LOAD"AMBER64",8,1

SEARCHING FOR AMBER64
LOADING
READY.
RUN

Amber64

Mining Hacker History from Over Half a Million Commodore 64 Disks

Dr. Wesley McGrew
Senior Cyber Fellow
martinfed

Agenda

- Background on 80’s computing and the C64
- The corpus of 653,446 disk images and how they came to be
- Motivations
- Applying modern digital forensics
- Data structures used by the C64 and 1541 disk drive’s Commodore DOS
- Processing the disk images into Elasticsearch for indexed search
- A sample of fun findings
- Future work and conclusions

Computing in the 1980s

- Hacking/Phreaking
  - Wardialing
  - Boxes/Tones
  - Codes
- BBS
  - Proto-forum/mailing list
  - File trade
  - Cross-platform
  - Motivation for hack/phreak activity
- Software Piracy
  - Cracking protections
  - Distribution
  - Groups
  - Bragging rights
  - Motivation for a lot of other activity (the pornography wasn't very good yet)
- Demo Scene
  - "hacking" in the sense of demonstrating low-level systems knowledge

Commodore 64

- Held the Guiness Book World Record, Highest Selling Single Computer Model. (12.5 to 17 million)
- Production: August 1982 to April **1994 (!)**
- Keyboard built into the computer
- Composite video, often connected to a television
- Huge library of games, a lot of applications
- First computer for “hackers of a certain age”
- First opportunity to connect via modem to BBS’s, Q-Link, CompuServe
- Most common disk drive: 5.25” Commodore 1541



https://en.wikipedia.org/wiki/Commodore_64#/media/File:Commodore-64-Computer-FL.jpg

C64

- MOS 6510 at ~1 MHz
- 64K Memory Space
  - Fully-backed by RAM
  - 20KB of ROM
    - KERNAL & BASIC
    - can be switched in and out
- VIC-II for graphics, video signals
- SID – Digitally controlled analog synth sound
- Serial IEC bus for communications with drives/etc.

1541

- MOS 6502 at ~1 MHz
- Memory
  - 2KB RAM
  - 16KB ROM – Commodore DOS (no relation at all to MSDOS)
- Serial IEC
- ~170KB data per floppy disk
- Slow – Lots of software and hardware modifications for speeding it up

Life of a C64 Disk

- Formatted
- Cycle:
  - Downloaded software from BBS, copied from a friend
  - Till the disk fills up
  - Delete some old ones, start the cycle again with fresh warez
  - Disk full of stuff you want to keep? Store it in the floppy box
  - *Rarely label anything accurately*
  - Rinse and repeat
- Other disks might contain data rather than programs
  - Text files
  - BBS sessions (recorded to minimize call times)
- Imaged and archived, distributed on the modern Internet as .d64

653,446 D64 Images

- The Old School Emulation Center – TOSEC
- C64 Preservation Project
- Internet Archive – Collections and individual downloads
- The Commodore 64 Scene Database (CSDB)
  - Scraped releases
  - Forum threads with large collections
- FTP sites – zimmers, arnold, padua, etc.
- Loadstar Compleat (disk magazine)
  - $15, amazing value: https://rodneylives.itch.io/loadstar
- Other
- Overall, many well-documented games, and an overwhelming number of disks that we have—at best—”surface” level knowledge of the contents

Classifying Artifacts

- A floppy disk as a collection of artifacts
  - Apparent
    - Labeled Program - Printed, on the sticker, on the disk
    - “First” program on the disk
      - Most people are going to **LOAD “*”,8,1**
    - Allocated files (**LOAD “$”,8**)
      - Mixture of other programs, data files that may or may not be related
  - Hidden
    - Unlabeled disks
    - “Extra” files
    - Deleted files
    - Unallocated/orphaned space
    - Shenanigans

Motivation

- Lost Media – Games, applications, demos
- Hidden text (such as that documented on The Cutting Room Floor (tcrf.net)
- Comparing versions of software
- Ephemera
  - Communications
  - User-generated files/data
- Hacking history: BBS, phreaking, codes, piracy, cracks, etc.

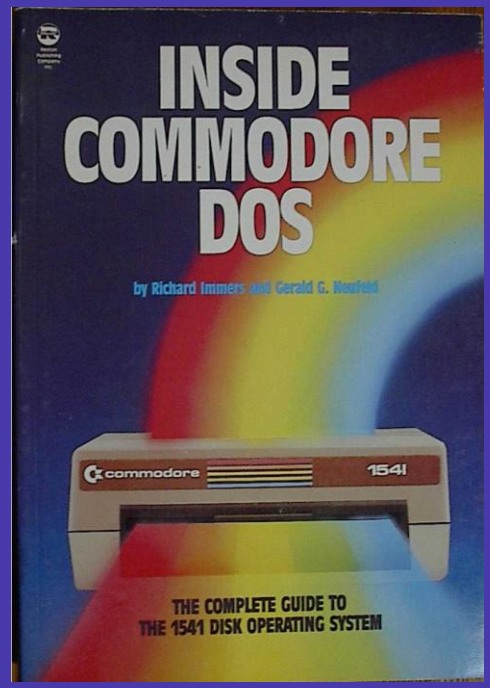
What we can apply

- File system forensic analysis (modern, relative to this domain)
  - Forensic imaging (fortune shines: .d64 format retains most of what we'd want, is essentially a sector-for-sector copy)
  - File system metadata parsing
  - Recovery of deleted files
  - Recovery from free space, allocated-but-not-used space
  - Slack space (between ends of files and ends of sectors, for example)
- Indexed text search
  - Near instantaneous text searching, allowing for easy exploration/analysis
- Modern computing environment
  - Disk space and processing time

Commodore 1541 Disk Layout & Data Structures

Diagrams are from Immers, Neufield, *Inside Commodore DOS,* DATAMOST Inc, 1984

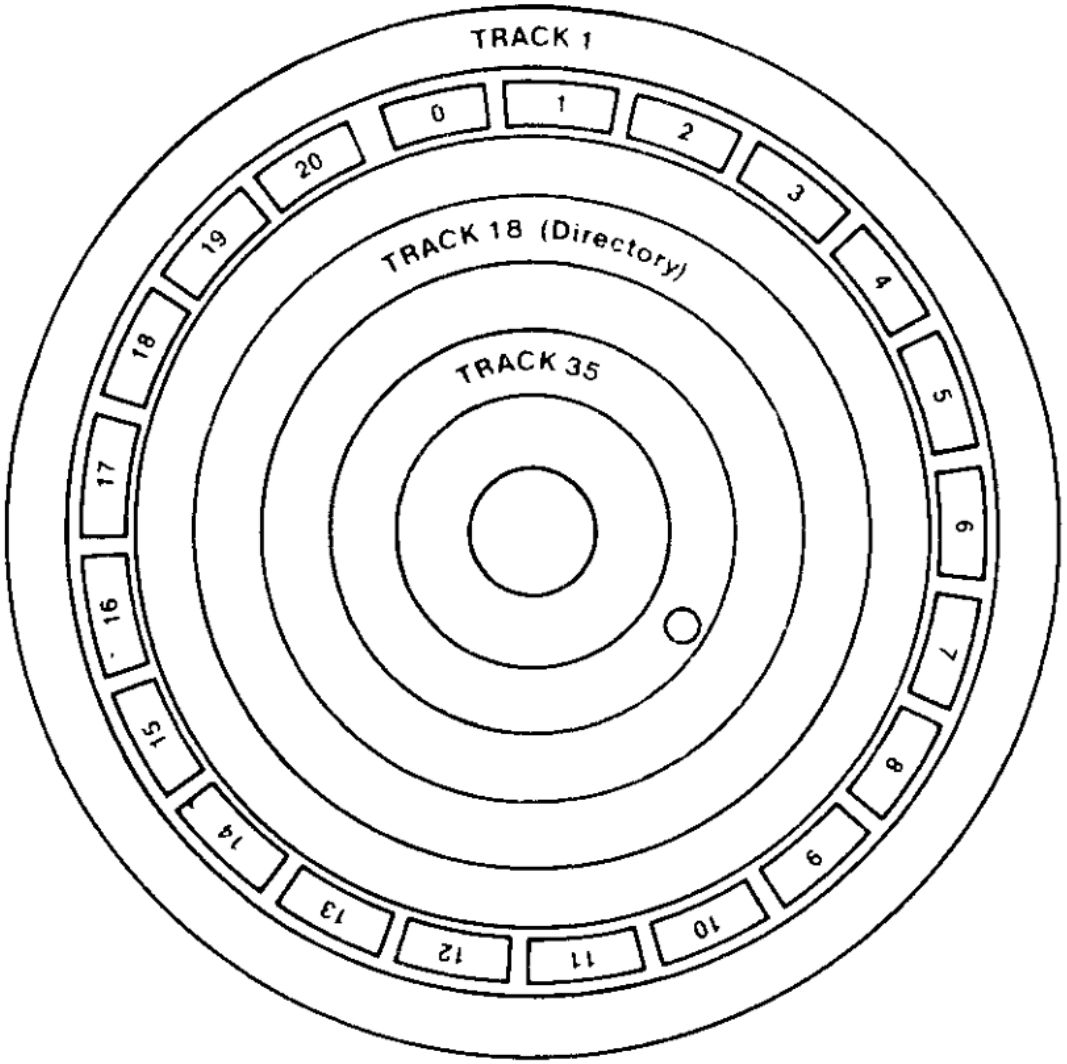
(the book credits technical illustrations to Diane M. Corralejo)



This section also known as

“Why can’t I just use strings and grep?”

