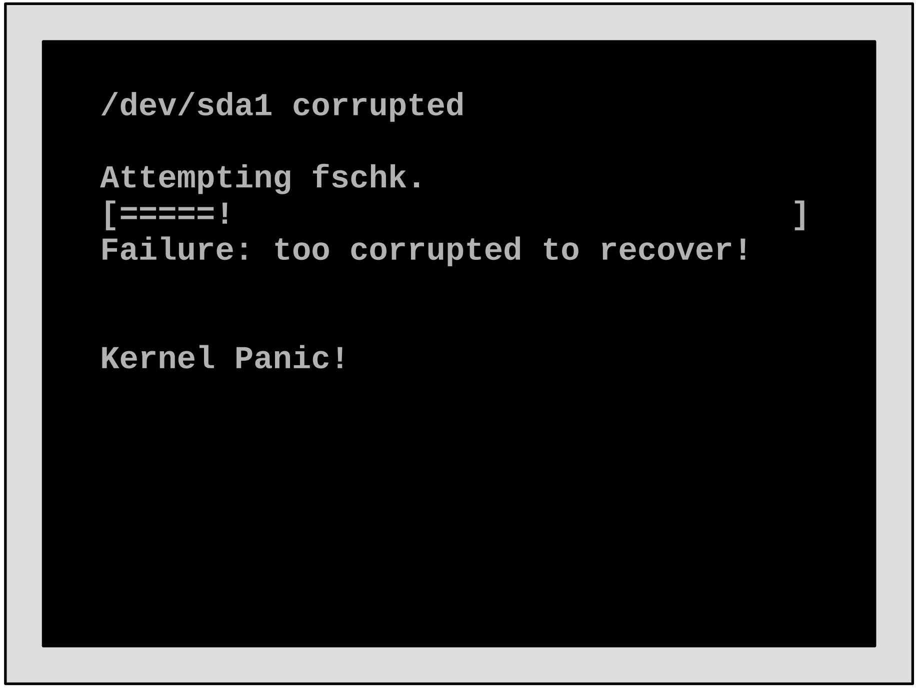
Protecting Your Files Against Bitrot And Other Corruption And Loss

Dr. Freja Nordsiek (GWDG AG-C)

1 month before Ph.D thesis deadline



NOOOOOOO!!!!

All my data is gone!!!

Ways Data Is Lost

Single file loss

- Bitrot inside file
- Accidental deletion
- Accidental overwrite
- Loss of software to read file

Losing many files at once

- Storage media failure
  - Laptop HDDs often fail in the first 2 years
- Losing storage media
  - Where did I store that USB stick again?
- Computer destruction
- Building destruction
  - Fire, flooding, etc.
- Don’t remember harddrive decryption key
  - Get back from 1 month vacation and forgot
- Malice
  - Ransomware
  - Sabotage (wipers, etc.)
  - Disgruntled co-workers, etc.

Human Storage Technology Robustness

Robust

Stone/glass (carved/etched)	millennia
Paper (alkaline)	centuries
Paper (acidic)	century
Tape drive	30 years
SSD	3-10 years
HDD	2-5 years
DVD/CD-RW	2-5 years
Floppy disk	do you still have a floppy drive?
Zip disk	do you still have a zip drive? the right kind?
Sticky notes	at most, till your desk is cleaned
RAM	seconds to days

USB sticks

Fragile

Bitrot and Other File Corruption

Bitrot - Steady accumulation of errors at the bit level in storage media

- The bigger the file, the less time it takes for the first bitrot to occur

Other Corruption

- Accidentally start overwriting data
- Accidentally remove end or beginning of file
- Errors when transferring files from one system to another

Corruption in the right place can destroy much or all of the rest of the file with some formats

- Compressed formats are particularly vulnerable
- Corruption of the beginning of the file is particularly damaging

Bitrot and Other File Corruption

Checksums to detect corruption

create: `sha256sum FILES > SHA256SUMS`

check: `sha256sum -c SHA256SUMS`

- SHA2 family (sha256 and sha512) are fast and available on most systems
- MD5 and SHA1 should be avoided
  - Malicious person could, at effort, change your file and the checksum would stay the same

Use file format's built in capabilities to detect bitrot and transport corruption
- HDF5 (and therefore NetCDF4) files allow fletcher32 in Dataset chunks
- ZIP files include a simple checksum (CRC) for every file

Use formats where corruption in one location damages less other parts of the file
- 1 damaged byte in an MJPEG file will cause at most two images to be damaged/lost
- xz compressed files are more vulnerable than lzip files to losing all data after damaged byte

Bitrot and Other File Corruption

Parity/repair data

- Error Correction Codes
- Like ECC RAM, but for files
- Some redundancy to repair corruption

PAR2 - https://github.com/Parchive/par2cmdline

- Parity/repair archive that can be made for a single file
- Can be used to detect corruption like a checksum
- Can be used to repair corruption or even small chunks deleted in the middle
- User chooses the level of redundancy
- Is slow, unfortunately, and makes several files

create (10% redundancy):	`par2 create -r10 FILENAME`
verify:	`par2 verify FILENAME.*par2`
repair:	`par2 repair FILENAME.*par2`

Protection Against Media Loss

Things that help, but are not completely sufficient

- More robust media
- Media-level redundancy
  - RAID 1, 5, 6
  - Note, it is common for more than one hard drive in a RAID array to fail around the same time
  - Does not protect against computer/building destruction

BACKUP !!!

Have more than one copy of your data stored in different locations

- Different media in different computers is good
- One backup that is in a completely different location is best
  - Different locations in same building is OK – building fire destroys all copies
  - Different nearby buildings is good – one bad fire could destroy all copies
  - Buildings in completely different areas of a city or different cities is best

Make sure your backups are actually intact periodically

Remember, a hacker might be waiting for a backup external drive to be plugged in to wipe/encrypt it too

What to Backup And How

What to backup?

1) Irreplaceable data (e.g. experimental data, software codes)
2) Difficult to replace data (e.g. digitization of all documents in a file cabinet)
3) Data that is expensive or time consuming to regenerate from 1 and 2

Data that can be easily regenerated from 1, 2, and 3 isn’t critical to backup, but is still recommended

How:

- External hard drive with a recent copy and another with an older copy in different safe places
- If it is small enough, cloud backup
- Backup/archival services (e.g. tape library)
- Printing a reduced/summary dataset to paper
  - For example, tables of some summarized form of your data that you could write your paper or thesis with in the event your raw data was lost for good
- Useful GWDG services:
  - OwnCloud (general files)
  - Gitlab (code and other text files)
  - ShareLaTeX (LaTeX documents)
  - Tape library

Loss of Software to Read Files

Software comes and goes, and updates
- NASA lost many images since the format documentation and software were lost
- Updated software sometimes loses the ability to read software made by older versions
- Software company goes bankrupt and there is no documentation on the file format

Mitigation:
- Best formats and codecs (encoding within format):
  - publicly available specifications or completely reverse engineered with documentation
  - more than one independently made codes to read them
- Next best:
  - Vendor of the software supplied you with well written source code
  - Format/codec is straightforward to reverse engineer (some plain text based formats)
- Store data in multiple formats/codecs coming from independent projects and vendors
  - Often, there is a standardized format that is not as fancy or widely used
    - RTF or ODT in addition to DOCX
    - PNG in addition to special proprietary image format
    - VP9 or FFV1 codec in addition to special proprietary video codec

Very useful resource: https://www.loc.gov/preservation/resources/rfs
https://www.loc.gov/preservation/digital/formats/intro/intro.shtml