WeGet letters

by Michael W Lucas

Dear Geezer,

You've been around a while. I know, because one of your books had a terrible accident when I was learning not to eat the cat and my dad was upset because you had signed it, meaning you had touched it, and he was afraid I might catch something from the pages. I know you won't be upset, because he had to buy a second copy, so you came out all right.

For almost that long, people have been babbling about this thing called "packaged base." It's supposed to solve everything, but it never happens. What's *really* going on with it?

—A Young, Smart Person Tired of Waiting

Dearest Useless Punk,

Just because my current age matches the freeway speed limit in oppressive regimes, like Salt Lake City, doesn't mean I'm a geezer. My experience and astute realism, which are often misperceived as undying impenetrable cynicism, make me the geezer. And packaged base has only fed that.

There is only one true way to upgrade any BSD system. You get the source code, preferably by checking it out via SCCS although floppies will do in a pinch and install it under /usr/src. You build the software. You can do this because real operating systems ship with fully functional compilers in the default install. Your phone is not a computer. It is an appliance. So is your doorbell, your pacemaker, anything running Windows, and most Linux installs. (Yes, with enough hacking you can get a compiler on your pacemaker, upgrading it to a computer, but that demands special skills and an impressive degree of reckless self-disregard.) You're running BSD, so you have a computer. Having built it, you install it on that same system, reboot, and voilà! You're upgraded. Yes, some optimizations are permissible — you can use NFS to share the source code and your hand-compiled binaries across your server farm, or better yet assign some flunky who has annoyed you to perform all the upgrades without disturbing you. This is the only way to be certain that the code you install is intended for your systems.

This is the natural state of any BSD system. Deviations from it are unnatural.

Unfortunately, a certain well-meaning but flawed person who I'm not going to specifically point out but whose name rhymes with *Polin Cercival* thought that FreeBSD needed an upgrade system usable by people with a morbid fear of compilers. (Never mind that such people should not be allowed near a computer, an appliance, or an abacus with three or more beads.) That's where freebsd-update(8) came from. It "conveniently" downloads the smallest possible binary diffs and applies them to a system, so that you can trivially upgrade thousands of systems without even working late. Working late, alone in the office, illuminated by only flickering emergency lights and with no sound but the hum of the air conditioner, is one of the fringe benefits of being a systems administrator. It gives you an excuse to be cantankerous the rest of the time. Why be a sysadmin if you can't surl at the lesser mortals? Fortunately, sysadmins still have developers, network administrators, and the entire sales department to provide an excuse.

And that's where we are today. FreeBSD can be upgraded by anyone who can weasel, wander, or whimsy their way into a root prompt.

Packaged base is the dread dragon of FreeBSD, devouring every developer who sets out to conquer it.

This deplorable state of affairs is somehow insufficiently welcoming for certain members of the community, however. They look at less magnificent operating systems and see that their so-called "base systems" are broken up into packages. User management software is a package. Network software? A package. Every little bitty piece of the system becomes its own package, with its own files and metadata and installation scripts and — worst of all — *dependencies*. Rule of System Administration #32 is very correct in that "Dependencies are the root of all suffering." We've all been trapped managing some barbarian system composed of dozens or hundreds of packages and discovering that essential programs like traceroute and ifconfig are not installed. You have to hunt around to figure out in which package this particular operating system imprisons those vital programs and try to install it, only to discover that the package management system itself needs updating and the package repository version has changed and a currently installed package isn't compatible with the new package and law enforcement officers show up to discuss what your boss keeps insisting was a "bit of an overreaction" when you know perfectly well the entire spree was justified and that the janitor will have no trouble getting the stains out of the carpet, ceiling, and driveway.

Who could possibly want to inflict this upon millions of FreeBSD users? Advocates say that packaging the base system would make it very easy to install minimal FreeBSD systems that contained only the programs needed to perform their assigned tasks. That sounds great, but it's like "exercise" and "eating healthy" and "not petting the adorable Sumatran tiger even though it's *right there*." It's not going to happen. Designing operating system installs that contain only what you need requires predicting infinite capacity to predict the future, or planning, neither of which is likely. You know perfectly well that the tiny system intended for use only for a nameserver is gonna wind up running CRM suites and video editing software for the CEO's nephew's girlfriend's glitterpunk band. That's the natural server lifecycle.

The correct way to get an uncomfortably sparse FreeBSD system is to build it from source. The FreeBSD build system includes options to include and exclude components. Michael Dexter has organized and tested all these options in his Build Options Survey (https://callfortesting.org/results/). You could even proceed directly to OccamBSD (https://github.com/michaeldexter/occambsd), a minimum viable FreeBSD build intended to host jails, bhyve, and Xen clients. OccamBSD is a good place to start, as re-enabling features is much simpler than tearing them out.

Fortunately, FreeBSD itself strongly resists being packaged. It is designed as a single cohesive system and does not like being teased apart into independent components. Sorting out what parts of the system truly depend on one another, and which are merely close personal friends, is a seriously hard problem that many developers have beaten their heads against for years. Many approaches have been attempted and failed. Packaged base is the dread dragon of FreeBSD, devouring every developer who sets out to conquer it. The world has an endless supply of optimistic developers, however, and I have no doubt that one day one of them will succeed and further weaken the moral fiber necessary to run FreeBSD.

With any luck, I'll be dead by then. Or at least not answering your letters.

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

MICHAEL W LUCAS' head is stuffed full of obsolete knowledge, much of it about FreeBSD, other BSDs, and a few other, lesser operating systems. To learn anything new, he'll have to throw out some of that junk. His latest books include *$ git sync murder*, *TLS Mastery*, and *SNMP Mastery*. *DNSSEC Mastery* should be out by the time you read this, but he's lazy so it probably won't be. Learn more at https://mwl.io.