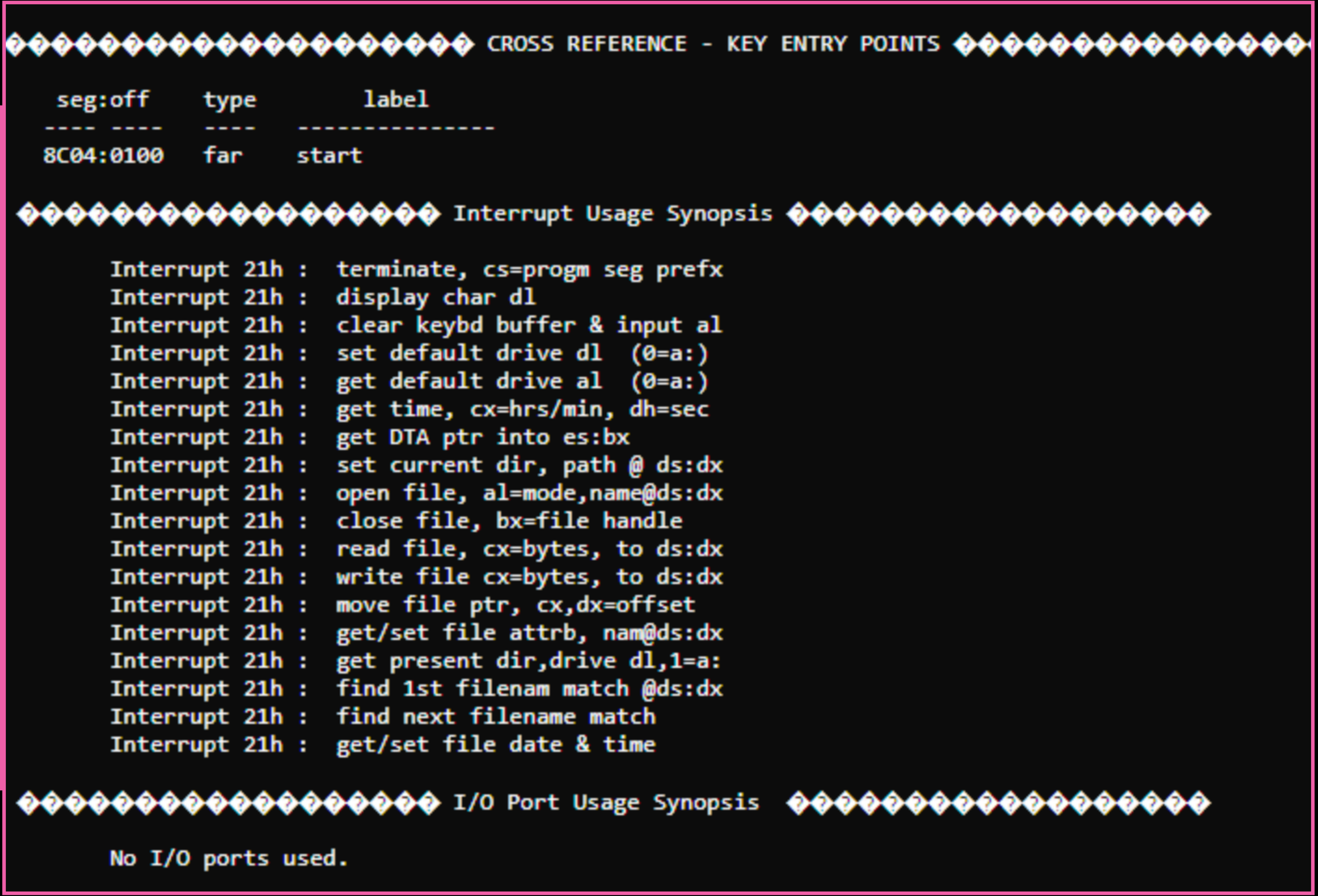
- System Interrupts (ROM BIOS):
  - Int 10h: Video services
  - ***Int 13h: Disk services***
  - Int 16h: Keyboard services
- MS-DOS Interrupts:
  - ***Int 21h - MS-DOS System Functions***
  - Int 25h - Absolute Disk Read
  - Int 26h - Absolute Disk Write

ROM Bios Interrupts are 05h, and 10h-1Fh

MS-DOS Reserved Interrupts: 20h-3Fh

Notable Interrupts for MS-DOS Malware

- RTFMSDOSS (RTF MS-DOS Source)
- Because it’s usually beautifully + succinctly documented by the virus authors themselves
- Thanks x a mill Ralf Burger (this is a beautiful asm file)

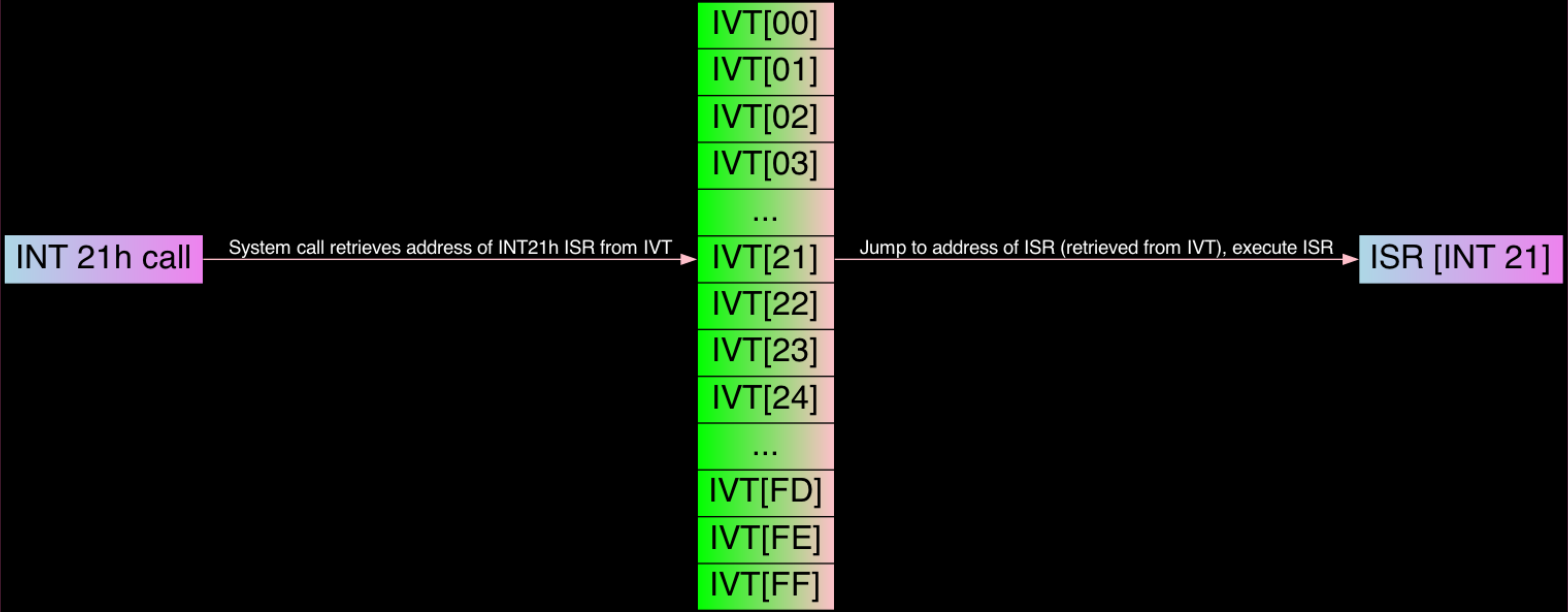


```
���������������������� CROSS REFERENCE - KEY ENTRY POINTS ����������������

  seg:off    type       label
 ---- ----   ----   ---------------
 8C04:0100   far    start

�������������������� Interrupt Usage Synopsis ��������������������

      Interrupt 21h :  terminate, cs=progm seg prefx
      Interrupt 21h :  display char dl
      Interrupt 21h :  clear keybd buffer & input al
      Interrupt 21h :  set default drive dl  (0=a:)
      Interrupt 21h :  get default drive al  (0=a:)
      Interrupt 21h :  get time, cx=hrs/min, dh=sec
      Interrupt 21h :  get DTA ptr into es:bx
      Interrupt 21h :  set current dir, path @ ds:dx
      Interrupt 21h :  open file, al=mode,name@ds:dx
      Interrupt 21h :  close file, bx=file handle
      Interrupt 21h :  read file, cx=bytes, to ds:dx
      Interrupt 21h :  write file cx=bytes, to ds:dx
      Interrupt 21h :  move file ptr, cx,dx=offset
      Interrupt 21h :  get/set file attrb, nam@ds:dx
      Interrupt 21h :  get present dir,drive dl,1=a:
      Interrupt 21h :  find 1st filenam match @ds:dx
      Interrupt 21h :  find next filename match
      Interrupt 21h :  get/set file date & time

�������������������� I/O Port Usage Synopsis  ��������������������

      No I/O ports used.
```

“Virdem” by R. Burger, 1989

Interrupt Vector Table

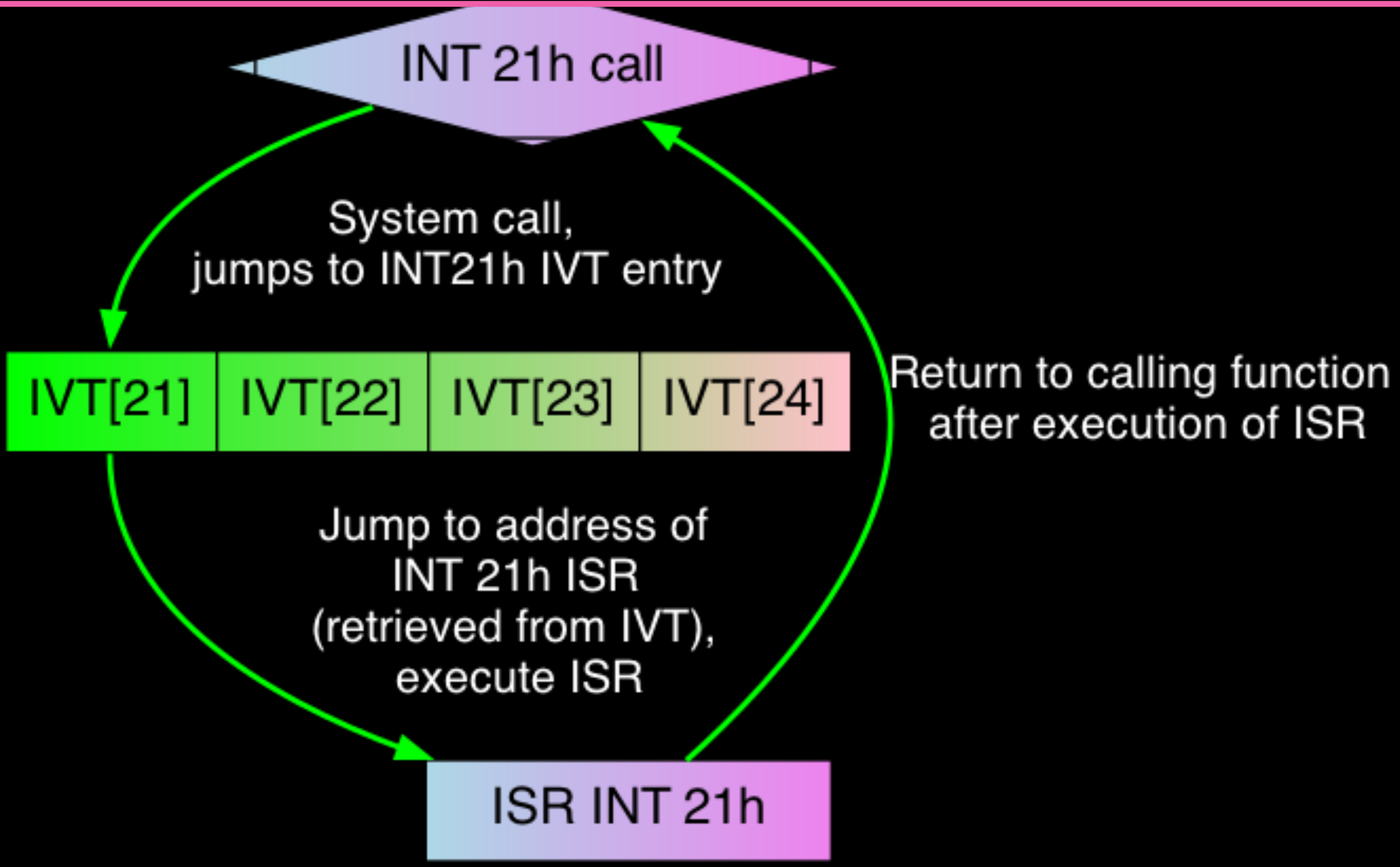
Invoking system calls on MS-DOS


INT 21h call
System call retrieves address of INT21h ISR from IVT
IVT[00]
IVT[01]
IVT[02]
IVT[03]
...
IVT[21]
IVT[22]
IVT[23]
IVT[24]
...
IVT[FD]
IVT[FE]
IVT[FF]
Jump to address of ISR (retrieved from IVT), execute ISR
ISR [INT 21]

Interrupt Vector Table and Interrupt Service Routines

Interrupt Vector Table

Invoking system calls on MS-DOS



Typical code flow of executing an Interrupt Service Routine on MS-DOS by invoking a system call

Terminate and Stay Resident Programs [TSRs]