Tracks and Sectors

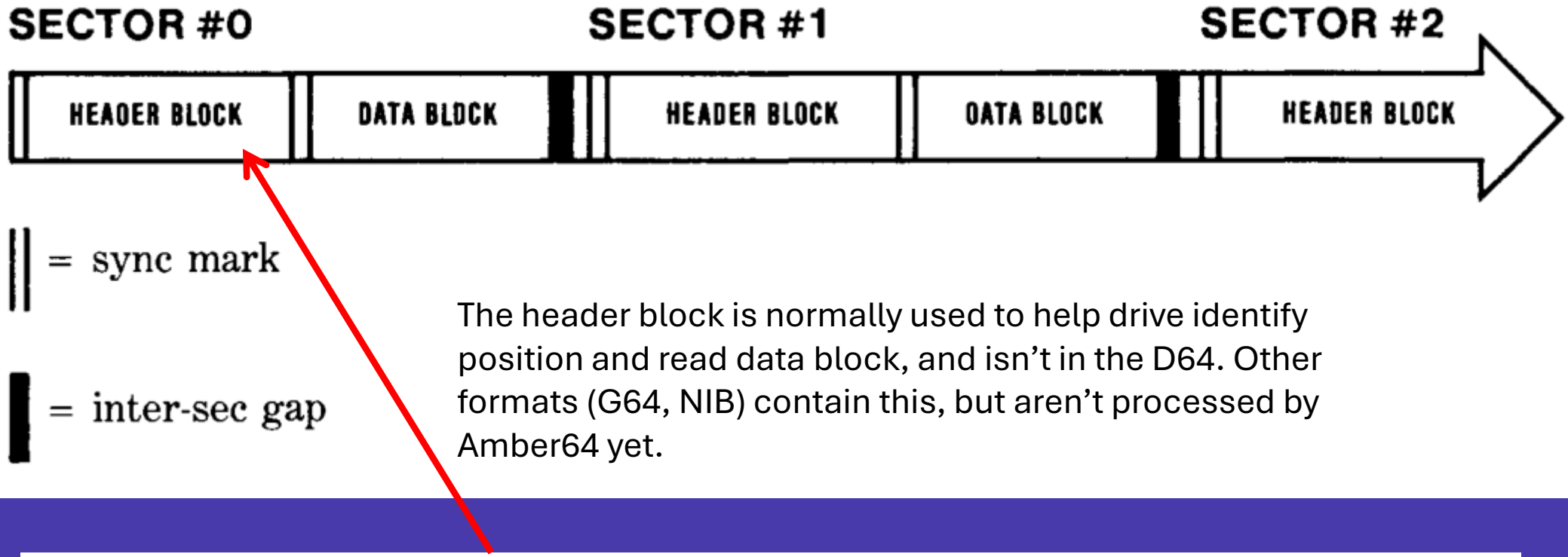


Organization of Tracks and Sectors on a 1541 Formatted Diskette

Zone	Track Numbers	Range of Sector Numbers	Total Sectors Per Track	Total Bytes Per Track
1	1 to 17	0 to 20	21	5376
2	18 to 24	0 to 18	19	4864
3	25 to 30	0 to 17	18	4608
4	31 to 35	0 to 16	17	4352

.D64 image format is simply a collection of all of these sectors, 256 bytes each, in order.

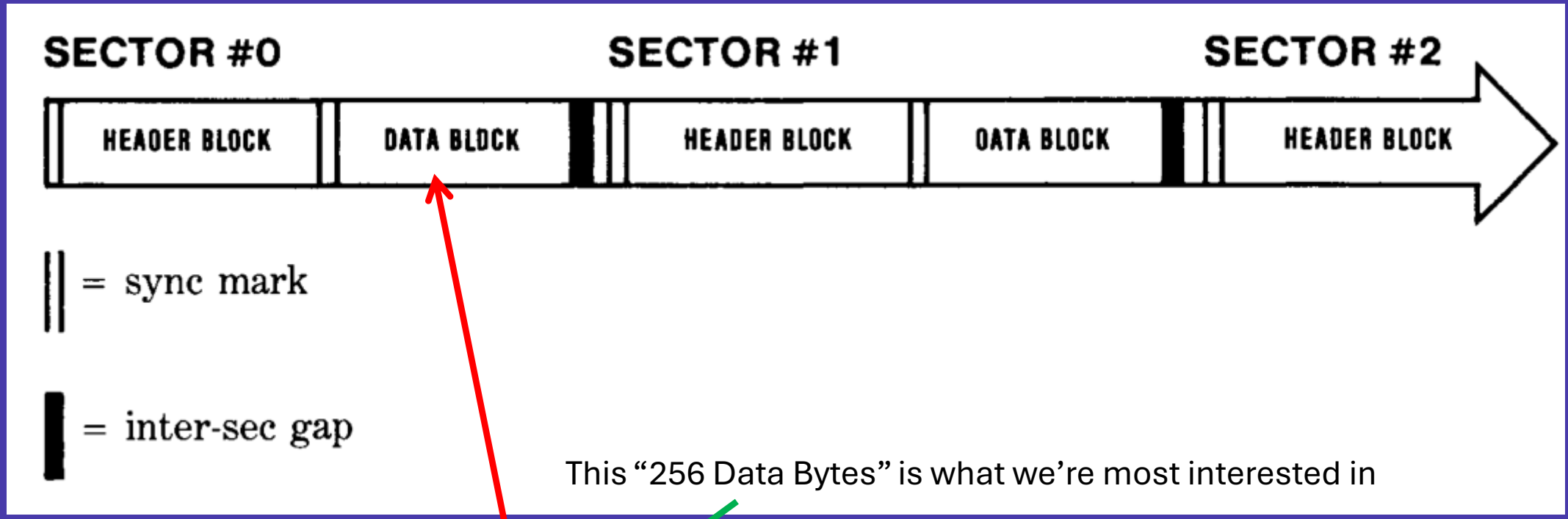
Sector Layout

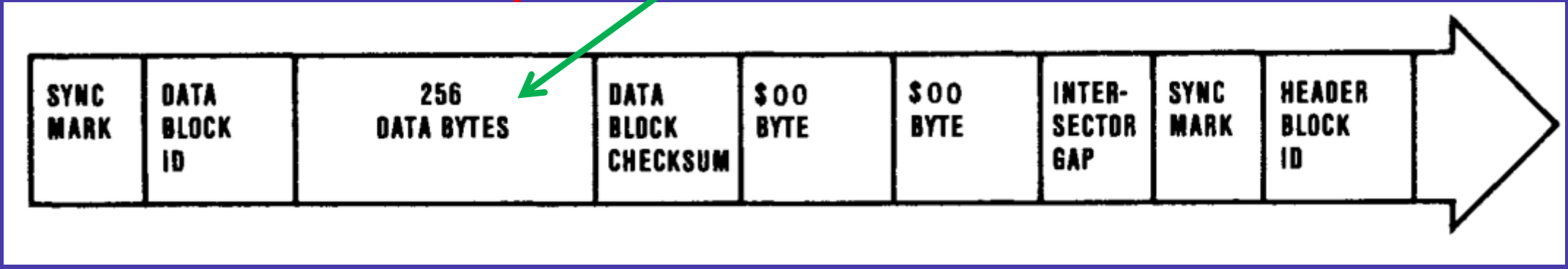


The header block is normally used to help drive identify position and read data block, and isn't in the D64. Other formats (G64, NIB) contain this, but aren't processed by Amber64 yet.

SYNC MARK	HEADER BLOCK ID	HEADER BLOCK CHECKSUM	SECTOR NUMBER	TRACK NUMBER	ID CHARACTER NUMBER 2	ID CHARACTER NUMBER 1	$OF BYTE	$OF BYTE	HEADER GAP

Sector Layout





Directory Track

- Track 18 typically contains directory information
- Start at Track 18, Sector 0 (18:0):

Bytes	Contents	Purpose
0/1	18/1	Pointer to first sector of directory entries
2	65	ASCII character A indicating 1541/4040 format
3	0	Unused
4-143		Block Availability Map (BAM)
144-159		Diskette name padded with shifted spaces
160-161	160	Shifted spaces
162-163		Diskette ID
164	160	Shifted space
165-166	50/65	DOS version and format type (2A)
167-170	160	Shifted spaces
170-255	?	Unused

Forensic Concepts (Brian Carrier, *File System Forensic Analysis*):
Essential Data – Required to be accurate to parse
Nonessential Data – Not required to be accurate

Also: space for anomalous usage

Directory Track

- BAM – Table of sectors and whether they're "free" or not.
- First example of sector chaining: First two bytes of the sector identify the next track and sector that continues the chain (file/directory/whatever)
  - First answer to "Why can't I just run strings and grep?"

Bytes	Contents	Purpose
0/1	18/1	Pointer to first sector of directory entries
2	65	ASCII character A indicating 1541/4040 format
3	0	Unused
4-143		Block Availability Map (BAM)
144-159		Diskette name padded with shifted spaces
160-161	160	Shifted spaces
162-163		Diskette ID
164	160	Shifted space
165-166	50/65	DOS version and format type (2A)
167-170	160	Shifted spaces
170-255	?	Unused

Directory Entry Blocks

Byte	Contents	Purpose
0		Track of the next directory block
1		Sector of the next directory block
2-31		File entry #1 in the directory block
32-33	0	Unused
34-63		File entry #2 in the directory block
64-65	0	Unused
66-95		File entry #3 in the directory block
96-97	0	Unused
98-127		File entry #4 in the directory block
128-129	0	Unused
130-159		File entry #5 in the directory block
160-161	0	Unused
162-191		File entry #6 in the directory block
192-193	0	Unused
194-223		File entry #7 in the directory block
224-225	0	Unused
226-255		File entry #8 in the directory block

- Note the directory chaining
- In practice these sectors/blocks aren't even contiguous (Interleaved)
  - Optimization for spinning disk
  - Another reason "Why can't I just run strings and grep?"

File Entry Format

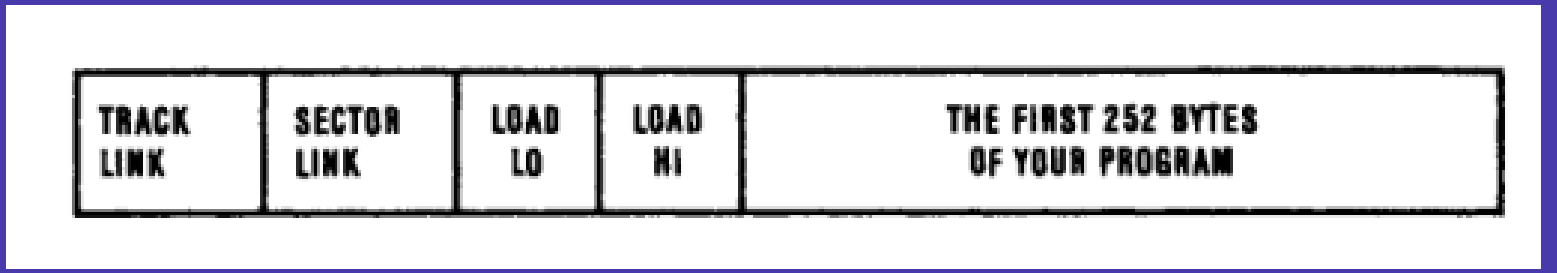
*STRUCTURE OF SINGLE DIRECTORY ENTRY

BYTE	CONTENTS	DEFINITION
0	128 + type	File type OR'ed with $80 to indicate properly closed file. TYPES: 0 = DELeted 1 = SEQential 2 = PROGram 3 = USER 4 = RELative
1—2		Track and block of 1st data block.
3—18		File name padded with shifted spaces.
19—20		Relative file only: track and block for first side sector block.
21		Relative file only: Record size.
22—25		Unused.
26—27		Track and block of replacement file when OPEN@ is in effect.
28—29		Number of blocks in file: low byte, high byte.

