acmqueue

Storage Strife

Beware keeping data in binary format

Dear KV,

Where I work we are very serious about storing all of our data, not just our source code, in our source-code control system. When we started the company we made the decision to store as much as possible in one place. The problem is that over time we have moved from a pure programming environment to one where there are other people—the kind of people who send e-mails using Outlook and who keep their data in binary and proprietary formats.

At first some of us dealt with the horrifically colorful e-mails by making our mail server convert all e-mail to plain text before forwarding it, but that's not much help when people tell you they absolutely must use Excel, and then store all of their data in it. The biggest problem is that these files take up a huge amount of space in our source-code control system, but we still don't want to store important information outside of it. Many of us are about ready to give up and just stop worrying about these types of files, and allow the company's data to be balkanized, but this doesn't seem like the right answer to me.

Binning Binary Files

Dear Binning,

While the size argument used to be a compelling one—perhaps even as recently as five years ago—we all know that terabyte disks are now cheap, and I would be quite surprised if you told me that your company didn't have a reasonably large, centralized filestore for your source-code control system. I think the best arguments against storing important company data in a proprietary or a binary format—and yes, there are open binary formats—are about control and versioning.

The versioning argument goes something like this. Let's say, for example, that the people who control your data center store their rack diagrams, which show where all your servers and network gear are located, as well as all the connections between that equipment, in a binary format. Even if the program they use to set up the files has some sort of "track changes" feature, you will have no way of comparing two versions of your rack layouts. Any company that maintains a data center is changing the rack layout, either when adding or moving equipment or when changing or adding network connections. If a problem occurs days or weeks after a change, how are you going to compare the current version of the layout to a version from days or weeks in the past, which may be several versions back? The answer, generally, is that you cannot. Of course, these kinds of situations never come up, right? Right.

The second and I think stronger argument has to do with control of your data. If a piece of data is critical to the running of your business, do you really want to store it in a way that some other company can, via an upgrade or a bug, prevent you from using that data? At this point, if you're trusting, you could just store most of your data in the cloud, in something like Google Apps. KV would never do this because KV has severe trust issues. I actually think more people ought to think clearly about where they store their data and what the worst-case scenario is in relation to their data.

I'm not so paranoid that I don't store anything in binary or proprietary formats, and I'm not so insanely religious, as some open source zealots are, as to insist that all data formats must be free, but I do think before I put my data somewhere. The questions to ask are:

- What is the impact of not being able to get this data for five minutes, five hours, or five days?
- What is the risk if this data is stolen or viewed by others outside the company?
- If the company that makes this product goes out of business, then how likely is it that someone else will make a product that can read this data?

The history of computing is littered with companies that had to pay exorbitant amounts of money to dead companies to continue to maintain systems so that they would not lose access to their data. You do not wish to be one of those casualties.

KV

Dear KV,

One of the earliest hires in the company I work for seems to run many of our more important programs from his home directory. These scripts, which monitor the health of our systems, are not checked in to our source-code control system. The only reason they are even mildly safe is that all of our home directories are backed up nightly. His home-directory habit drives me up a wall, and I'm sure it would aggravate you if you were working here, but I can't really scream at employee number six to clean his home directory of all important programs.

Employee 1066

Dear Employee,

I agree that you can get away with yelling at employee number six only if you are, for example, employee number two. Of course, that's rarely stopped me from yelling at people, but then I yell at everyone, so people around me are used to it. There really is no reason for allowing anyone, including a high-ranking engineer, to run code from a home directory. Home directories are for a person's personal files, checkout from source-code control, temporary files, generated data that the person doesn't need to share, and, of course, pirated music and videos. All right, perhaps that last one shouldn't be there, but it's better than putting it on the central file server!

There are two problems with people running things from their home directories. The first is the issue of what happens when they quit or are fired. At that point you have to lock them out of the account, but the account has to remain active to run these programs to maintain your systems. Now you have an emergency on your hands, as you immediately have to convert all these programs—without the authors' help—to be generic enough to run in your system. Such programs often depend on accreted bits of the author's environment, including supporting scripts, libraries, and environment variables that are set only when the original author logs into the account.

The second problem is that the user who runs these programs usually has to have a high level of privilege to run them. Even if the person is not actively evil, the consequences of that person making a mistake while logged in as himself/herself are much greater if the person has high privileges. In the worst cases of this, I've seen people who have accounts that, while they aren't named root, have rootly powers when they're logged in, meaning that any mistake, such as a stray `rm *` in the wrong directory, would be catastrophic. "Why are they running as root?" I hear you cry. For the same reason that everyone runs as root, because anything you do as root always succeeds, whether or not it was the right thing to do.

I know this is out of character, but if you're not the yelling type, I suggest nagging, cajoling, and even offering to convert the code yourself in order to get it out of this person's home directory.

KV

KODE VICIOUS, known to mere mortals as George V. Neville-Neil, works on networking and operating system code for fun and profit. He also teaches courses on various subjects related to programming. His areas of interest are code spelunking, operating systems, and rewriting your bad code (OK, maybe not that last one). He earned his bachelor's degree in computer science at Northeastern University in Boston, Massachusetts, and is a member of ACM, the Usenix Association, and IEEE. He is an avid bicyclist and traveler who currently lives in New York City.

© 2011 ACM 1542-7730/11/0400 \$10.00