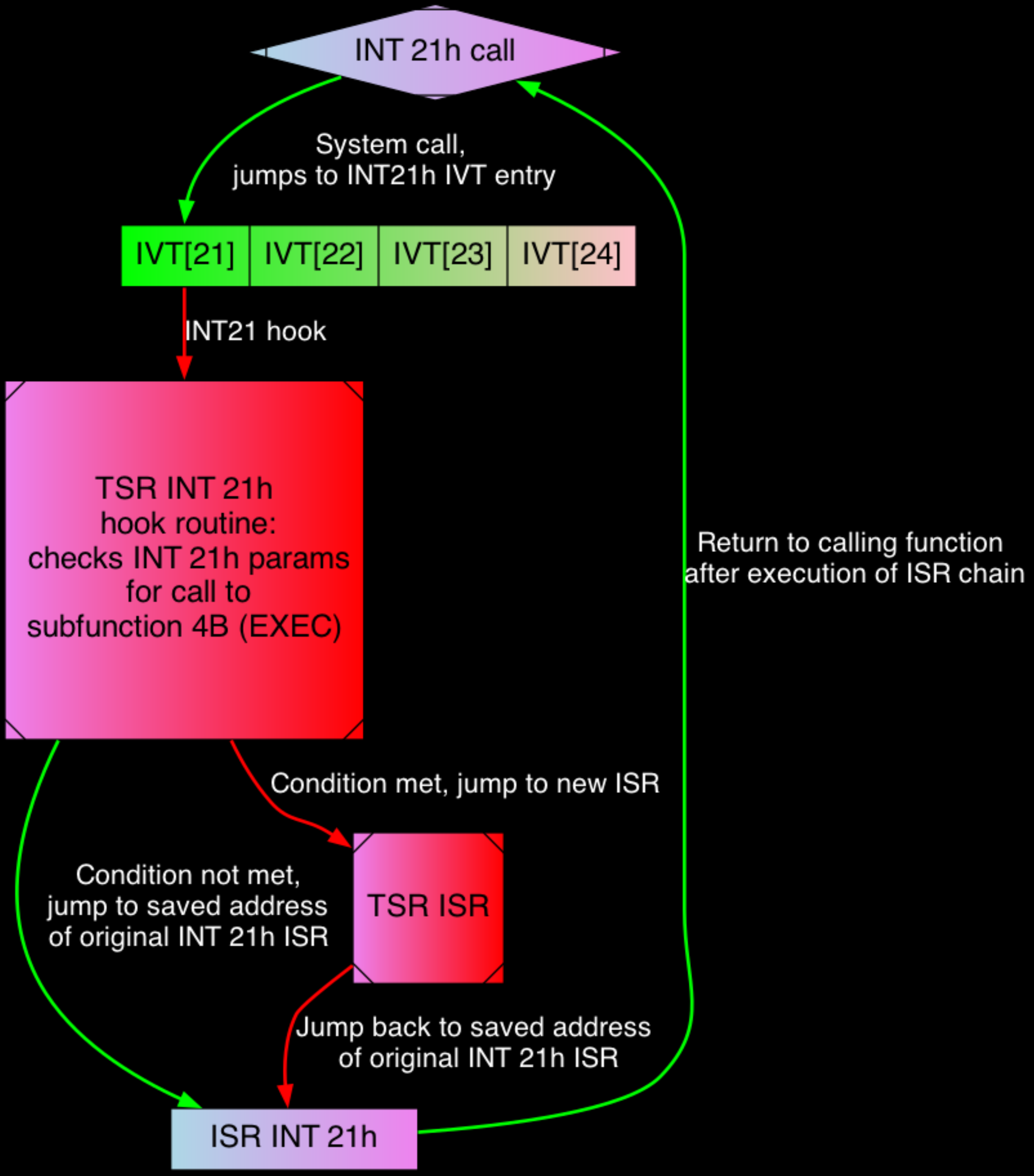
- **TSR = a feature of MS-DOS that allows a user to bypass the limitations of a single-task OS by installing a persistent program in RAM, which would be invoked by subsequent interrupts**
- In order to install a TSR, one had to modify several components of the Interrupt Vector Table, which was the precursor to the Interrupt Descriptor Table, and that defined the addresses of all of the 256 interrupts in 8086 real-mode.
- The basic formula went as follows:

1. Find the address of a desired interrupt in the IVT
2. Retrieve both address components of the target interrupt ("address components" = the original segment and the original offset, because DOS used a segmented addressing scheme)
3. The original interrupt's address components (segment and offset) are saved to a specific address (i.e. two variables in the data segment or to some other location in memory, defined by the virus writer)
4. A new interrupt handler is installed in the IVT
5. That new interrupt handler's interrupt routine concludes by jumping back to the original address and passing control back, creating the illusion that the original interrupt has proceeded as per usual

More detailed walk-throughs of TSR techniques are available on my website:
https://ic3qu33n.fyi/projects/16bitm4lw4r3-MSDOS/TerminateStayResidentPrograms-part1
https://ic3qu33n.fyi/projects/16bitm4lw4r3-MSDOS/TerminateStayResidentPrograms-part2

Interrupt Vector Table

Hooking system calls on MS-DOS



Modifying control flow of Interrupt Service Routines by hooking the IVT with a TSR

Terminate and Stay Resident Programs [TSRs] - Demo

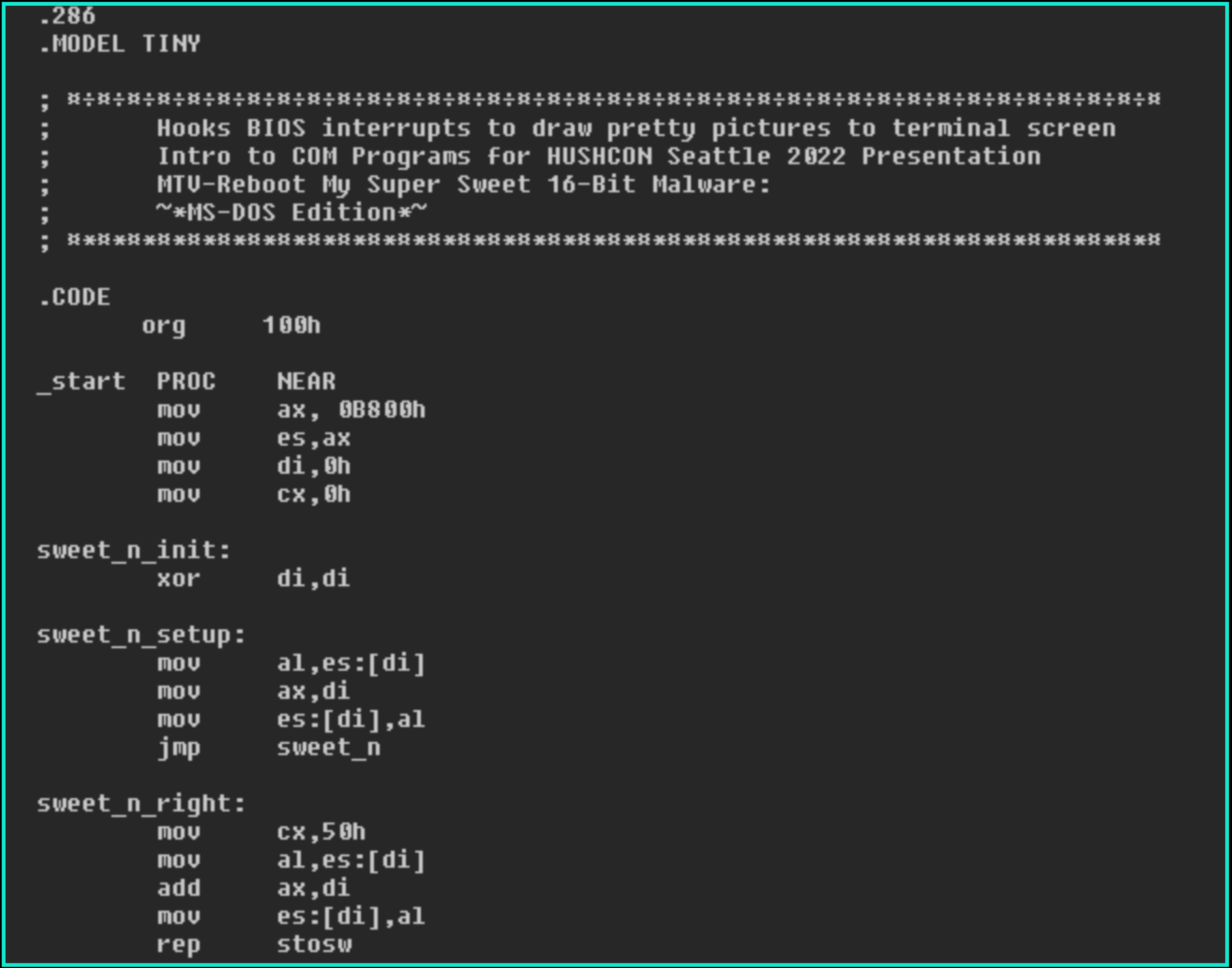
```
C:\>
```

```
My Super Sweet 16-Bit Malware:────────
$MS-DOS Editionddddddddddddddddddddddddd
MS-DOS Edition°ⁿ ♦•♀►¶↑∟ $(,048<@DHLPTX
 ♦•♀►¶↑∟ $(,048<@DHLPTX\`dhlptx¦Çäêîéöÿ
```

A Whirlwind Tour of MS-DOS COM Programs

COM Programs

- .COM programs fit the TINY memory model of Intel 8086 ISA
- Must always have an origin of 100h [This is the length of the Program Segment Prefix or PSP]
- All segment registers contain the same value — code and data are mixed together
- No header, no identifying information, no relocation information
- No parents, no rules! (Not quite, but almost)



```
.286
.MODEL TINY

; ¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤÷¤
;       Hooks BIOS interrupts to draw pretty pictures to terminal screen
;       Intro to COM Programs for HUSHCON Seattle 2022 Presentation
;       MTV-Reboot My Super Sweet 16-Bit Malware:
;       ~*MS-DOS Edition*~
; ¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤*¤

.CODE
       org     100h

_start  PROC    NEAR
        mov     ax, 0B800h
        mov     es,ax
        mov     di,0h
        mov     cx,0h

sweet_n_init:
        xor     di,di

sweet_n_setup:
        mov     al,es:[di]
        mov     ax,di
        mov     es:[di],al
        jmp     sweet_n

sweet_n_right:
        mov     cx,50h
        mov     al,es:[di]
        add     ax,di
        mov     es:[di],al
        rep     stosw
```

COM Programs

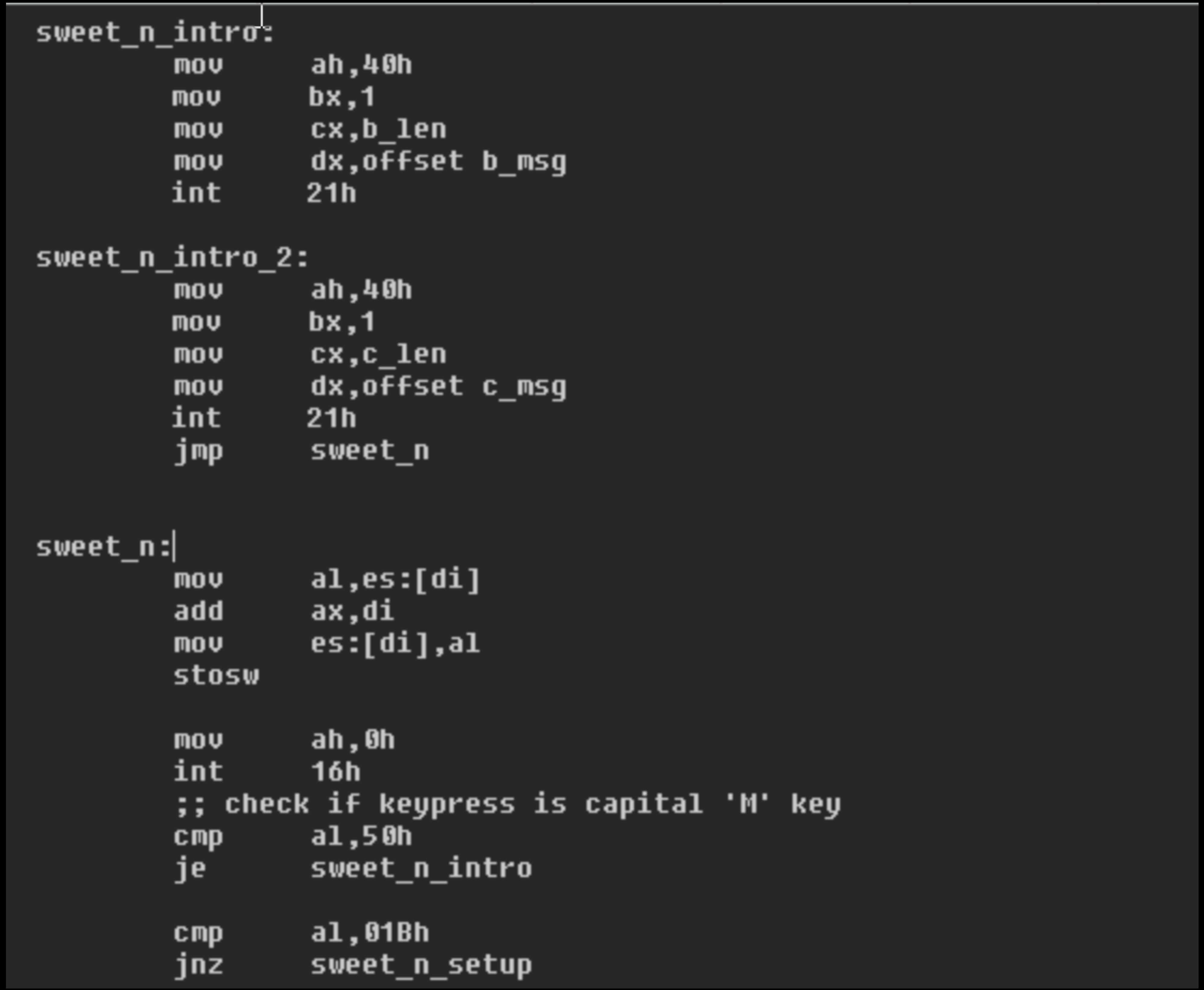
- Max size of .COM program:
  65536 bytes - length PSP (256 bytes) - word of stack (2 bytes) = 65278 bytes (~63kB)