HEX	ASCII	FILE TYPE	DIRECTORY SHOWS
$00	0	Scratched	Does not appear
$80	128	Deleted	DEL
$81	129	Sequential	SEQ
$82	130	Program	PRG
$83	131	User	USR
$84	132	Relative	REL
$00	0	Unclosed deleted	Same as scratched
$01	1	Unclosed sequential	*SEQ
$02	2	Unclosed program	*PRG
$03	3	Unclosed user	*USR
$04	4	Unclosed relative	Cannot occur
$A0	160	Deleted @ replacement	DEL
$A1	161	Sequential @ replacement	SEQ
$A2	162	Program @ replacement	PRG
$A3	163	User @ replacement	USR
$A4	164	Relative @ replacement	Cannot occur
$C0	192	Locked deleted	DEL <
$C1	193	Locked sequential	SEQ <
$C2	194	Locked program	PRG <
$C3	195	Locked user	USR <
$C4	196	Locked relative	REL <

File Formats

PRG

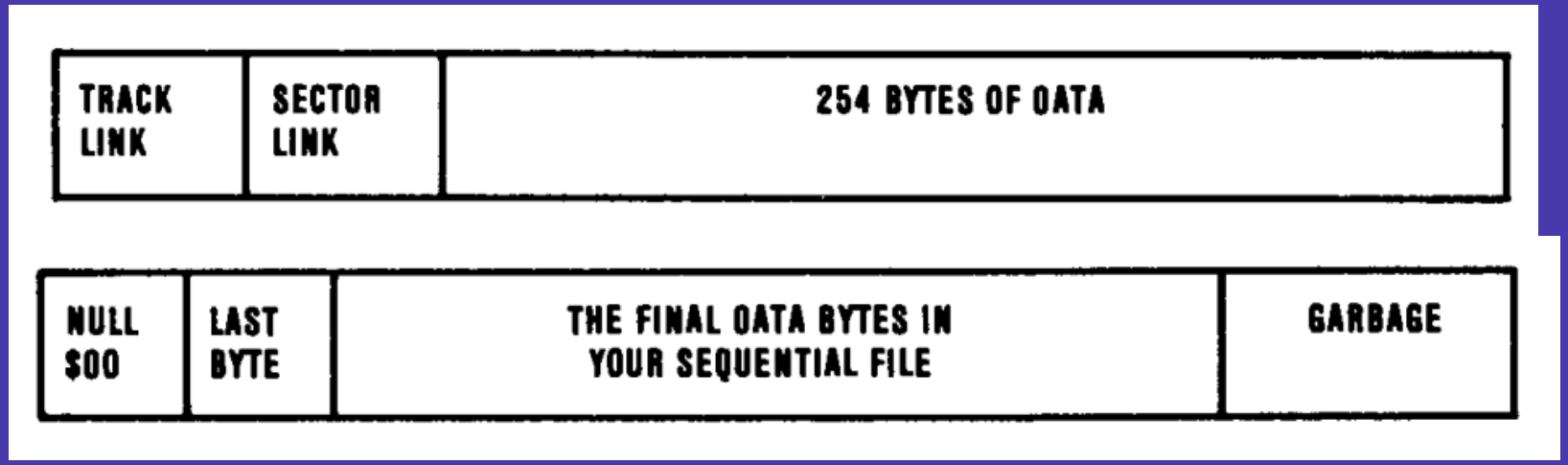


Note how a sector chain ends

Slack! Often from the 1541 RAM buffers

File Formats

SEQ



File Formats

- REL
  - More complicated and rare. Unsupported on some non-Commodore disk drives.
  - Two sets of content
    - A SEQuential file with fixed-length records
    - A “side sector” file with pointers to sectors of the sequential file to allow for fast access to specific records
  - File entry (in the directory) contains info on location of side sectors and record length
  - Amber64 can parse this in the most naïve way possible and it’s fine for indexing purposes
- USR – Program-defined, but still should have valid sector linking

Other misc. weirdness we must handle

(650,000+ disk images means you'll see every possible thing that folk would do intentionally or unintentionally to a disk)

- 40-track disks
- Invalid tracks/sector pointers (maybe someone just put normal data there)
- Sector chain loops
- Sector chains that merge into each other
- GEOS disks & filetypes
- Unexpected values in data structures with otherwise very predictable values
- **TEXT ENCODING**

On Text Encoding

- Commodore PETSCII isn’t exactly ASCII.
  - Two separate “fonts” that can be switched out, “shifted” and “unshifted”
    - This is system state, not stored with the text
    - Given a stream of bytes it’s difficult to tell programmatically which font it’s meant to be rendered in—a lot of times the case in our parsing will be off
  - Uppercase ASCII text lines up with “unshifted” PETSCII uppercase or “shifted” PETSCII lower case.
  - Lowercase ASCII text lines up with “shifted” PETSCII uppercase.





https://en.wikipedia.org/wiki/PETSCII#/media/File:C64_Petscii_Charts.png

On Text Encoding

- In video memory, “Screen codes” are used instead of PETSCII or ASCII.
- More ranges for what we’d like to index as printable text
- Separate ranges for plain and reverse-color
- Amber64 has to normalize all of the PETSCII and Screen Codes to ASCII.

Screen code (dec, hex; reverse dec, hex)				Character (up/gfx, lo/up)		Screen code (dec, hex; reverse dec, hex)				Character (up/gfx, lo/up)	
0	$00	128	$80	@		64	$40	192	$C0	─	
1	$01	129	$81	A	a	65	$41	193	$C1	♠	A
2	$02	130	$82	B	b	66	$42	194	$C2	┃	B
3	$03	131	$83	C	c	67	$43	195	$C3	─	C

Amber64

- amber64.py contains code for processing all of the previously described data structures in a disk (and more)
- Creates JSON data suitable for Elasticsearch indexing and searching
  - Can also create a bit nicer of a nested structure with little modification

Amber64 Output

- Image name
- SHA256
- Disk Name/ID/Type
- Files

- Strings from
  - Chains
    - Contents of files
    - Directory chains
    - Chains not associated with files
      - Allocated space
      - Free space
  - Slack space from end of chains
  - Naïve “strings” across image
  - SHA256 hashes for everything

Using Amber64

- Set up Elasticsearch, the local dev quick start is fine
  - https://www.elastic.co/docs/deploy-manage/deploy/self-managed/local-development-installation-quickstart
- Ingesting .D64
  - Set up a directory of .d64 files
  - Change es_url and api_key in initialize.py and ingest.py to match up with your instance
  - ./initialize.py to set up the index
  - ./ingest.py <dir of d64s>
  - To do it faster, split up your input directories and run multiple instances of ingest.py
  - Processed d64s are moved to store/
- Query it with your favorite Elasticsearch frontend

Using Amber64

- DirMaster – Excellent tool for interactively exploring a D64 image
- Emulation
  - Make a working copy of the disk image you're about to load (many games and applications make changes to their own disk)
  - VICE - https://vice-emu.sourceforge.io/
- Real Hardware
  - Be comfortable with through-hole desoldering/soldering
  - Modern replacements of proprietary chips are available
  - Keyboard will likely need a complete teardown, cleaning of carbon stems, lube
  - Ultimate 1541-II: Cartridge-based emulation of the disk drive, modem, and a lot more. Network accessible.

CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!
CRAZY FINDINGS TIME!

DISCLAIMER: Any phone numbers are likely owned now owned by completely different, non-80s hacking scene people that have NO IDEA what you're going on about

