Reliability and the Software Developer

Sh*t is going to hit the fan, not if, but when.
A problem has been detected and Windows has been shut down to prevent damage to your computer.

The problem seems to be caused by the following file: SPCMDCON.SYS

PAGE_FAULT_IN_NONPAGED_AREA

If this is the first time you've seen this stop error screen, restart your computer. If this screen appears again, follow these steps:

Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any Windows updates you might need.

If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.

Technical information:

*** STOP: 0x00000050 (0xFD3094C2,0x00000001,0xFBFE7617,0x00000000)

*** SPCMDCON.SYS - Address FBFE7617 base at FBFE5000, DateStamp 3d6dd67c
Nothing is as it seems

- Hardware breaks
- Software is buggy
- Distributed system unhinges
- ...but not all hope is lost
Disk

• We’ve discussed how to handle problems in the FS since operations can be interrupted (e.g. crash)

• Lower level – what if your disk is broken?
  • Journaling and other FS mechanisms to salvage
  • RAID and other redundancy for sector/disk failures
  • Individual sectors also have checksum on disk
Disk – Many Moving Parts

- Multiple platters thinly coated with magnetic materials
- Spinning at 1000s RPM
- Arms move at high speed
- Writing head is tiny – and "floats" on top of the platted
- All potential sources for problems
Disk

• Disks try to save as much as possible - sectors have checksums that is verified upon access

• Redundant sectors for backup

• Most of it is transparent to users (maintains LBA interface)

• Disk might be problematic but not a complete loss
  • Some reads/writes are slow, ticking noise, ...
Disk – S.M.A.R.T.

• Standard* hard drive self-reporting of internal statistics
• Read errors, relocated sectors, power-on hours...
• Many tools for all OSs to decipher
• Unfortunately a lot of fields are vendor-specific and depends on how good the tool is at interpreting
• But still is a good starting point to understand the state of the disk, and give early warnings before data loss!
Disk – File Systems

• Additionally, other FS features to help avoid data loss in the face of faulty disks
  • e.g. redundant superblocks
• Additional checksums (not the hard drive)
  • ZFS, Btrfs
• Scrubbing – background process that slowly read and verify every drive sector
• RAID
• Takeaway – redundancy at multiple levels
Memory

- Array of individual cells
- Capacitor stores charge
  - Full = 1, empty = 0
  - Like water buckets
- Again, many things can go wrong
  - Charge can be lost
  - Wires can short
  - Material can weaken
  - Cosmic rays?
- At the nm, microscopic interference = catastrophe!
Memory – ECC

• Storing additional parity in spare chip
• Correctional power depending on algorithm and additional space – can even operate with entire failed chips
• $$$ - typically reserved for servers
Memory – Page Retirement

• Hardware can supply hints when a particular page of memory is bad (ECC errors, whether correctable or not)

• OS can take action and remove the page frame from the allocator
  • Never given out to processes

• Small space tradeoff for maintaining stability
  • Or buy time to save work/gracefully shut down
Many more examples

• The network stack – failed equipment, loose cables
• Not even CPUs are safe – Pentium FDIV bug (1994), TSX bug (2014!)
  • How do you “patch” CPUs?
• Power supply failures, cooling system failures, fire suppression failures, ...raccoons?
• The real world is analog and messy
Regardless of system type

...no one is immune
Distributed Systems

• Modern systems are impossibly complex
• Single Facebook page load can hit 100s, even 1000s of machines
• Good developer has to have some idea of all the components (and how they can fail)
Distributed Systems

• As we build larger and larger systems, almost always guaranteed to encounter failures

• Source can be hardware or software

• If a single machine have $x\%$ chance of failure per hour, what are the chances that I can complete my [_____] that takes $y$ hours?

• Pile on network failures, environmental and geopolitical factors

• How does anyone get any useful work done?
scenes from distributed systems

a "linearizable" system
(like etcd)
I can has answer?
NO, can't you see we're having a leader election?

an eventually consistent system
{like DNS}
what's the IP for julia.com
right
1.1.1.1
2.2.2.2
... 2 hours later...
... um, 1.1.1.1
(you can always get an answer though!)

replication is hard
just copying data from my primary
OMG
hate to tell you but uh the primary changed 5 minutes ago

the network is fine BUT


clocks lie
it's 5pm
it's 5pm
2 minutes later...
5:02
5:17

with 1000s of machines... it gets weird

https://jvns.ca
Distributed Systems

• Consistency – which copy is the right copy? When is it available to different users?
• Clocks – machines don’t tick at the same rate
• Unreliable network – How do I proceed when all messages are dropped?
• Global order – who did what, and when?
• Failed nodes, or worse, malicious nodes
• Capacity provisioning is a dark art
• Cascading errors
Distributed Systems

• Replicas – multiple copies of the same data
  • Or multiple workers doing the same work
  • What happens for reads? Writes?
  • How do we synchronize data?

• Checkpoint / snapshot
  • Save entire state of the system at one point in time
  • Costly – can only be done infrequently
  • Mostly in tightly-coupled systems such as HPC

• Other concerns
  • Stragglers, timeouts, etc.
Distributed Systems

• Availability – multiple data centres located across the globe, with dedicated connections
  • Which is closest to the user?
  • Which country to service from?
    • Regulatory requirements?
• Caching and CDNs
• DDOS and DNS attacks
Unexpected Sources

• The OS kernel and libraries
  • Yes, you may have to debug those too
• Workloads can run inside VM, containers, etc.
  • How about their reliability (&performance?)
• Other things you didn’t know existed
• ...and your own mistakes...
  • More often than you think
Still not perfect
The Developer (You)

- Understand all parts of the stack, hardware + software
- Know your assumptions
- Don't have to check ALL errors, but be able to identify and locate problems when they occur
  - Befriend tracing and debugging tools!
  - The ability to spot problems is as important as writing code
- Build systems that fails early, fail often
- Failure is not the exception, it’s the common case
- Prepare and practice for it

- Lots to learn (CSC469 and beyond), even more to research!