- .COM resides in memory as an absolute memory image

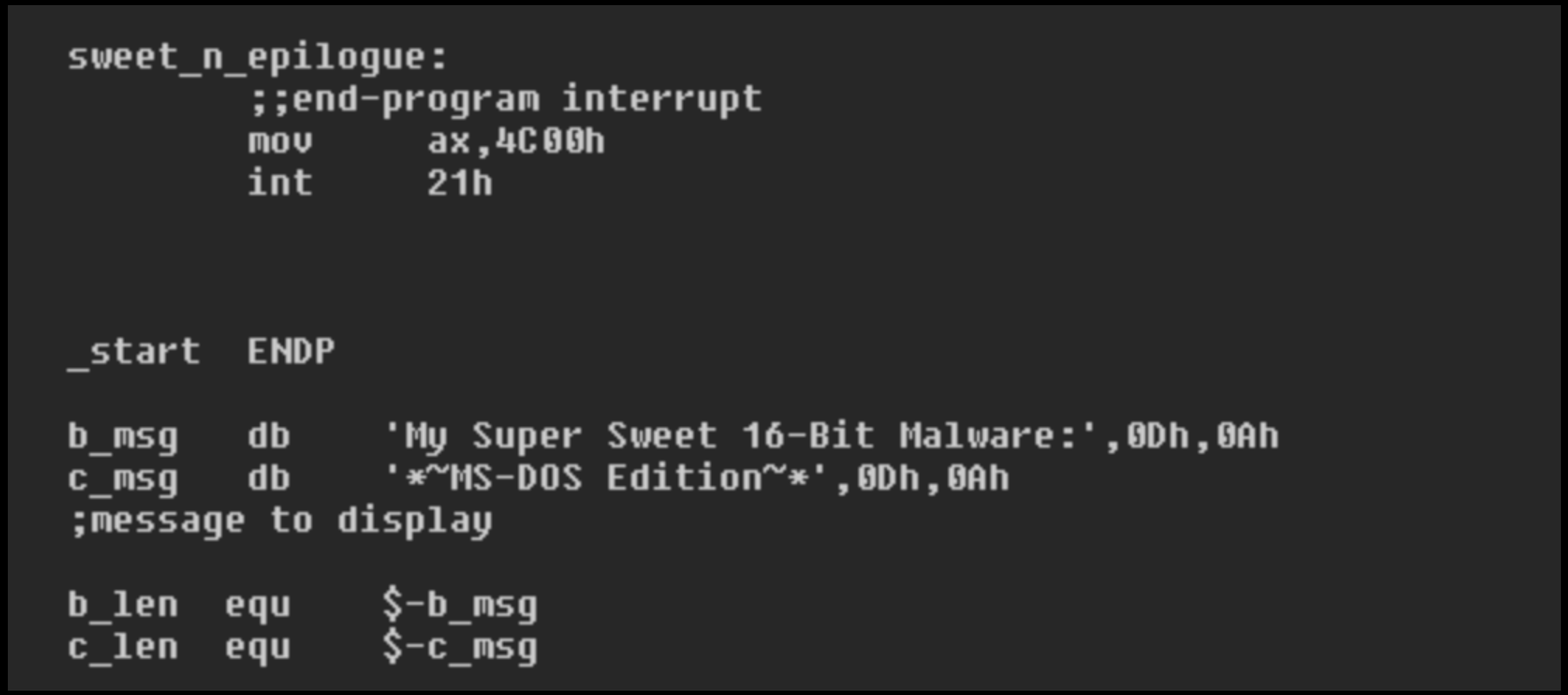
  - resides (is loaded into) a single segment of memory [a segment = 64k]

Uses segmented addressing scheme of 16-bit architecture (again we're running in 16-bit real mode, but accessing addresses in a range of a 20-bit address space)

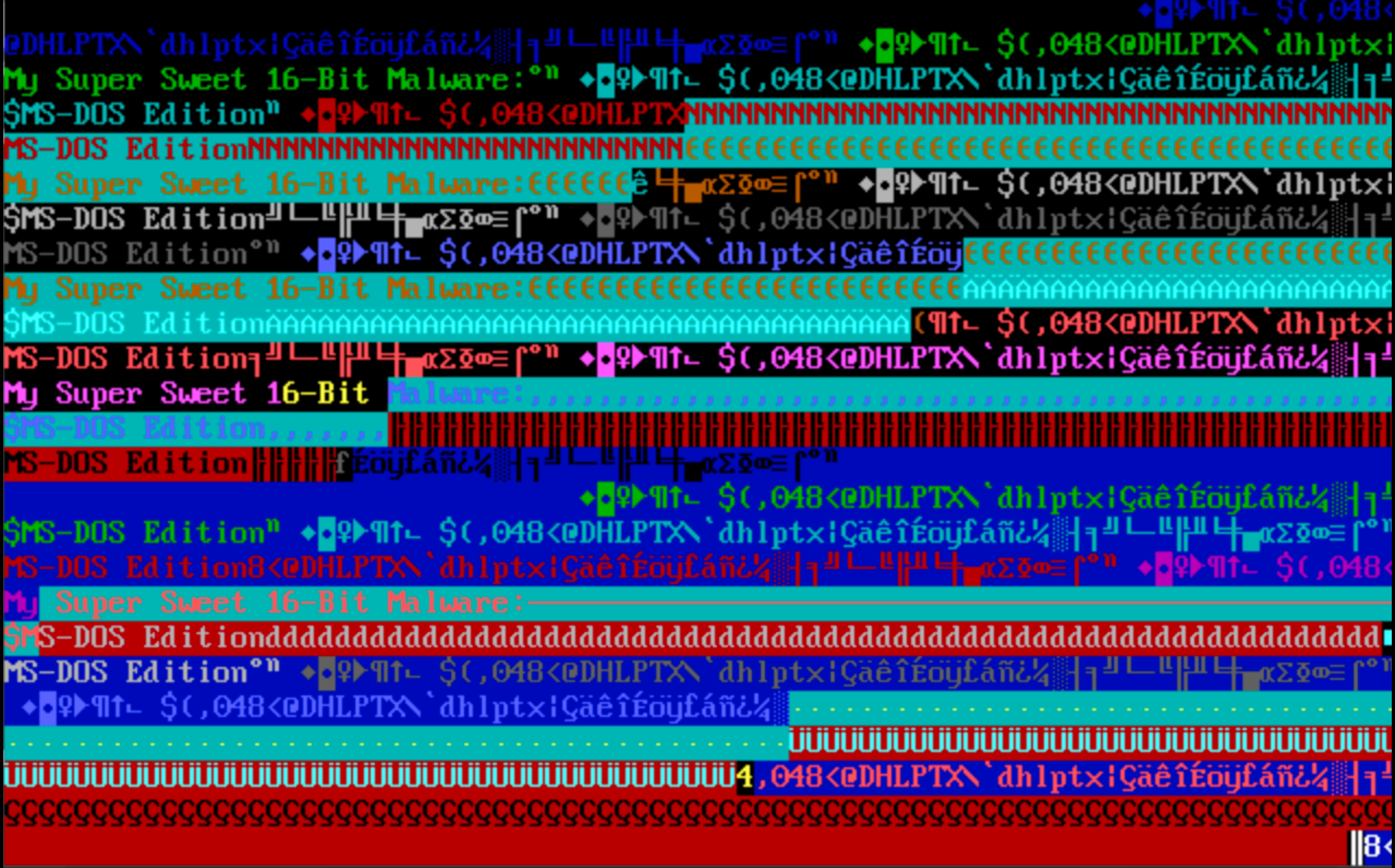
[segment]:[offset]



```
sweet_n_intro:
        mov     ah,40h
        mov     bx,1
        mov     cx,b_len
        mov     dx,offset b_msg
        int     21h

sweet_n_intro_2:
        mov     ah,40h
        mov     bx,1
        mov     cx,c_len
        mov     dx,offset c_msg
        int     21h
        jmp     sweet_n


sweet_n:
        mov     al,es:[di]
        add     ax,di
        mov     es:[di],al
        stosw

        mov     ah,0h
        int     16h
        ;; check if keypress is capital 'M' key
        cmp     al,50h
        je      sweet_n_intro

        cmp     al,01Bh
        jnz     sweet_n_setup
```



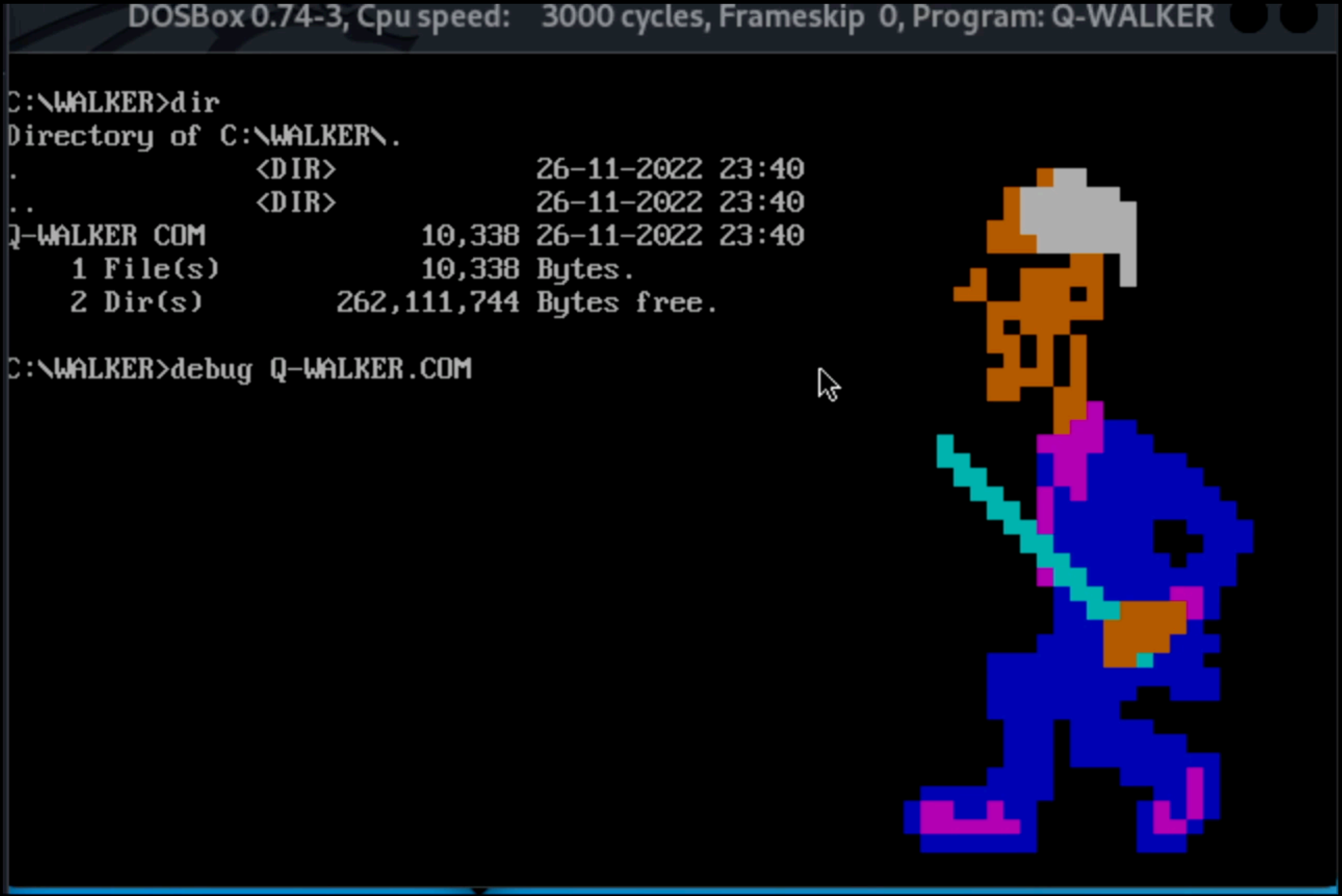
```
sweet_n_epilogue:
        ;;end-program interrupt
        mov     ax,4C00h
        int     21h



_start  ENDP

b_msg   db      'My Super Sweet 16-Bit Malware:',0Dh,0Ah
c_msg   db      '*~MS-DOS Edition~*',0Dh,0Ah
;message to display

b_len   equ     $-b_msg
c_len   equ     $-c_msg
```



```
                                                                ♦◘♀►¶↑∟ $(,048<
@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|
My Super Sweet 16-Bit Malware:°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕
$MS-DOS Editionⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTXNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
MS-DOS EditionNNNNNNNNNNNNNNNNNNNNNNNNN€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€€
My Super Sweet 16-Bit Malware:€€€€€€ê╘╪▄αΣΦ∞≡⌠°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|
$MS-DOS Edition╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕
MS-DOS Edition°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ€€€€€€€€€€€€€€€€€€€€€€€€€
My Super Sweet 16-Bit Malware:€€€€€€€€€€€€€€€€€€€€€€€€€AAAAAAAAAAAAAAAAAAAAAAAAA
$MS-DOS EditionAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA(¶↑∟ $(,048<@DHLPTX\`dhlptx|
MS-DOS Edition╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕
My Super Sweet 16-Bit Malware:,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
$MS-DOS Edition,,,,,,,╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫╫
MS-DOS Edition╫╫╫╫╫fÉöÿ£áñ¿¼░┤╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ
                                   ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕
$MS-DOS Editionⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ
MS-DOS Edition8<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ ♦◘♀►¶↑∟ $(,048<
My Super Sweet 16-Bit Malware:──────────────────────────────────────────────────
$MS-DOS Editionddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddddd
MS-DOS Edition°ⁿ ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕╝└─╚╠╨╘╪▄αΣΦ∞≡⌠°ⁿ
 ♦◘♀►¶↑∟ $(,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼. . . . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . .ÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜ
ÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜÜ4,048<@DHLPTX\`dhlptx|ÇäêîÉöÿ£áñ¿¼░┤╕
ÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇÇ
                                                                             ║8<
```

Greatest Hits of MS-DOS Malware
or
“Not just a Pretty Payload”

WALKER



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: Q-WALKER

```
C:\WALKER>dir
Directory of C:\WALKER\.
.                <DIR>             26-11-2022 23:40
..               <DIR>             26-11-2022 23:40
Q-WALKER COM              10,338 26-11-2022 23:40
    1 File(s)             10,338 Bytes.
    2 Dir(s)         262,111,744 Bytes free.

C:\WALKER>debug Q-WALKER.COM
```

MS-DOS malware Techniques in the MITRE ATT&CK Framework [non-extensive but wow this list looks so boring you wouldn’t know it!]