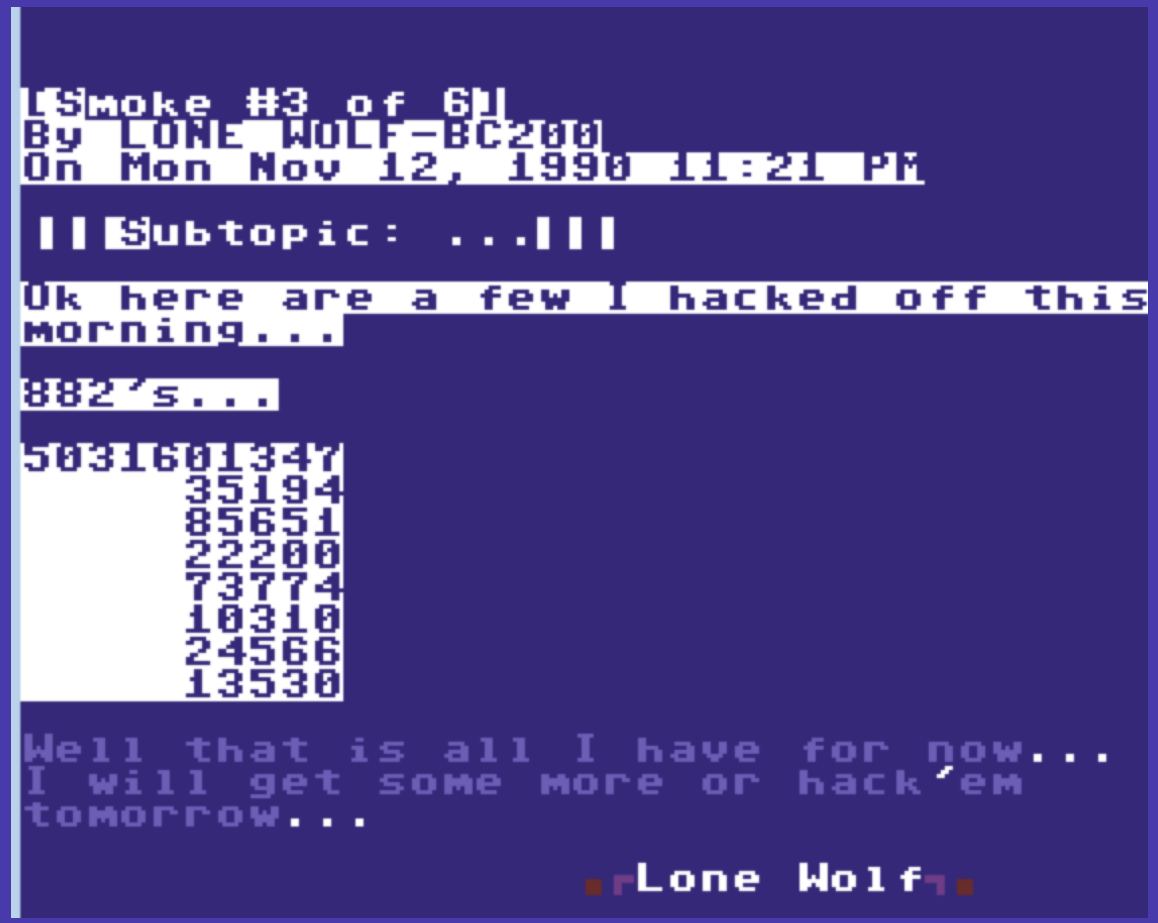
Lost Games

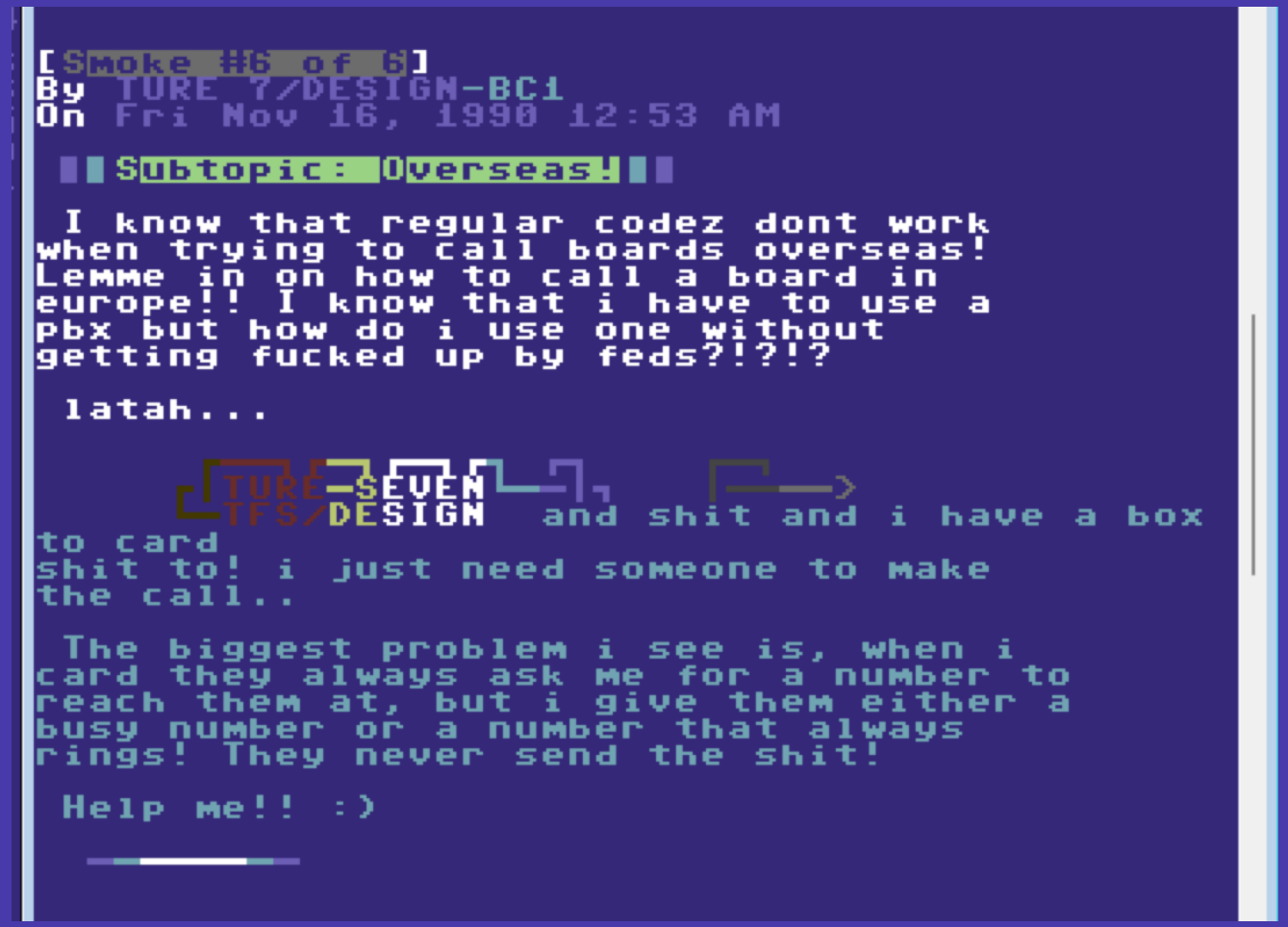
- Assisted researcher Charlotte Cortois
  - PhD Candidate, Université de Montréal
  - History of pornographic video games, among other very interesting work
  - https://miramar.itch.io/
- Text-based “Sex Quiz” games for parties
  - Found a pirated-but-not-obfuscated copy of “Ladies’” version, suitable for study/editing
  - Found references to the lost “Men’s” version
- In the process, found “Asteroid Miner”, not previously identified in current game collections



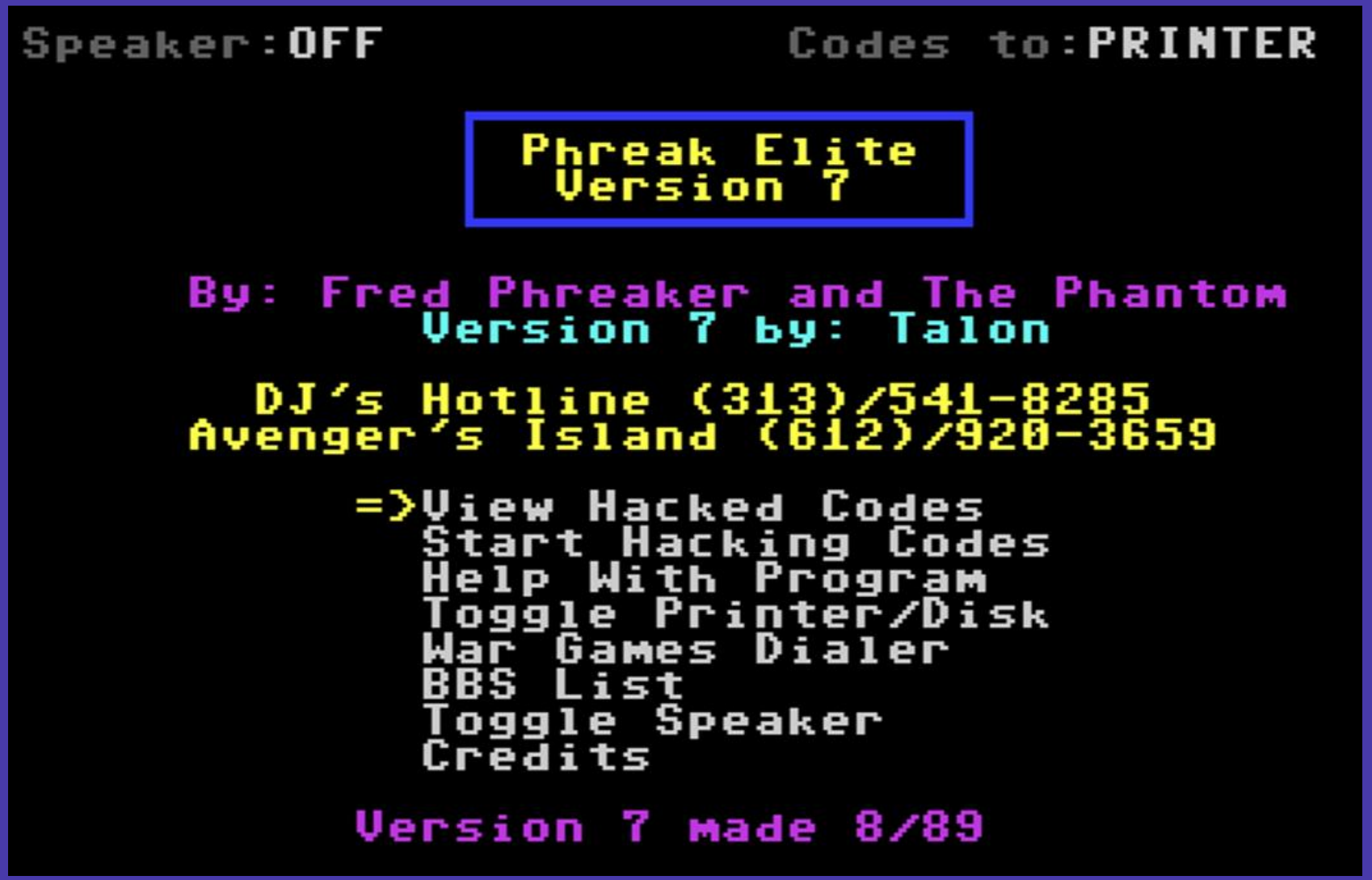
Compute Gazette, July 1983, P229

Codes and Cards





Codes and Cards



Codes and Cards

Description of Commands

Dialup Number The number of the long-distance company that you will use.

Start Code When hacking SEQ, this number is the first code that will be tried. When hacking RAN, the length of this number determines the length of the random code.

End Code Used in conjunction with above, this will be the last code that will be tried.

Test Number A number that will ALWAYS give a 1200 baud carrier when it answers. The number given WILL work.

