WeGet letters

by Michael W Lucas

Oh! Incorruptible Single Source of Truth,

FreeBSD 13 tears me between lust and loathing. I need the improvements, but I have this superstitious fear that number 13 is unlucky. I know it's irrational, but that superstition couldn't have endured for centuries without there being some truth to it, could it?

Tell me everything will be okay.

—Hesitant Upgrader Geek

HUG,

What? Sorry, I was thinking of something else.

Don't feel insulted. You're not special. It's what I do. Think of other things, that is. Not apologize. Saying "sorry" isn't an apology, it's a statement that I am vaguely aware that you might get all emotional at me and the annoying screech of your tantrum would interfere with my digestion. An apology includes a recapitulation of your actions, an acknowledgement those actions harmed others, a statement of regret, and a query as to how you can compensate others for the damage you've inflicted. All four long-time readers of this column immediately comprehend that's all far too much work for me.

If you want an apology, you'll need to make it yourself.

And isn't that why we work with machines? The machine has no feelings and doesn't care about yours. Those 32-bit timers roll over and crash the system without regard for your spouse giving birth or the new Star Trek's release date or you giving birth. The machine treats us with undifferentiated indifference. So many sysadmins want to treat other humans with that same indifference, but all too often devolve into thoughtless, reflexive contempt.

You aren't equipped for indifference. The chunk of electrified fat occupying your skull called "you" (whatever that means) evolved for caring. Indifference got eaten off the evolutionary tree.

The machine is glorious in its indifference.

Best of all, the machine is ultimately logical.

The CPU is nothing but a collection of logic gates. The video card is so stuffed with logic gates that we don't even call it a video card anymore, it's a Graphics Processing Unit and it's most valuable as a ScamCoin Environmental Destruction Node. All those chips and circuits and ports on the motherboard are nothing but carefully intertwined wires charged with carefully regulated electricity.

Okay, the electricity has some quantum in it. Electrons can't help the quantum—our universe defines them that way. Don't blame anyone for how they're made, blame them for their choices and actions. (Like writing letters to advice columnists. That was certainly a choice.)

Quantum aside, at the macro-but-still-microscopic level? The whole machine is ultimately knowable.

If only there wasn't so much of it.

Think about what happens when you try to watch a conference video. You move the mouse over the play button. That mechanical mouse motion is transformed into electrical signals, which get dumped into some sort of operating-system-level interpreter, deciphered, and transformed into pointer motion on the screen. This is all operating system level work, originally developed by people from a previous generation. We consider these functions well-tested, even if the overwhelming majority of computer "experts" have no idea how it really works. Our predecessors wrote this code, and it basically works, so other than a few hard-core operating system developers we trust that the pointer will move.

The click goes through the same process, for the same reason.

The video is where things get really hinky.

Writing individualized instructions for each of the billions of transistors inside the hardware is so much work that, like apologizing, we refuse to do it. Propose writing a video player in pure assembler and any programmer capable of the task would either deride your parentage or charge enough up front to live comfortably in a non-extradition country. Even if you blackmailed a competent programmer into accomplishing such a task, they wouldn't really be addressing individual transistors. Primordial assembly, the sort that Kernighan and Ritchie wrote C to escape from, doesn't represent modern hardware. Assembly is closer to the logic gates than any other language, and it runs on top of processor microcode.

So, you add an abstraction, like C.

C lets us craft miraculous programs, like device drivers and text editors and segmentation faults. Some programmers can deftly hand-twiddle a "stack" that's a representation of computer memory in the 1970s. Forget mastering C; achieving journeyman C programmer status requires a certain species of electrified skull-fat, ample time, and either dedication or stubbornness.

Of those qualities, I possess only stubbornness. I do have laziness, which leads directly to Perl. Perl is written in C.

Let's say your video player is written in Perl. (You laugh, but I learned decades ago to never underestimate Perl programmers. A Perl programmer can achieve anything in the name of avoiding work.) Your code is an abstraction, running on an abstraction, running on an abstraction, running on a representation of hardware that was obsolete before Richard Nixon resigned. Every one of these abstractions has bugs.

By any reasonable logic, computers should not work. At all.

And yet, we've managed to make them work.

Realistically, your video player isn't written in Perl. It's in a web browser. The web browser is written using some sort of programming language or application toolset like Javascript or Go or Fortran or Haskell. Whatever. I don't know the real details and neither, unless you are extremely unfortunate, do you.

That's only the main engine of your web browser. It probably has add-on components written in Forth or Pascal or, Beastie help you, C++.

So, we don't have abstractions on abstractions. We have multiple piles of interlinked abstrac-

tions, all simultaneously affecting and rewriting one another as they co-operatively re-architect the contents of the machine's processor and memory. Yes, we've added "protections" to a bunch of these, but they're afterthoughts. Afterthought Security is not a thing.

Oh, I remember what I was thinking about!

Humanity's greatest invention? No, not the wheel. Or fire. Or even gelato.

It's bureaucracy.

A society is a machine made out of meat. We all have places in it. We're all continuously re-architecting its contents. Each of us can see only a tiny part of the machine. No one person can see the entirety of the machine; we can only truly see our little bit of it. We have opinions on the part of the meat machine that's most frustrating at the moment, because we're sure we wholly understand the issue even though others have spent years or decades maintaining it.

We sysadmins, we think we understand the machine when in reality we understand only a tiny slice of one of the many abstractions. A person who writes scripting languages thinks they have a good handle on memory management when what they really understand is the abstraction that the layer beneath provides to them. Repeat this for every single abstraction.

A modern computer is a giant bureaucracy. You understand, at most, an office. You could devote your life to comprehending the logic of one of these systems—but understanding the whole is nearly impossible. Evolving languages, evolving standards, evolving hardware mean that even if you achieve Buddha-level enlightenment, the machine will leave you behind.

Declaring a language "safe"? Read that as "We've done our best to isolate our mistakes from the other departments." I appreciate the effort even as I know failure is both inevitable and inexorable.

We march on a bridge of shifting sand across a bottomless chasm with no far end.

"Tell you everything will be fine?" No. Nothing will be fine. FreeBSD 13 is no different from any other operating system in that respect. It's merely the most honestly numbered release in history.

And it will never, ever apologize for it.

Have a question for Michael?
Send it to letters@freebsdjournal.org

MICHAEL W LUCAS The latest books to emerge from society's Michael W Lucas Abstraction Layer are TLS Mastery, Only Footnotes, and the imminent $git sync murder. Thirty years in systems administration have purged him of bitterness, cynicism, and sarcasm. Learn more at https://mwl.io.