- Data Manipulation [T1565] https://attack.mitre.org/techniques/T1565/
- Replication through Removable Media https://attack.mitre.org/techniques/T1091/
- ***Masquerading https://attack.mitre.org/techniques/T1036/004/***
- Masquerading: Match Legitimate Name or Location [T1036:005] https://attack.mitre.org/techniques/T1036/005/
- Masquerading: Masquerade Task or Service [T1036:004] https://attack.mitre.org/techniques/T1036/004/
- Data Obfuscation [T1001] https://attack.mitre.org/techniques/T1001/
- System Services [T1569] https://attack.mitre.org/techniques/T1569/
- Direct Volume Access [T1006] https://attack.mitre.org/techniques/T1006/
- File and Directory Discovery [T1083] https://attack.mitre.org/techniques/T1083/
- Boot or Logon Autostart Execution [T1547] https://attack.mitre.org/techniques/T1547/
- Defacement [T1491] https://attack.mitre.org/techniques/T1491/
- Pre-OS Boot: Bootkit

MS-DOS Malware Techniques

Classic Malware
Stealth
+
Persistence

Level 10 SAVAGE
Destruction of the
MBR and/or
boot sector

Exquisite Graphical Rendering/
Data Manipulation
using system functions

VX Sources

- vx-underground GitHub — MS-DOS Malware collection: https://github.com/vxunderground/MalwareSourceCode/tree/main/MSDOS
- “Internet Archive — Malware Museum,” Mikko Hypponen, https://archive.org/details/malwaremuseum
  NOTE: These are defanged binaries, they are useful for preliminary research but lack the malicious functionality that it interesting from an RE/malware analysis perspective
- The zine archives on VX-UG, primarily 40hex and 29a zine archives
- A myriad of knowledgeable experts who wish to remain anonymous

16-bit Malware RE Methodology

- Preliminary research
- Static Analysis:
  - radare2 (I wrote an r2 plugin for automatically identifying interrupts + adding annotations to the disassembly)
  - Cutter (for when I’m too tired to use r2)
  - IDA Free 5.0 (rip 16-bit support </3)
  - Reading the source files (majority of the source files are written in x86 assembly, with syntax specific to a range of assemblers (MASM, TASM, FASM, A86, etc…)
    - Assembling the source using one of the many assemblers
    - … or making modifications to the source for use with a different assembler (NASM); mixed results
- Dynamic Analysis:
  - QEMU + FreeDOS
  - Bochs
  - DosBox (more useful for testing sample programs and performing basic analysis, not as flexible as QEMU+FreeDOS which is better for more involved dynamic analysis)
- *For samples where a compiled binary was not available for dynamic analysis, an auxiliary source of information is danooct1 YouTube channel: https://www.youtube.com/@danooct1
  Specifically their “MS-DOS malware” playlist:
  https://youtube.com/playlist?list=PLi_KYBWS_E71ObQ8QpGj5zIDXHREbdWaM

MS-DOS Edition

My Super Sweet 16-Bit Malware:

$MS-DOS Edition

MS-DOS Edition

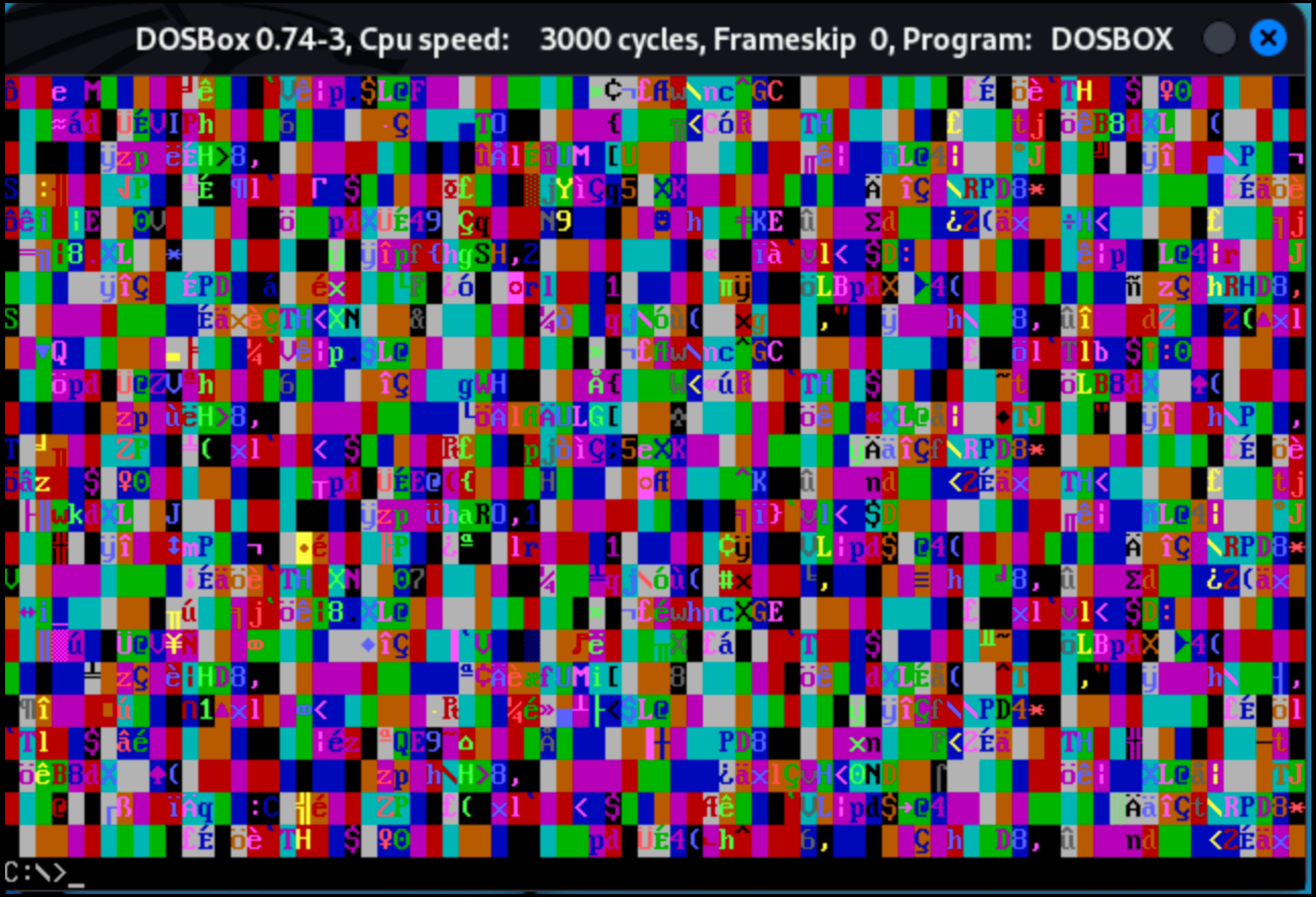
Greatest Hits of MS-DOS Malware: ~~The Clash~~ CRASH.COM

CRASH.COM

Infinite loop of pretty animation
Direct write to the VGA buffer makes the computer unusable and forces a user to reboot to use their machine

Copies the payload [~*pretty animation on infinite loop*~] to target files on the machine

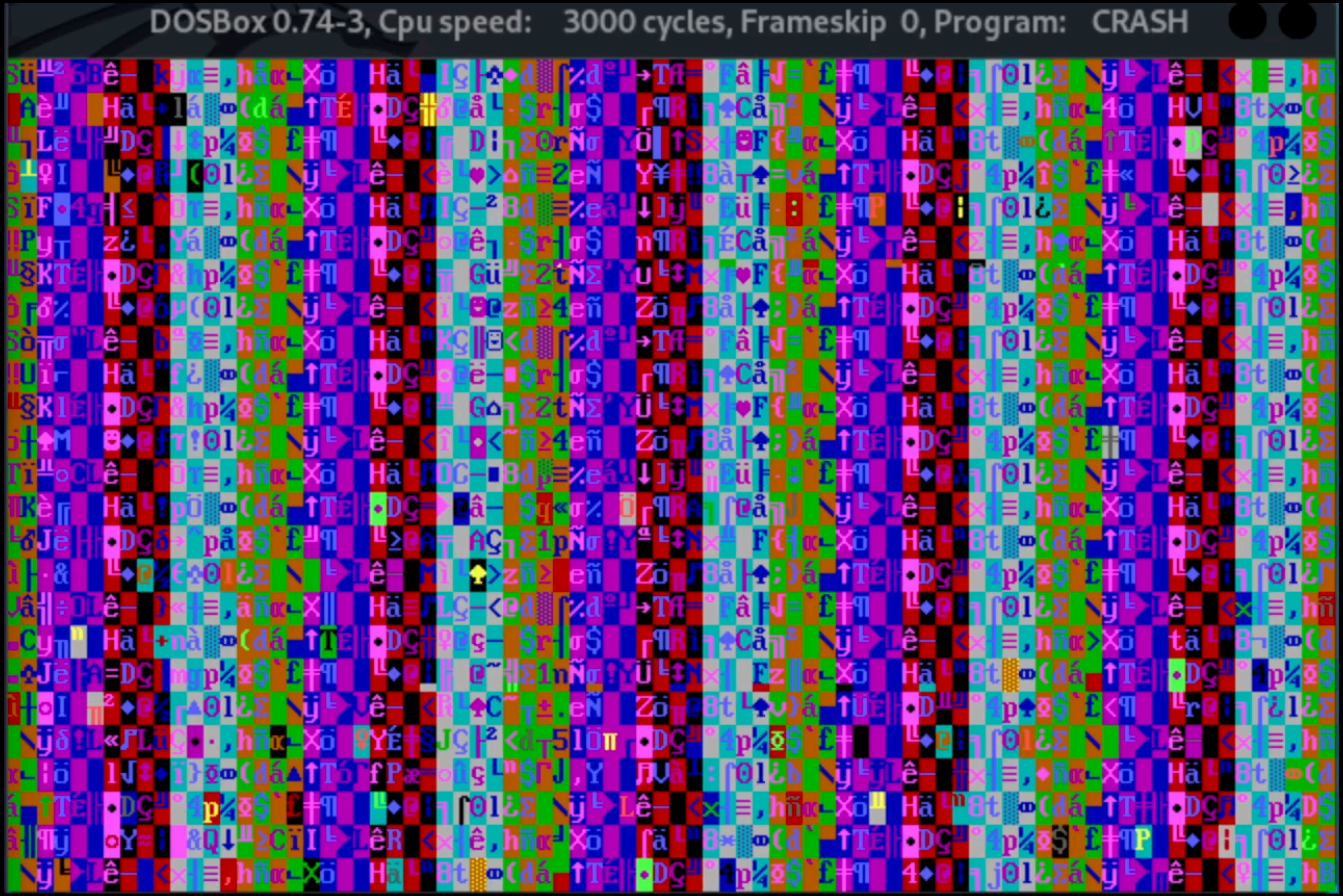
Less destructive ** than other viruses of the time (especially compared with those that used this same VGA buffer technique)


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

```
C:\>_
```

CRASH.COM

WARNING: Mild Flashing Lights
[appx. 6 seconds]



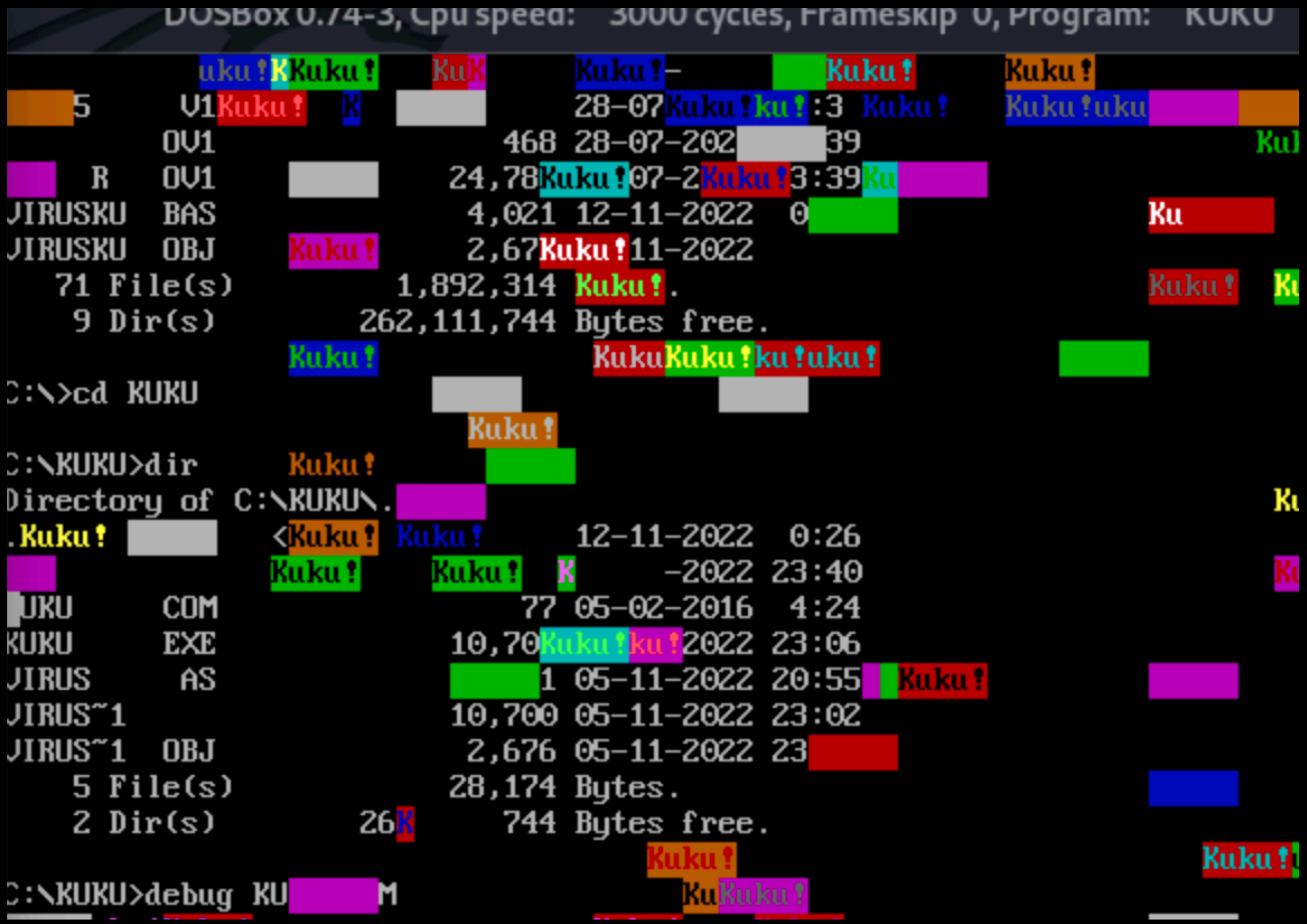
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: CRASH