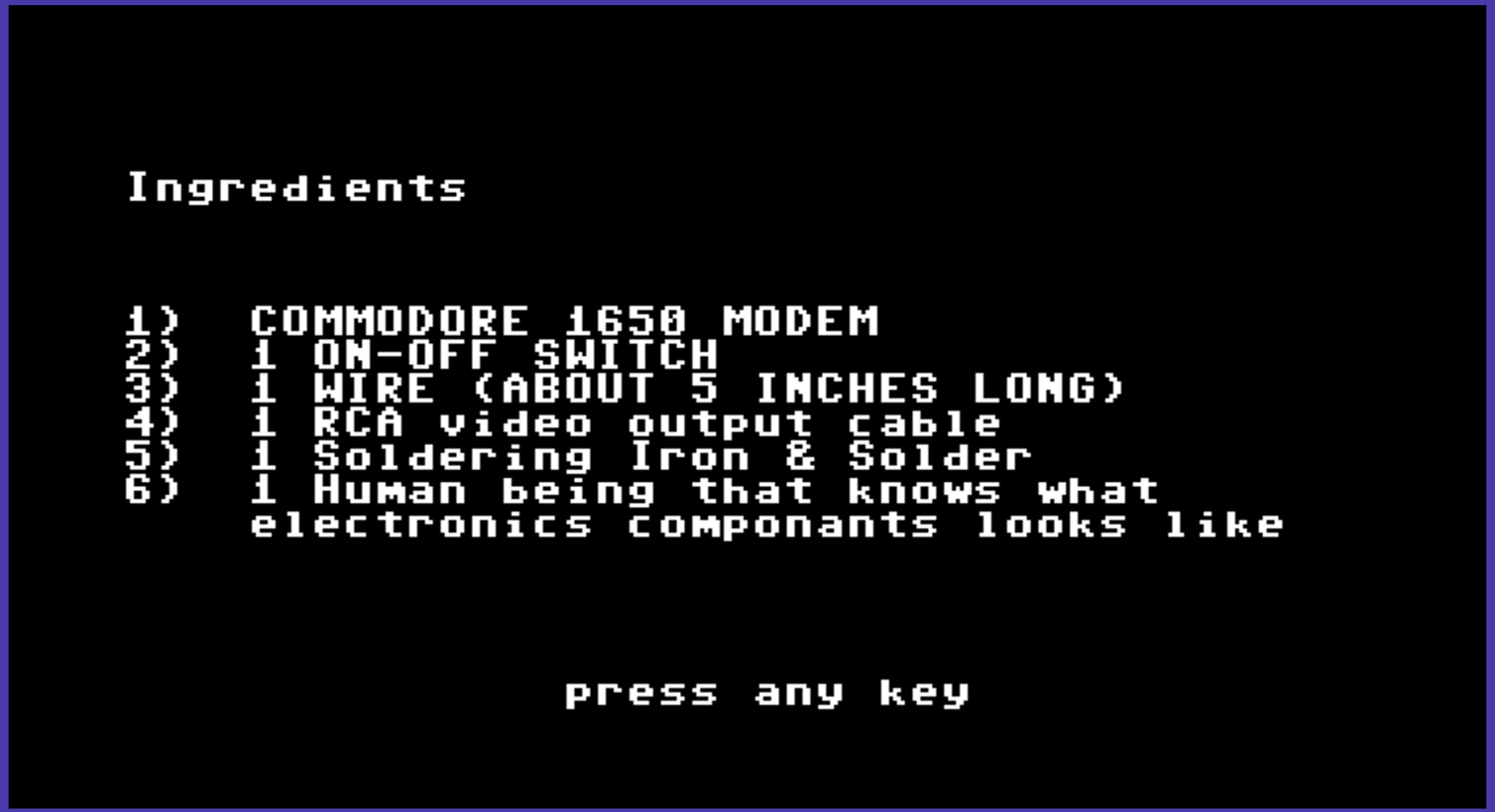
Commas Before The number of two-second 'pauses' that should exist between dialing the Dialup Number and the code/Test Number.

Hit [RETURN]

Phone Phreaking



Wiring up your SID chip and modem



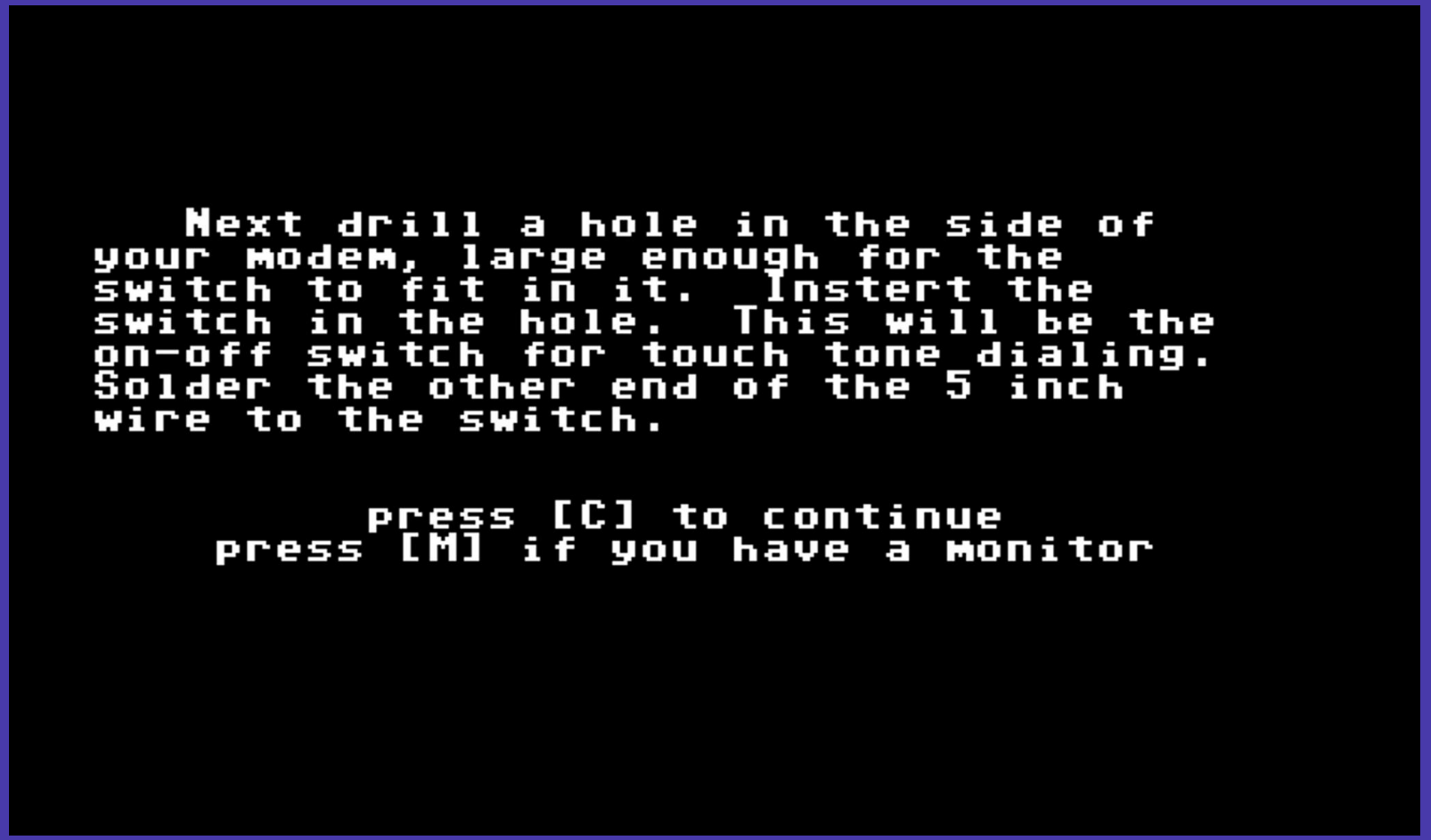
Wiring up your SID chip and modem

First, open your 1650 modem. Second, you must connect the 5 inch wire to a resistor. Here are some steps to find that resistor.

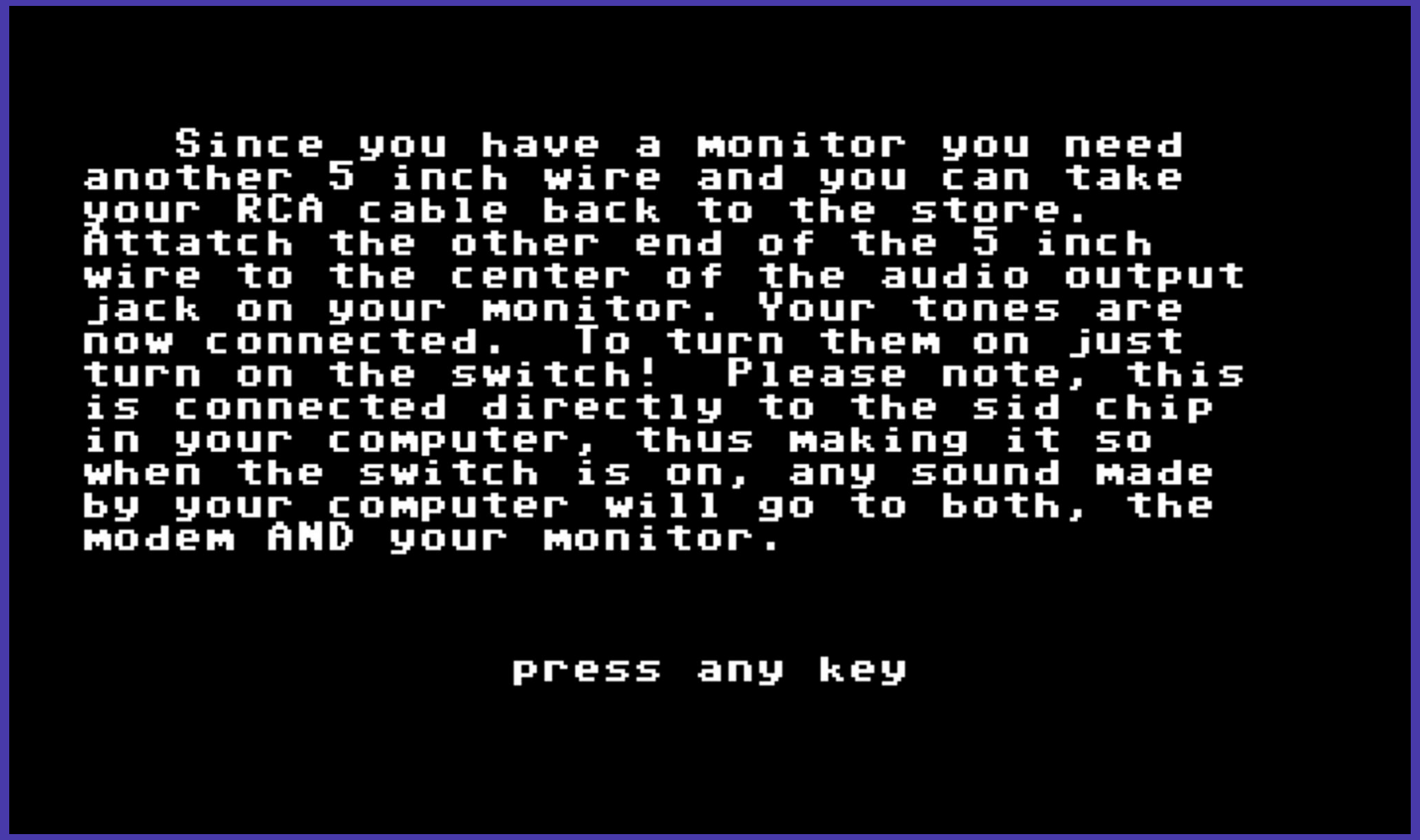
Find the crystal in the modem. Above this crystal their is a diode. Directly to the left of the diode is a metal wire. Directly to the left of this metal wire is a chip. It is small in size, compared to the rest of the chips in the modem. On the lower left edge of the chip, is a resistor. This resistor is Directly above a big capasitor. Solder the 5 inch wire on the right edge of the resistor.

press any key

Wiring up your SID chip and modem



Wiring up your SID chip and modem

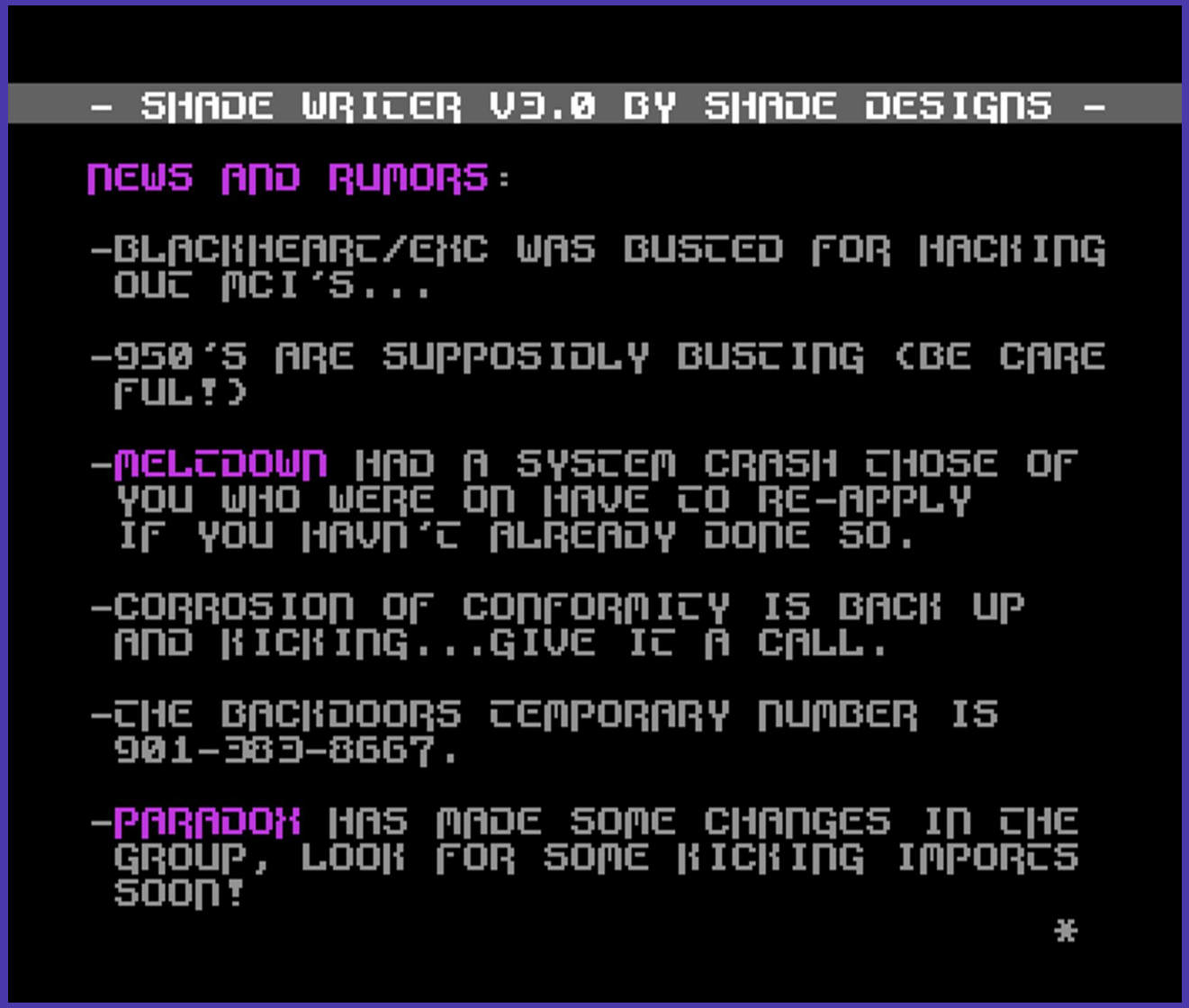


Wiring up your SID chip and modem

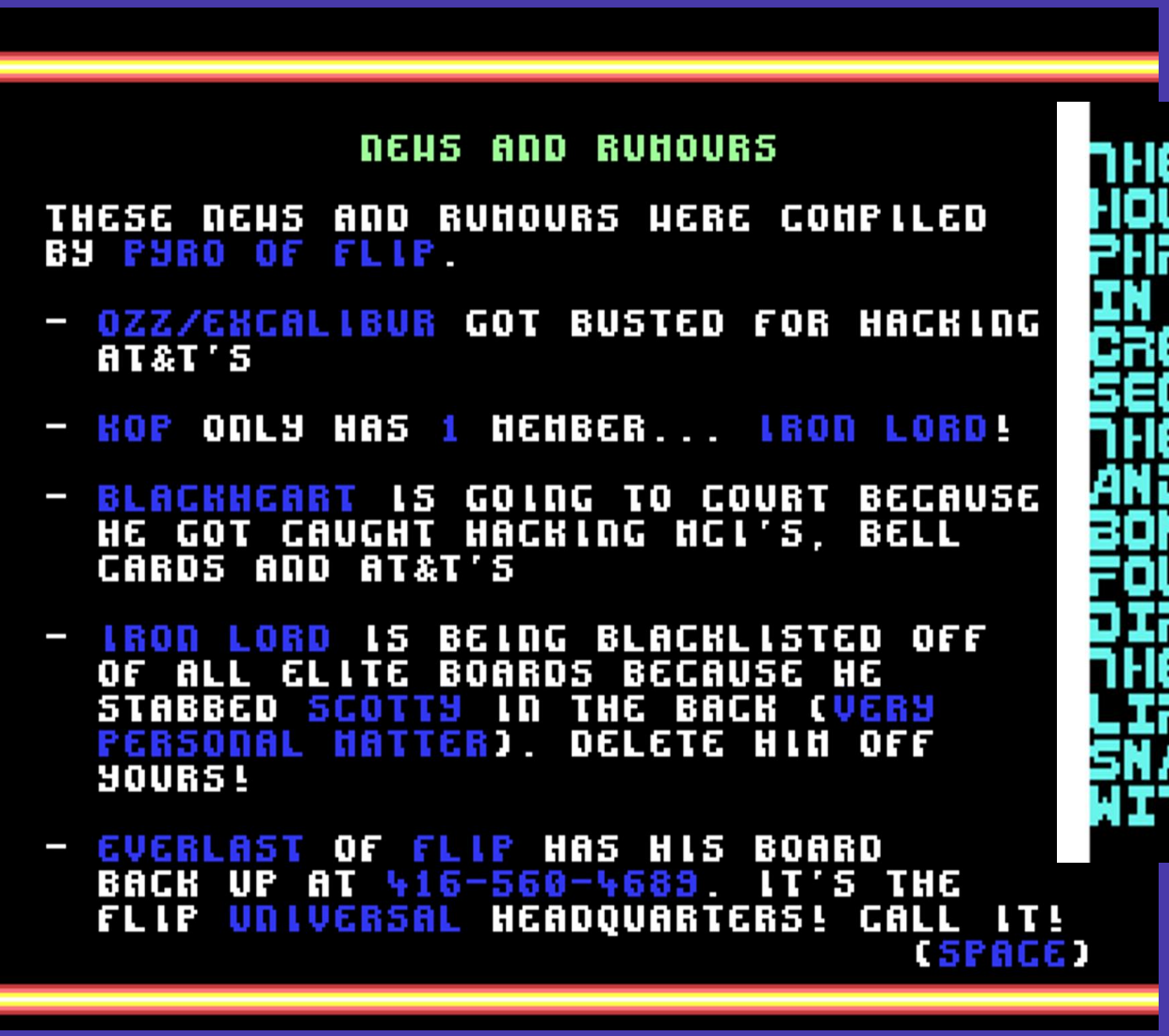
Next, strip the end of the RCA cable that hooks up to a stereo and connect the BLACK wire to the other end of the switch. Your tones are now connected! Now, to turn on the tones just turn on the switch! Please note these tones come from the sid chip in your computer, so by turning on the switch and playing something out of your computer you can hear it through your modem too!