Greatest Hits of MS-DOS Malware

~~The Kooks~~ KUKU.COM

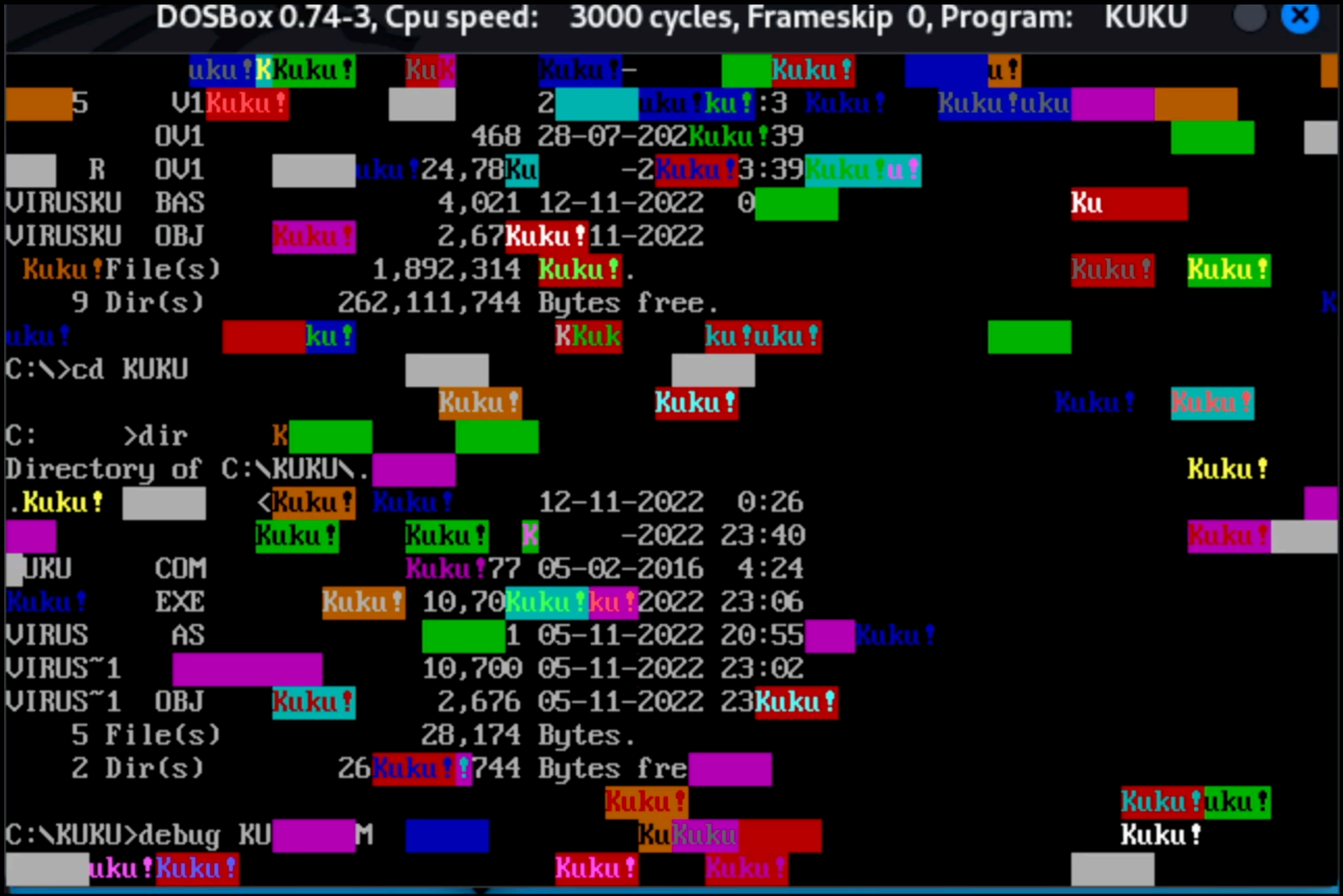
KUKU

- Searches the filesystem for .exe and .com files, overwrites them with the KUKU.COM virus payload
- When executed, displays this obscene sequence of colorful boxes with the phrase “Kuku!” to the command prompt
- Fun (?) fact: kuku (kyky) means “peekaboo” in Russian. The virus is searching for target files to infect and displaying a message “peekaboo, I see you!” every time it finds one.


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: KUKU

```
          uku!KKuku!    Kuk       Kuku!-         Kuku!     Kuku!
    5     V1Kuku!   K             28-07Kuku!ku!:3 Kuku!    Kuku!uku
         OV1                  468 28-07-202     39                 Kul
     R   OV1                24,78Kuku!07-2Kuku!3:39Ku
JIRUSKU  BAS                 4,021 12-11-2022  0              Ku
JIRUSKU  OBJ       Kuku!     2,67Kuku!11-2022
   71 File(s)          1,892,314 Kuku!.                        Kuku!   Ku
    9 Dir(s)         262,111,744 Bytes free.
                   Kuku!            KukuKuku!ku!uku!
C:\>cd KUKU
                             Kuku!
C:\KUKU>dir        Kuku!
Directory of C:\KUKU\.                                                Ku
.Kuku!               <Kuku! Kuku!          12-11-2022  0:26
                    Kuku!    Kuku!  K         -2022 23:40           Ku
UKU       COM                      77 05-02-2016  4:24
KUKU      EXE                  10,70Kuku!ku!2022 23:06
JIRUS      AS                       1 05-11-2022 20:55 Kuku!
JIRUS~1                        10,700 05-11-2022 23:02
JIRUS~1   OBJ                   2,676 05-11-2022 23
    5 File(s)                  28,174 Bytes.
    2 Dir(s)          26K         744 Bytes free.
                                        Kuku!                  Kuku!
C:\KUKU>debug KU     M                    KuKuku!
```

KUKU



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: KUKU

```
           uku!KKuku!     KuK     Kuku!-       Kuku!      u!
  5     V1Kuku!         2    uku!ku!:3 Kuku!  Kuku!uku
         OV1              468 28-07-202Kuku!39
     R   OV1         uku!24,78Ku    -2Kuku!3:39Kuku!u!
VIRUSKU  BAS            4,021 12-11-2022  0                Ku
VIRUSKU  OBJ     Kuku!     2,67Kuku!11-2022
 Kuku!File(s)       1,892,314 Kuku!.                   Kuku!  Kuku!
    9 Dir(s)      262,111,744 Bytes free.                          K
uku!           ku!          KKuk    ku!uku!
C:\>cd KUKU
                       Kuku!      Kuku!                Kuku!  Kuku!
C:     >dir      K
Directory of C:\KUKU\.                                        Kuku!
.Kuku!          <Kuku! Kuku!    12-11-2022   0:26
                 Kuku!    Kuku!  K     -2022 23:40            Kuku!
 UKU     COM              Kuku!77 05-02-2016   4:24
Kuku!    EXE        Kuku! 10,70Kuku!ku!2022 23:06
VIRUS     AS                1 05-11-2022 20:55 Kuku!
VIRUS~1                 10,700 05-11-2022 23:02
VIRUS~1  OBJ     Kuku!     2,676 05-11-2022 23Kuku!
    5 File(s)          28,174 Bytes.
    2 Dir(s)        26Kuku!!744 Bytes fre
                                  Kuku!                Kuku!uku!
C:\KUKU>debug KU    M           KuKuku                Kuku!
     uku!Kuku!                 Kuku!   Kuku!
```

Greatest Hits of MS-DOS Malware

VIRDEM.COM

VIRDEM

- A demo virus written by Ralf Burger (he also wrote this book: “Computer Viruses and Data Protection: Unclassified,” Ralf Burger, Abacus Software, 1991) [I bought it on Amazon but there may be copies online somewhere]

- According to Ralf, the VIRDEM virus does the following:

1. All COM files up to 2nd subdirectory are infected

2. Does not infect 1st COM file in root dir (this 1st COM file is usually COMMAND.COM)

3. COM files > 1.5kb, length increased by ~1.5kb;
   COM files < 1.5kb, length increased by ~3.0kb

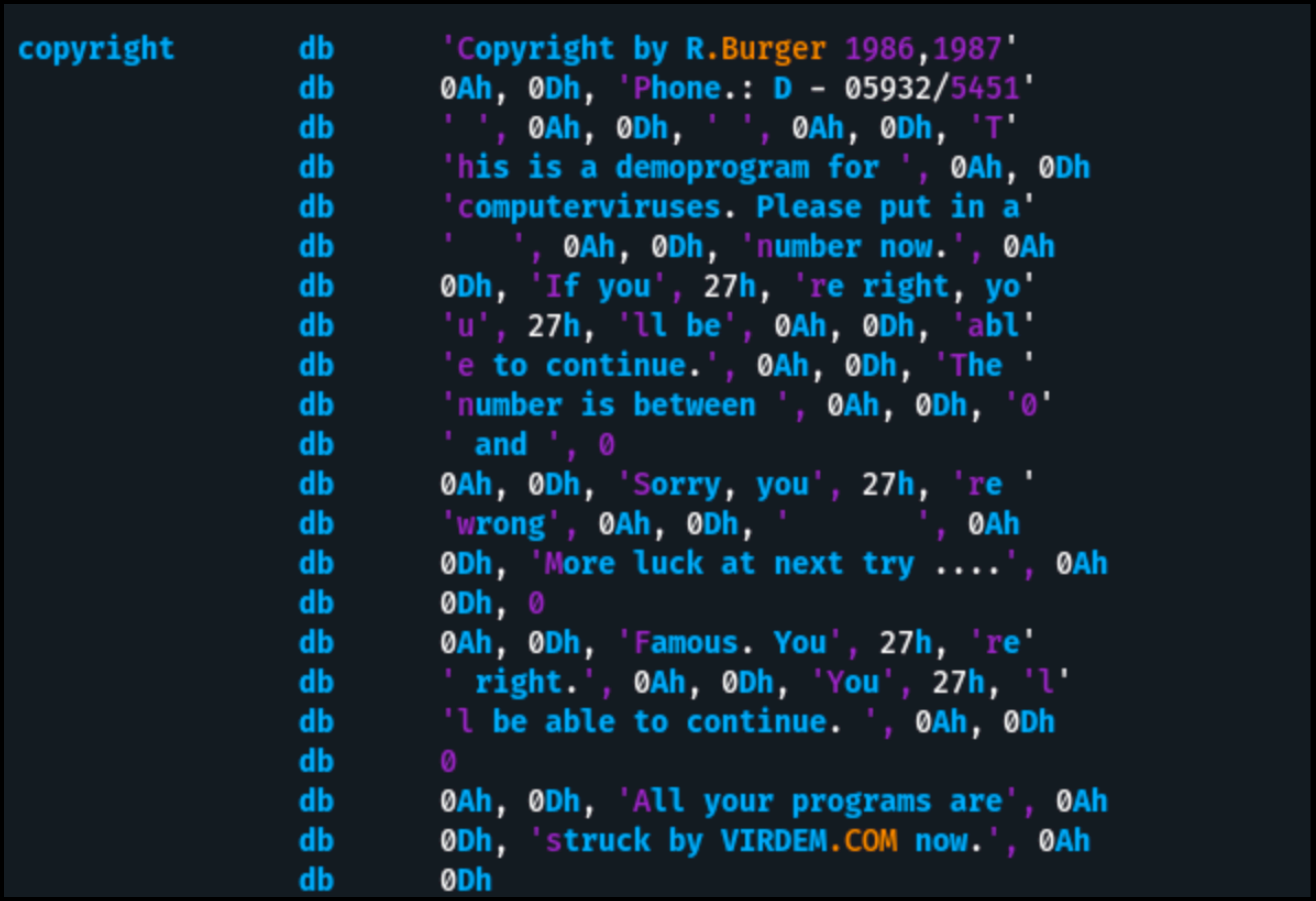
4. “Infected programs remain completely functional” (OKAY RALF)

5. An infected program is recognized and cannot be infected twice

6. VIRDEM.COM inserts an additional function into the infected program, a bizarre guessing game “whose difficulty level is dependent on the virus generation”

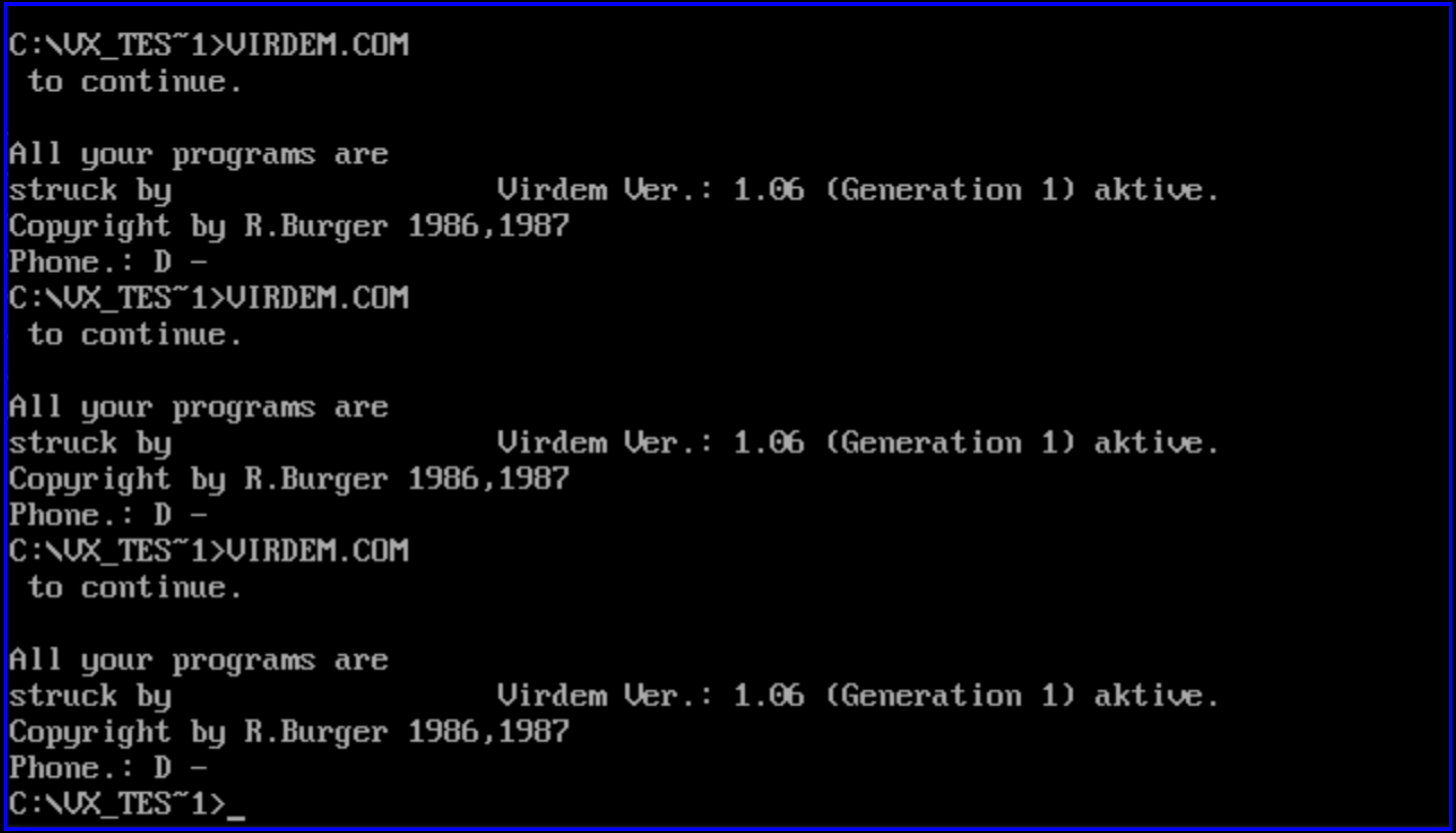
7. VIRDEM mutates up to the 9th generation, and then stops mutating

**[“Computer Viruses and Data Protection: Unclassified,” Ralf Burger, pages 210-211]



```
copyright       db      'Copyright by R.Burger 1986,1987'
                db      0Ah, 0Dh, 'Phone.: D - 05932/5451'
                db      ' ', 0Ah, 0Dh, ' ', 0Ah, 0Dh, 'T'
                db      'his is a demoprogram for ', 0Ah, 0Dh
                db      'computerviruses. Please put in a'
                db      '   ', 0Ah, 0Dh, 'number now.', 0Ah
                db      0Dh, 'If you', 27h, 're right, yo'
                db      'u', 27h, 'll be', 0Ah, 0Dh, 'abl'
                db      'e to continue.', 0Ah, 0Dh, 'The '
                db      'number is between ', 0Ah, 0Dh, '0'
                db      ' and ', 0
                db      0Ah, 0Dh, 'Sorry, you', 27h, 're '
                db      'wrong', 0Ah, 0Dh, '       ', 0Ah
                db      0Dh, 'More luck at next try ....', 0Ah
                db      0Dh, 0
                db      0Ah, 0Dh, 'Famous. You', 27h, 're'
                db      ' right.', 0Ah, 0Dh, 'You', 27h, 'l'
                db      'l be able to continue. ', 0Ah, 0Dh
                db      0
                db      0Ah, 0Dh, 'All your programs are', 0Ah
                db      0Dh, 'struck by VIRDEM.COM now.', 0Ah
                db      0Dh
```

VIRDEM



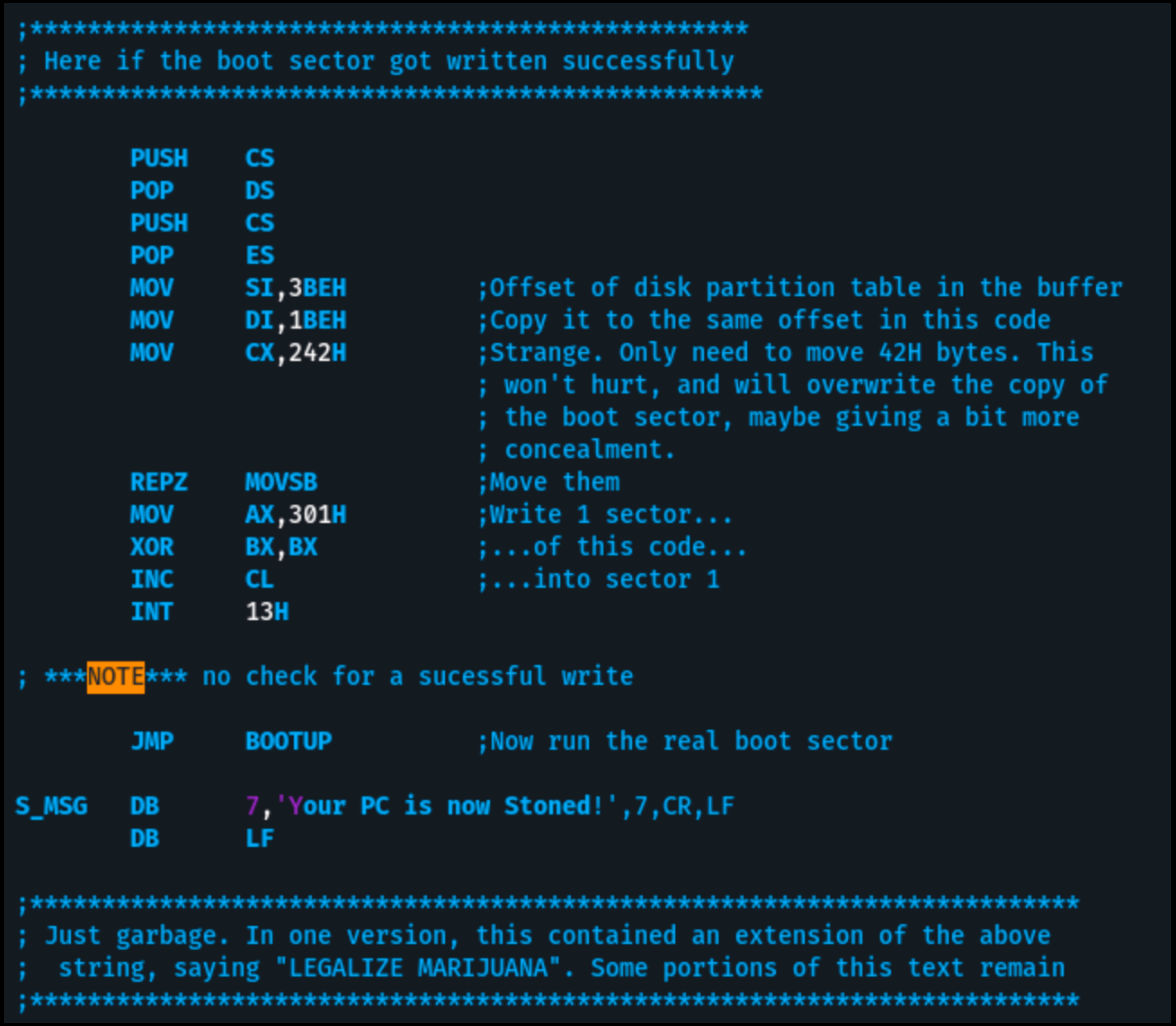
```
C:\VX_TES~1>VIRDEM.COM
 to continue.

All your programs are
struck by                  Virdem Ver.: 1.06 (Generation 1) aktive.
Copyright by R.Burger 1986,1987
Phone.: D -
C:\VX_TES~1>VIRDEM.COM
 to continue.

All your programs are
struck by                  Virdem Ver.: 1.06 (Generation 1) aktive.
Copyright by R.Burger 1986,1987
Phone.: D -
C:\VX_TES~1>VIRDEM.COM
 to continue.

All your programs are
struck by                  Virdem Ver.: 1.06 (Generation 1) aktive.
Copyright by R.Burger 1986,1987
Phone.: D -
C:\VX_TES~1>_
```

Greatest Hits of MS-DOS Malware

STONED.COM

STONED

- Famous bootkit — inspired a range of related bootkits in this virus family, of varying levels of sophistication
  [Michelangelo, what an absolute flop]
- Able to infect boot sectors of multiple different formats of storage media (routines for both floppy diskettes, and for hard drives)
- Stealth
  - Saved the original MBR on a hidden area of the disk
  - Spoofed valid INT 13h reads/writes with a TSR
- Logic bomb - only displayed the famous “Your PC is now Stoned!” message 1/8 times (using PC timer)



```
;**************************************************
; Here if the boot sector got written successfully
;***************************************************


        PUSH    CS
        POP     DS
        PUSH    CS
        POP     ES
        MOV     SI,3BEH                 ;Offset of disk partition table in the buffer
        MOV     DI,1BEH                 ;Copy it to the same offset in this code
        MOV     CX,242H                 ;Strange. Only need to move 42H bytes. This
                                        ; won't hurt, and will overwrite the copy of
                                        ; the boot sector, maybe giving a bit more
                                        ; concealment.
        REPZ    MOVSB                   ;Move them
        MOV     AX,301H                 ;Write 1 sector...
        XOR     BX,BX                   ;...of this code...
        INC     CL                      ;...into sector 1
        INT     13H

; ***NOTE*** no check for a sucessful write

        JMP     BOOTUP                  ;Now run the real boot sector

S_MSG   DB      7,'Your PC is now Stoned!',7,CR,LF
        DB      LF

;*************************************************************************
; Just garbage. In one version, this contained an extension of the above
;  string, saying "LEGALIZE MARIJUANA". Some portions of this text remain
;*************************************************************************
```