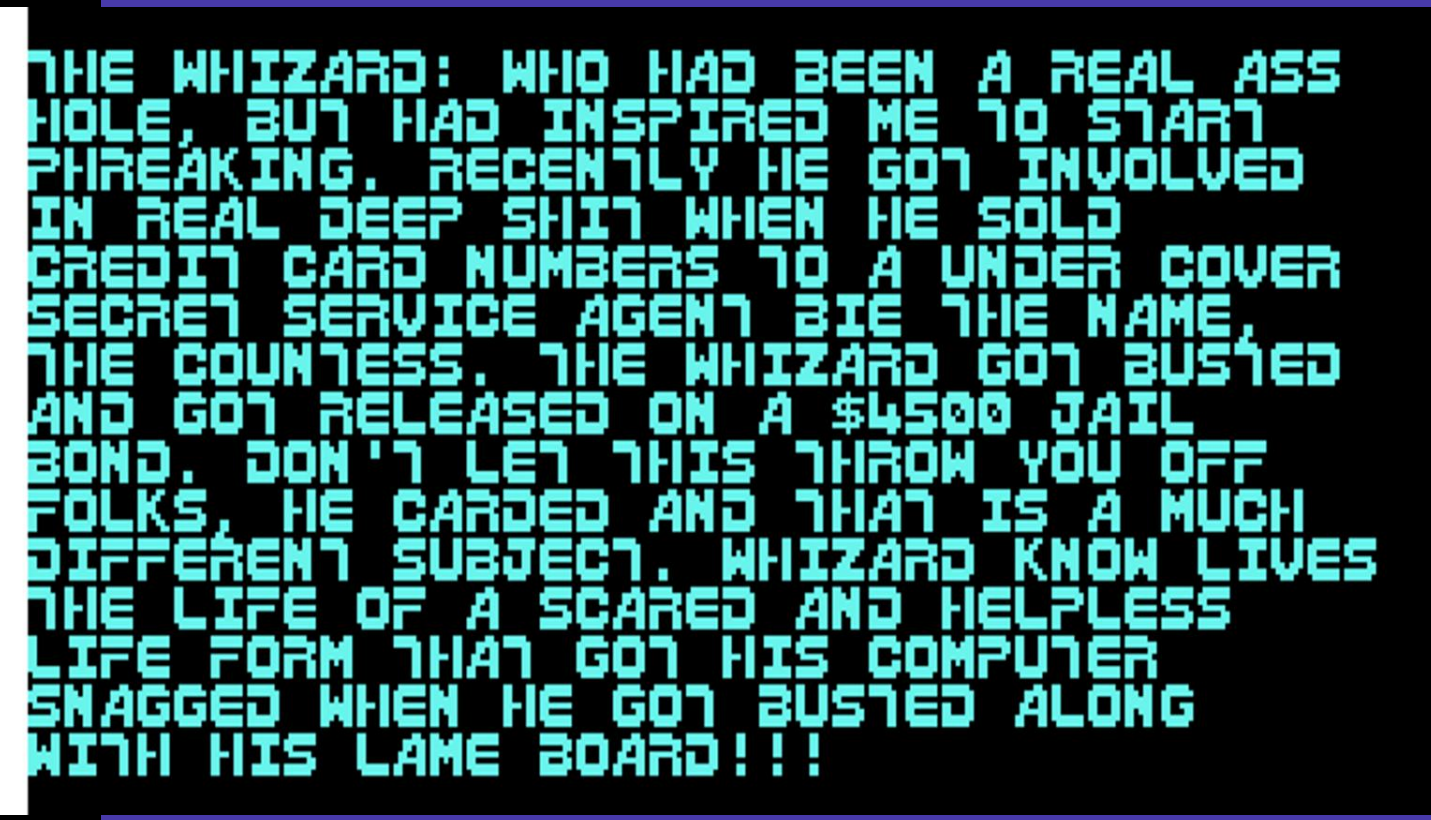
press any key

Getting Busted



Getting Busted

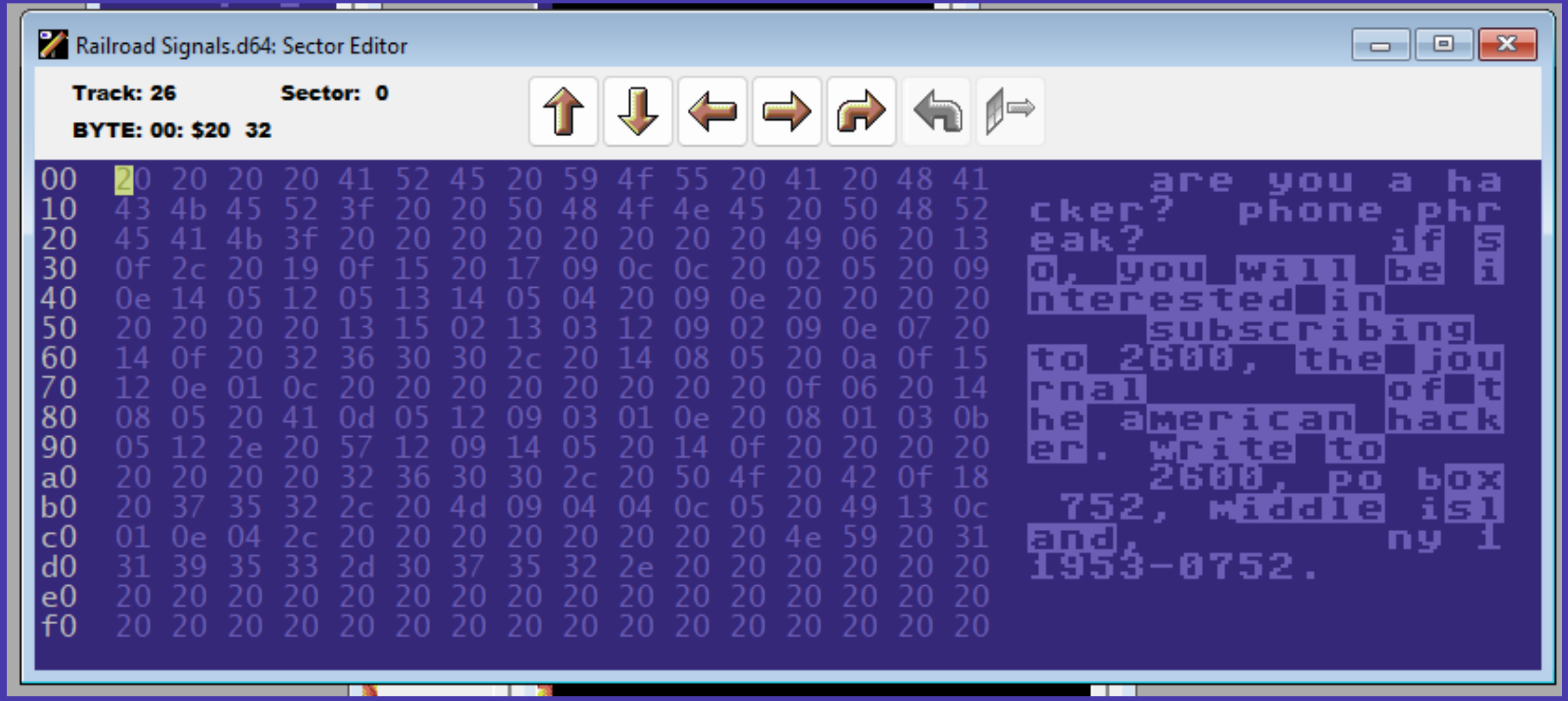


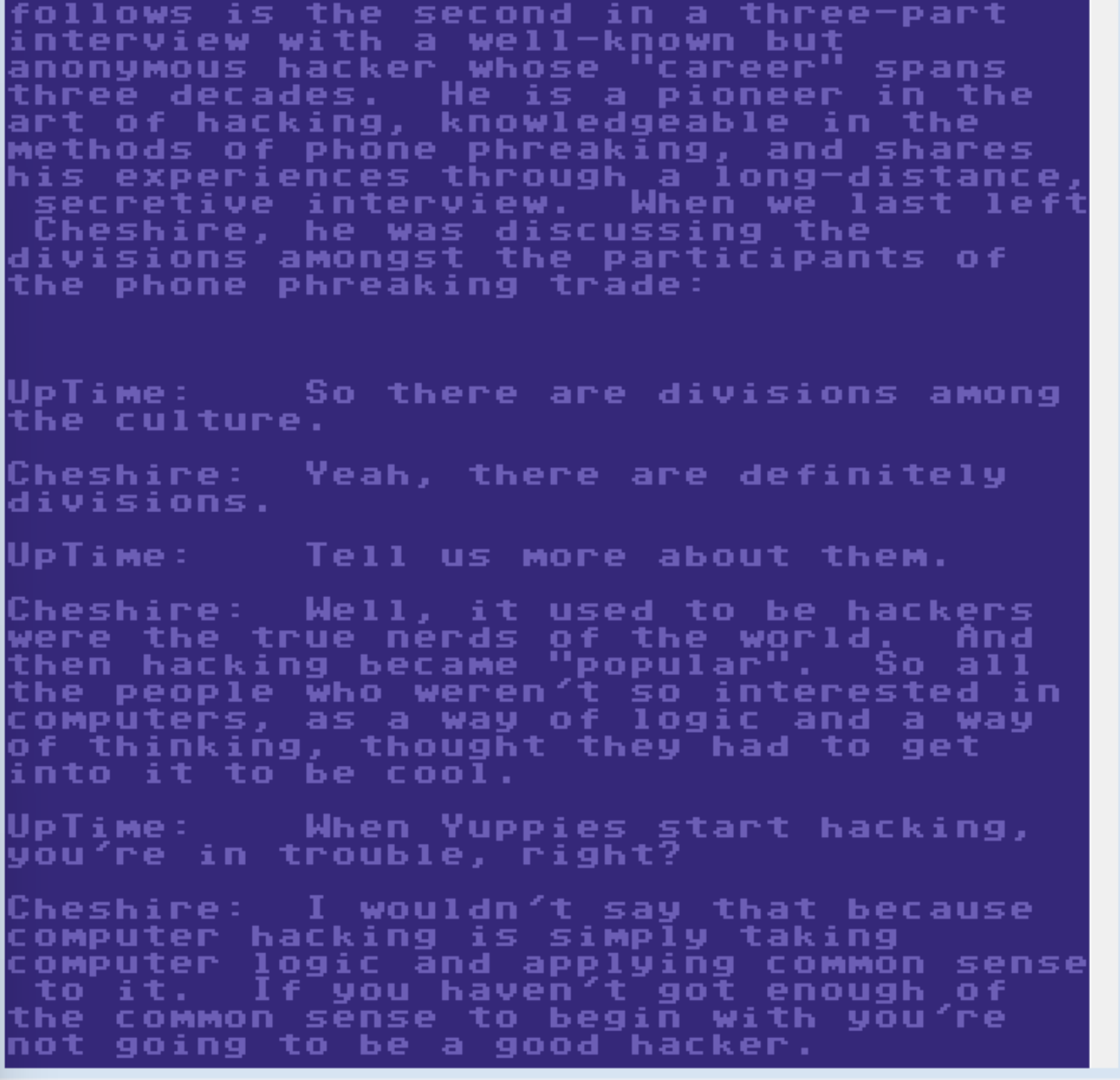


Applying to traffic in warez


```
    "string": "highlander (413-538-5864)"
},
{
    "position": 1509,
    "string": " 300\nLogon at  11:06 PM\nMessage from highlander (413-538-5864)\nTues.,
    3/28  11:08 PM\n\n\nHello, yes I do realize that I am\n300 baud, however I am ordering
    my\n2400 buad this weekend, At the moment\nbecause of my baud rate, I am unable\nto
    keep up with transfers. However\nupon receipt of my 2400 and the \nsoon to be rental of
    a P.C Pursuit\naccount, I can assure you that I\nwill be able to keep up with the\n
    transfers.\nThank You,\nThe Immortal Highlander\nLogoff at  11:12 PM\nConnect time"
},
{
    "position": 1991,
    "string": " minutes.\n\n"
},
```

A Very Obscure Ad for 2600 Magazine





follows is the second in a three-part interview with a well-known but anonymous hacker whose "career" spans three decades. He is a pioneer in the art of hacking, knowledgeable in the methods of phone phreaking, and shares his experiences through a long-distance, secretive interview. When we last left Cheshire, he was discussing the divisions amongst the participants of the phone phreaking trade:

UpTime: So there are divisions among the culture.