MS-DOS Edition

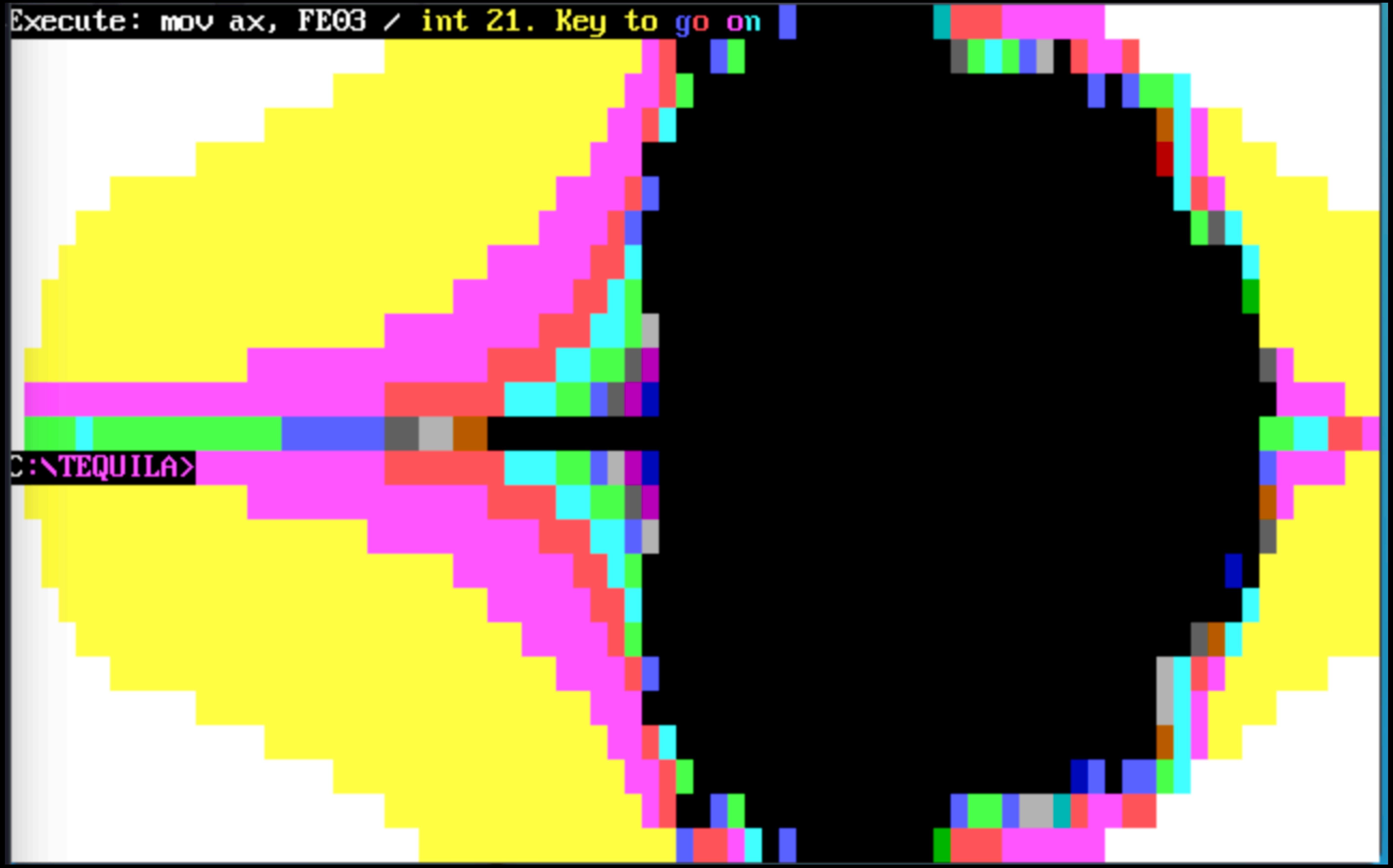
My Super Sweet 16-Bit Malware

MS-DOS Edition

Greatest Hits of MS-DOS Malware ~~Margaritaville~~ TEQUILA.COM

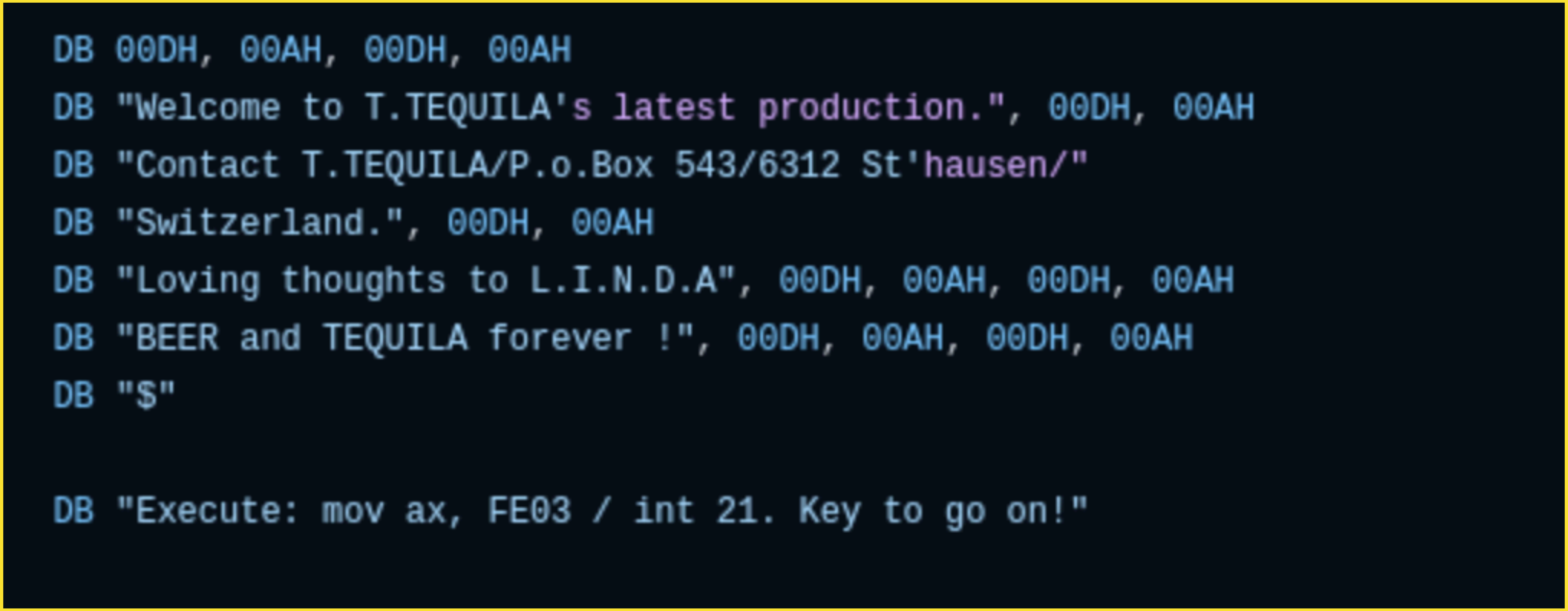
TEQUILA

- Fractal animation (Mandelbrot)
- Savage - infects MBR partition table and installs interrupt handlers to run as a TSR
- What's with the "Mov ax FE03 / INT 21" instruction??
- Multi-part payload — requires user interaction to reveal …



```
Execute: mov ax, FE03 / int 21. Key to go on
C:\TEQUILA>
```

TEQUILA



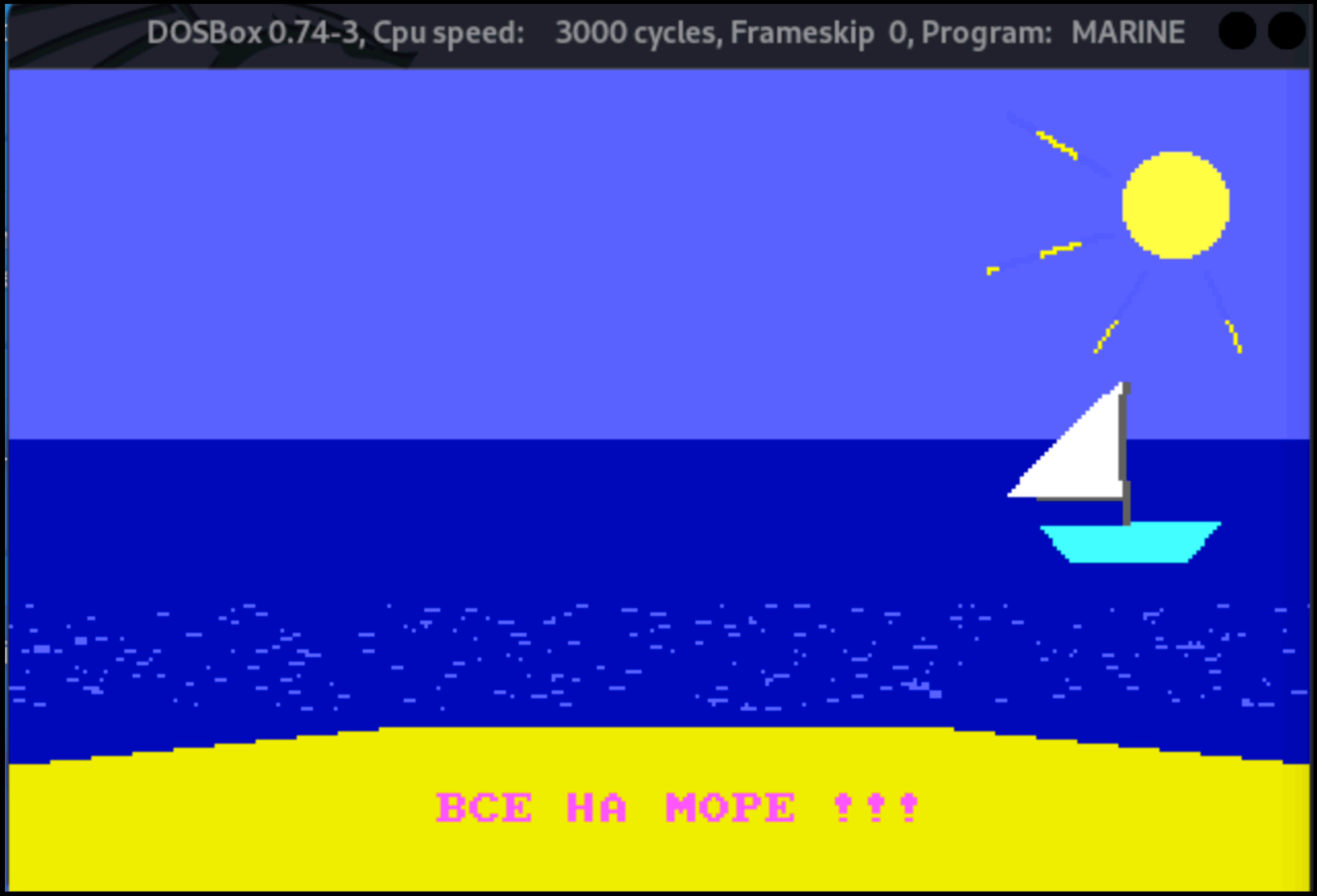
```
DB 00DH, 00AH, 00DH, 00AH
DB "Welcome to T.TEQUILA's latest production.", 00DH, 00AH
DB "Contact T.TEQUILA/P.o.Box 543/6312 St'hausen/"
DB "Switzerland.", 00DH, 00AH
DB "Loving thoughts to L.I.N.D.A", 00DH, 00AH, 00DH, 00AH
DB "BEER and TEQUILA forever !", 00DH, 00AH, 00DH, 00AH
DB "$"

DB "Execute: mov ax, FE03 / int 21. Key to go on!"
```

Greatest Hits of MS-DOS Malware

~~Yellow Submarine~~ MARINE.COM

MARINE

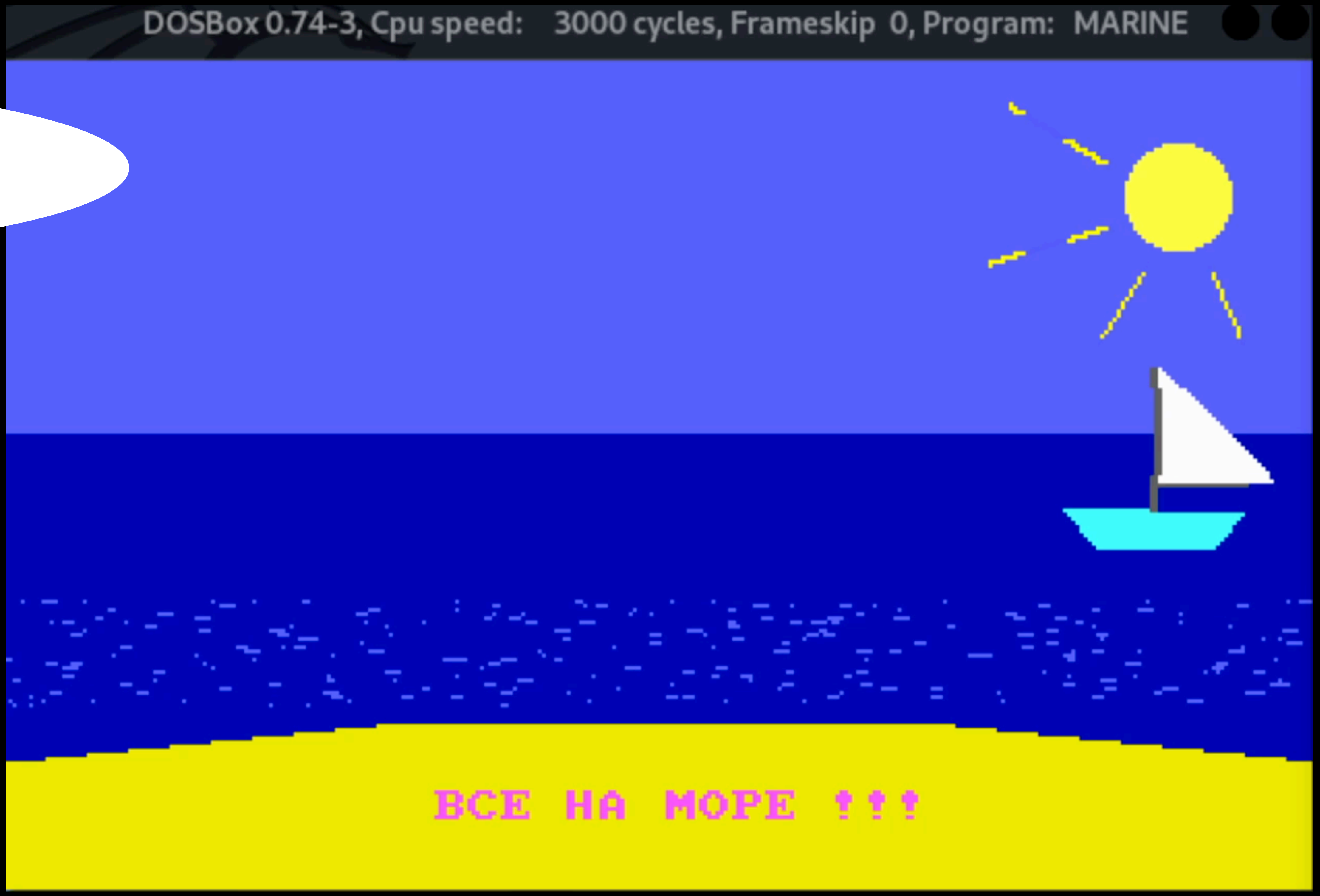
- LEVEL 10 Savage
- Several different payloads, which trigger based on conditions of the virus’ various logic bombs
- The most brutal payload ***shreds the MBR*** while a little boat animation plays
  - Specifically it encrypts the drive
- “Bcë na mope” in Russian means “everyone to the ocean” but the vibe is basically like…



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: MARINE

ВСЕ НА МОРЕ !!!

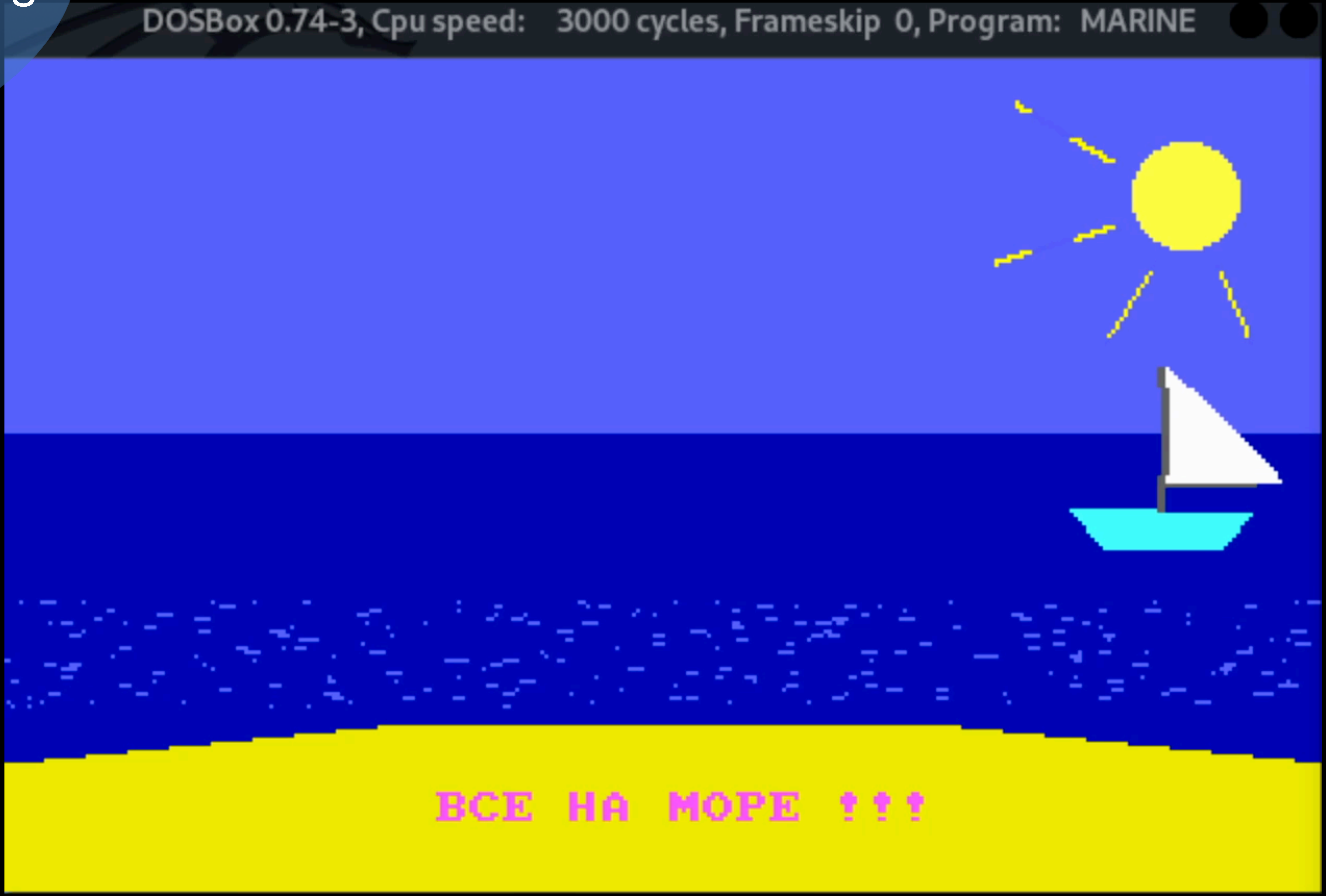
MARINE


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: MARINE

MARINE

Все хорошо

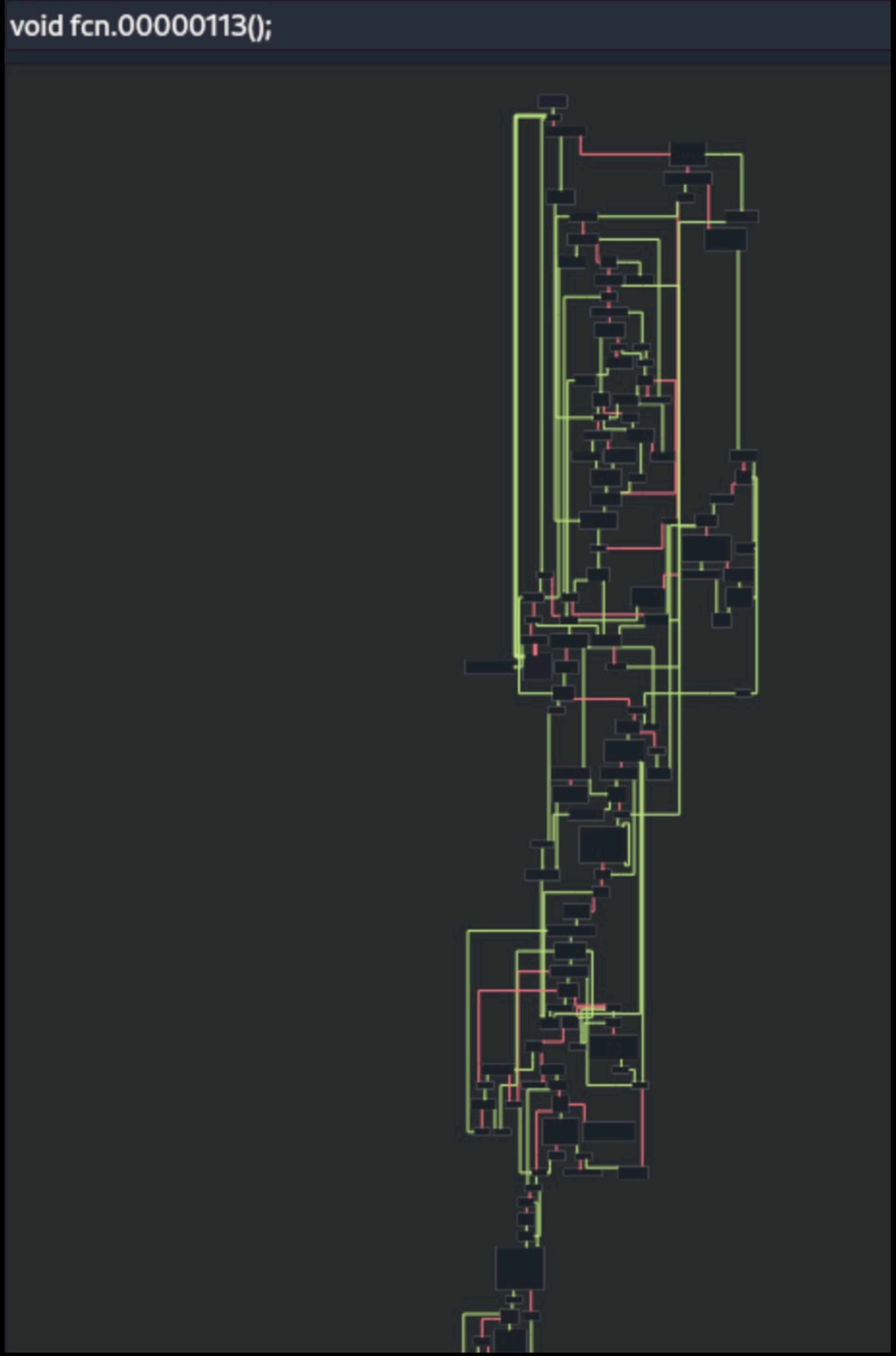
Nothing bad happening to the drive rn



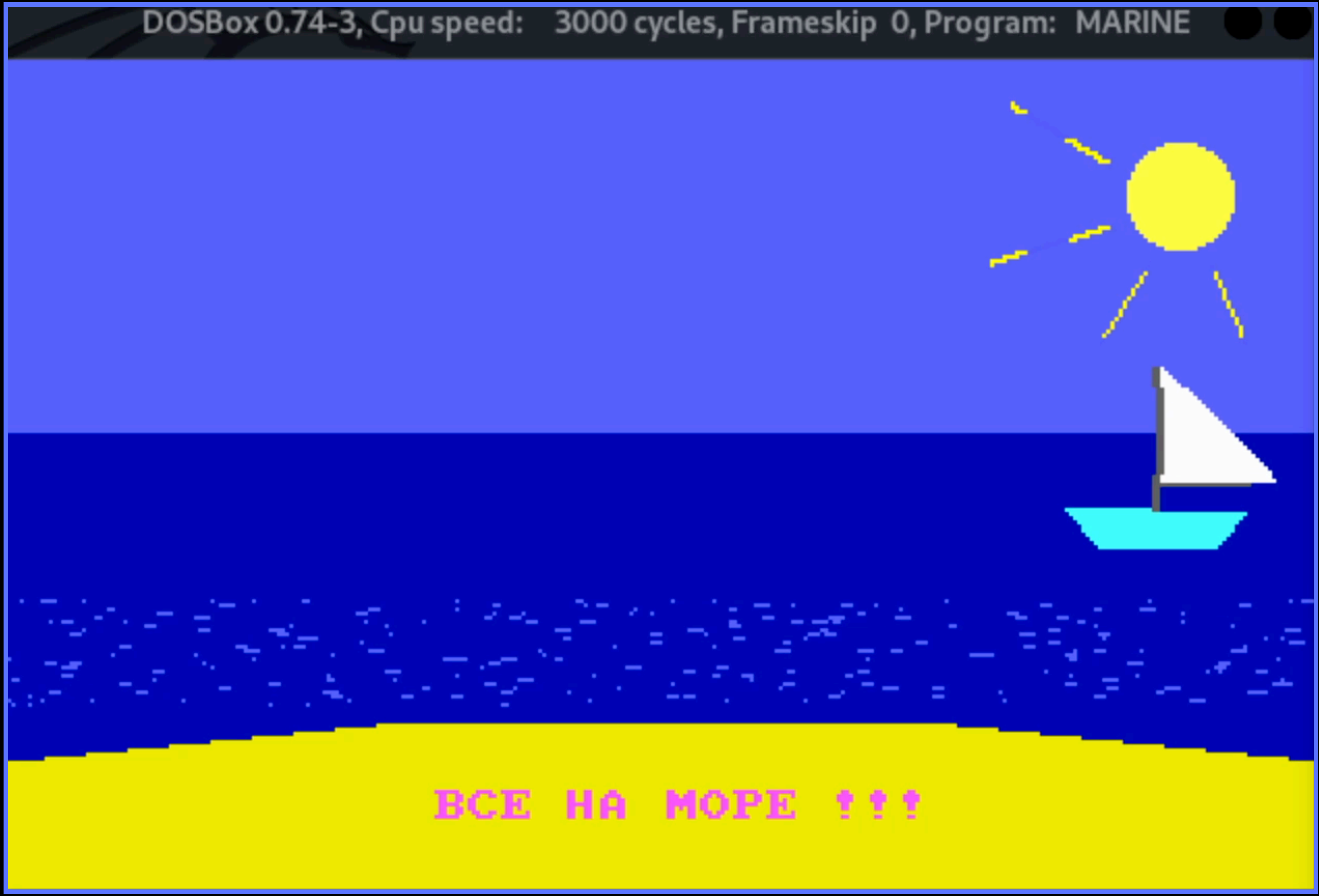
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: MARINE

ВСЕ НА МОРЕ !!!

MARINE



```
void fcn.00000113();
```



DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: MARINE

ВСЕ НА МОРЕ !!!

Why Study an EOL OS?

Malware — same as it ever was…

“Once in a Lifetime,” The Talking Heads, 1980

Why Study MS-DOS malware?

- Foundational techniques of malware masters of the ‘80s and ‘90s inform malware techniques up to the present day
- For one, legacy BIOS interrupts featured prominently in malware like bootkits through much later versions of Windows [specifically through Windows 7 before the switch to UEFI from the legacy boot process]
- Relevant case study: NotPetya
  - Prominent use of the INT 13h BIOS interrupt to call disk services functions
- And while the switch to UEFI firmware from the legacy boot process (meaning the use of legacy BIOS interrupts) effectively changed the landscape for Windows bootkits, modern bootkits still use techniques that were developed by the earliest bootkit writers in the “BSI heyday”

“Rootkits and Bootkits: Reversing Modern Malware and Next Generation Threats,” Alex Matrosov, Eugene Rodionov, and Sergey Bratus

Why Study MS-DOS malware?

- “Different names for the same thing” — reusing the IVT in its new incarnation, the IDT, for similar malicious activity
  - Rovnix, reusing the IDT above 0x80 (Bootkit from 2011)[1]
- Become inspired to rewrite an old classic and show that a modern OS can still be vulnerable to boot process exploits due to its backwards compatibility with the legacy BIOS boot process
  - The Stoned Bootkit… at Blackhat 2012
- And while the IVT got upgraded to the IDT, it remained a useful data structure to leverage for hooking system calls on later versions of Windows; the IDT and SSDT (System Service Descriptor Table) were both used to this effect. And hooking the SSDT and IDT were techniques employed by rootkits of the Windows NT rootkit heyday [2]
- Looking into the past can give you ideas for where to search for inspiration for exploits next, and which vectors are ignored now but were favorites in past eras.

[1]“Rootkits and Bootkits: Reversing Modern Malware and Next Generation Threats,” Alex Matrosov, Eugene Rodionov, and Sergey Bratus
[2]“Rootkits: Subverting the Windows Kernel” by Greg Hoglund and James Butler, page 82-95, “Chapter 4:The Age-Old Art of Hooking.”

Where to now?

- I have a repo on GitHub with some sample COM programs, learning resources, a guide for setting up a reversing lab: https://github.com/nikaroxanne/supersweet16bit-m4lw4r3
- Check out my website for deep-dive blog posts exploring specific subtopics (i.e. TSRs, early bootkits, etc.): https://ic3qu33n.fyi/projects/mySuperSweet16BitMalwareMSDOSEdition
- I've written an r2 plugin for reversing 16-bit binaries (specifically malicious COM files); contact me on one of the socials (on in person after the talk) if you would like to be involved in ~*test driving*~ the r2 plugin
- I made you a playlist for when you're reversing 16-bit malware: https://open.spotify.com/playlist/7KKMdu8JaLEoLV66uZqKG0?si=bf876ac6c71f4ecf [If you are adamantly opposed to Spotify as a platform, then provide me with your medium of choice and I will burn you a copy. Bonus points if that medium of choice is a FAT stack of floppy disks (someone better appreciate this joke I stg)]

References

“Advanced MS-DOS Programming,” Ray Duncan, Microsoft Press, 1986

“Microsoft MS-DOS Programmer’s Reference,” Microsoft Corporation, 2nd ed.: version 6.0., Microsoft Press, 1993

"The Giant Black Book of Computer Viruses," Mark Ludwig, 2nd ed., American Eagle Books, 1998.

"Rootkits and Bootkits: Reversing Modern Malware and Next Generation Threats,”Alex Matrosov, Eugene Rodionov, and Sergey Bratus, No Starch Press, 2019

“Computer Viruses and Data Protection: Unclassified,” Ralf Burger, Abacus Software, 1991

“A Look Back at Memory Models in 16-bit MS-DOS,” Raymond Chen, The Old New Thing, Microsoft Blogs, July 28, 2020, https://devblogs.microsoft.com/oldnewthing/20200728-00/?p=104012

“On Memory Allocations Larger Than 64KB on 16-Bit Windows,” Raymond Chen, The Old New Thing, Microsoft Blogs, https://devblogs.microsoft.com/oldnewthing/20171113-00/?p=97386

“Retired Malware Samples, Everything Old is New Again,” Lenny Zeltser, August 1, 2018. https://zeltser.com/retired-malware-samples-retrospective/