Cheshire: Yeah, there are definitely divisions.

UpTime: Tell us more about them.

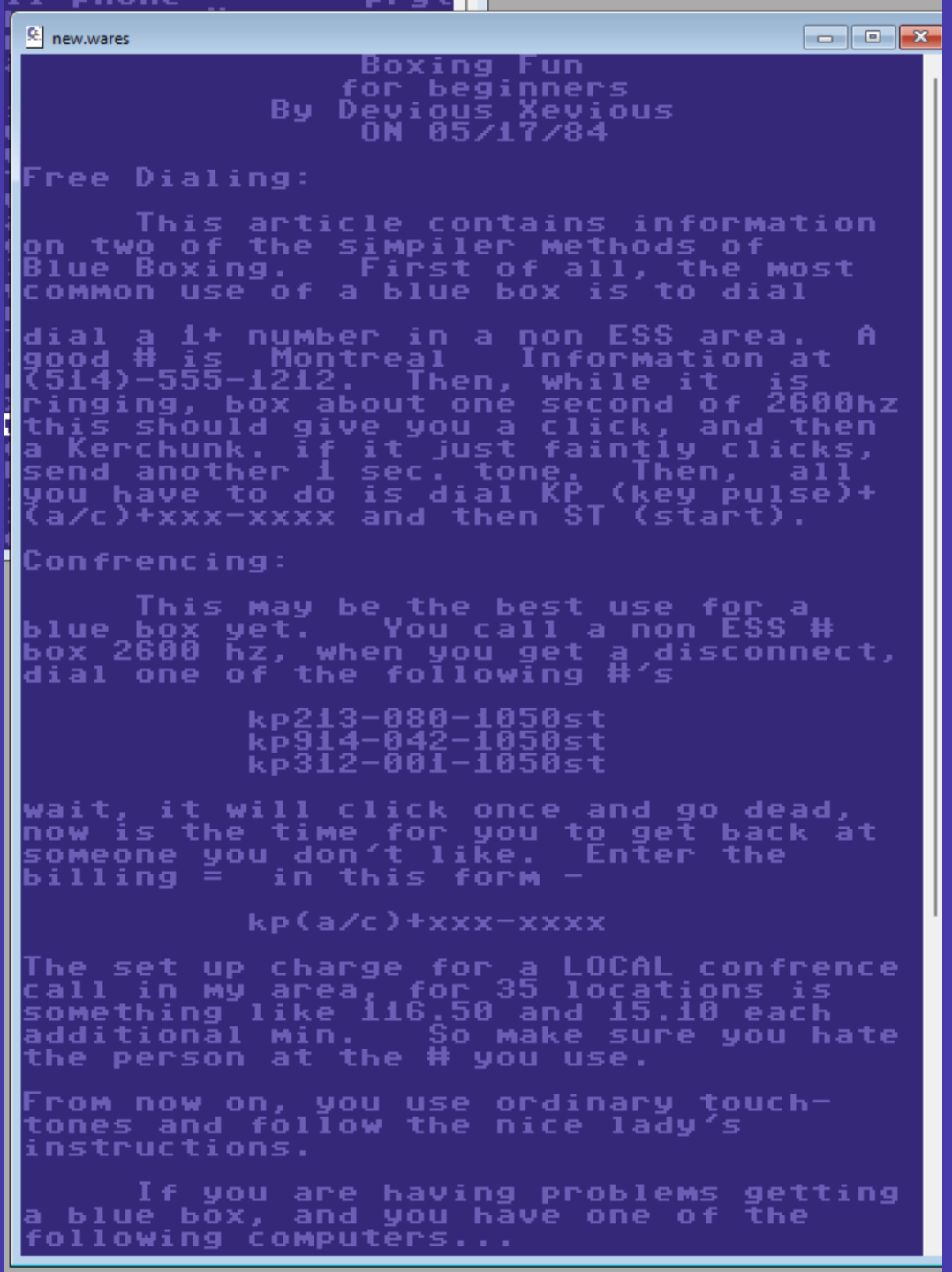
Cheshire: Well, it used to be hackers were the true nerds of the world. And then hacking became "popular". So all the people who weren't so interested in computers, as a way of logic and a way of thinking, thought they had to get into it to be cool.

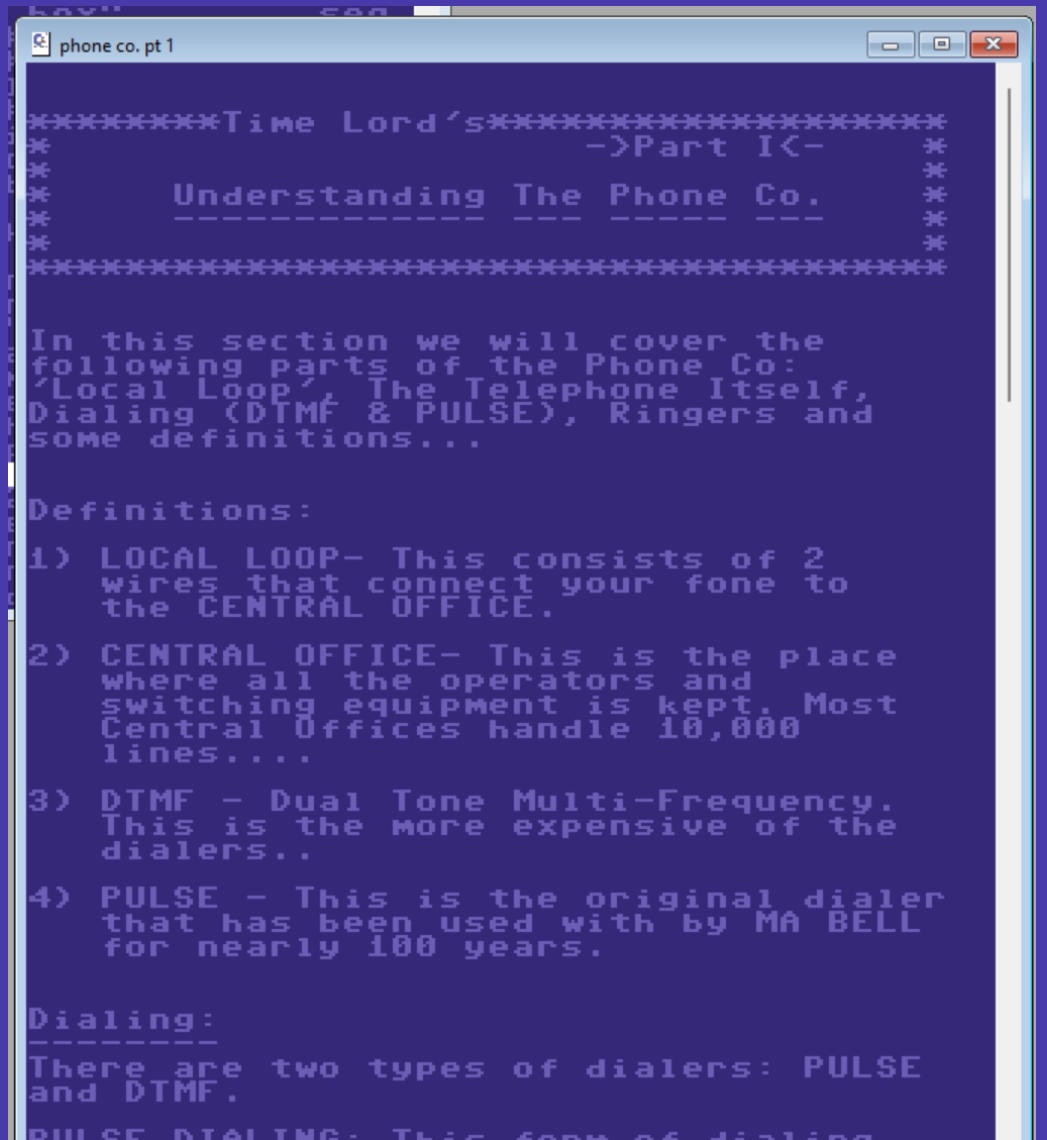
UpTime: When Yuppies start hacking, you're in trouble, right?

Cheshire: I wouldn't say that because computer hacking is simply taking computer logic and applying common sense to it. If you haven't got enough of the common sense to begin with you're not going to be a good hacker.

A 1987 interview with
Cheshire Catalyst
(Editor of TAP Newsletter,
HOPE conference, etc.)
in “UpTime”, a disk magazine.

Text files I haven’t seen on modern archives





Conclusions

- Amber64 allows us to query a very large set of vintage floppy disk images comprehensively and quickly, through:
  - …understanding the details of 1541 disk layout and file system data structures
  - …normalizing potential text of interest into ASCII
  - …creating a JSON data structure representing the structure of data on a disk and indexable text
  - …leveraging an easy-to-use engine for indexed search (Elasticsearch)
- Enables research for others with interesting questions about this era of computing

Conclusions

- Future work
  - Identifying topics of interest (contact me, I'm happy to run queries on your old handles, BBS, etc.)
  - More comprehensive research, extraction, documentation
  - Other formats (1571, 1581, PRG, Cartridge, Tape)
  - Other platforms (Amiga, PC, etc.)

Contact

Dr. Wesley McGrew
Senior Cyber Fellow
MartinFed
w.mcgrew@martinfed.com

X: @McGrewSecurity
Personal Site: http://mcgrewsecurity.com

martinfed