ALVA COUCH

"standard deviations" of the average system administrator

Alva Couch is an Associate Professor of Computer Science at Tufts University, where he and his students study the theory and practice of network and system administration. He served as Program Chair of LISA '02 and was a recipient of the 2003 SAGE Professional Service Award for contributions to the theory of system administration. He currently serves as Secretary of the USENIX Board of Directors.

couch@cs.tufts.edu

A FORMER SYSTEM ADMINISTRATOR, turned analyst, muses about standards in system administration and other professions and the profound social effects of establishing or ignoring standards.

System administrators have a surprising amount in common with electricians. Both professions require intensive training. Both professions are plagued by amateurs who believe (erroneously) that they can do as good a job as a professional. Both professions are based upon a shared body of knowledge.

But electricians can call upon several resources that system administrators lack. Electricians have a legally mandated mentorship/apprenticeship program for training novices. They have a well-defined and generally accepted progression of job grades, from apprentice to journeyman to master. They advance in grade partly through legally mandated apprenticeship and partly through legally mandated certifications. These certifications test for knowledge of a set of standards for practice—again, mandated by law. The regulations are almost universally accepted as essential to assuring quality workmanship, function, and safety.

In short, one electrician can leave a job and another can take over with minimal trouble and without any communications between the two, and one can be sure that the work will be completed in the same way and to the same standard. Can any two system administrators, working for different employers, be interchanged in such a fashion?

At present, system administrators are at a critical juncture. We have functioned largely as individuals and individualists, and we greatly value our independence. But the choices we make as individuals affect the profession as a whole. I think it is time for each of us to act for the good of the profession, and perhaps to sacrifice some of that independence for what promises to be a greater good. This will be a difficult sacrifice for some, and the benefits may be intangible and long-term rather than immediate. But I think it is time now for us to change the rules.

From standards for distributions (e.g., the Linux Standard Base) to standards for procedures (e.g., those upon which Microsoft Certified Engineers are tested), I believe that—although standards may annoy us as individuals— standards for our profession (and certification to those standards) help build respect for system administration as a profession. Compliance with standards gives us a new and objective way to measure the quality of man-

agement at a site. Standards not only make the task easier but also enforce desirable qualities of the work environment and help to justify appropriate practices to management. Adoption of standards also has a profound effect upon our ability to certify system administrators and even changes the meaning and form of such a certification.

Learning from Electricians

Is a system administrator accorded the same respect as an electrician? I think the answer is an emphatic "no," at least for those electricians who hold a master's license. There are two factors that engender respect for a master electrician: legally mandated standards linked closely to legally mandated apprenticeship and certification.

One difference between being a senior system administrator and being a master electrician is the existence of the National Electrical Code and related local codes. The Code is a set of standards for wiring that broadly defines how licensed electricians must do wiring, as well as how vendors of electrical appliances and devices must construct those devices. The code has a specific and powerful property that *compliance to the code may be checked and certified by an independent examiner.*

Another difference is that *electricians go through a legally mandated apprenticeship and certification* before being allowed to practice as a master. There are three levels of electrician: apprentice, journeyman, and master. Becoming a journeyman requires both a mandatory apprenticeship (to either a journeyman or master) and certification of knowledge of both the National Electrical Code and any local codes that may contradict or strengthen the national code. Becoming a master requires serving as a journeyman under another master and passing another, more stringent certification exam.

System administrators could benefit greatly from such a mentoring system. By contrast, a typical beginning system administrator receives little training, guidance, or supervision, and there is no clear and universal path into the profession. We acknowledge the need for mentoring but have difficulty finding mentors or even clearly defining what mentorship should comprise. The basis by which electricians obtain all of these desirable things—and the central component of their training and certification—is a set of externally verifiable standards, embodied as the Code.

The standards for electrical work are an inextricable part of the certification process for electricians. But there are many system administrators who find standards annoying. Many system administrators have complained to me that the answers they must give about "best practices" on certification exams are "wrong." We pride ourselves on individuality, and on coaxing the last ounce of performance out of any system, and in reacting faster than anyone else in repairing a problem. To improve our personal "reaction time," we construct systems "in our own images" and, upon coming to a new site, face an irresistible urge to force it to comply with our personal standards even before we force it to function properly. But the term "personal standards" is really an oxymoron, because anything that only one person considers to be a standard cannot—by definition—be one.

A Tale of Four Repositories

No one really cares where software is actually located in a filesystem, provided that it is in the user's path. But apparently system administrators do

care because at my site there are currently four competing systems for locating software in filesystems. These evolved over time as follows:

- I created a system based upon the pattern /loc/category/package-revision back in the early 1990s, documented in a 1996 LISA paper [1]. User software installed into /loc is mapped into user-space at /local. This scheme was used for many years, while I managed the repository between 1990 and 2000.
- When I hired my own replacement, he decided this scheme was "inelegant," and started installing packages in /loc/packages/package/revision. I did not stop him because I did not want that job back!
- When he hired a student assistant, the assistant did not like that scheme and started installing packages in /rel instead, using a format not easily explainable and whose description—mercifully—has not survived the ravages of time!
- When these folks left for greener pastures, the next sysadmin decided "all packages should be locally installed" and installed them in /var /local/.

And so it goes. What was the result of this? As I write, three of these four repository schemes are still active. Part of the reason for this is that the packages make reference to themselves at their installed locations. System administrators were quick to adopt new "personal standards" (oxymoron intended), but the work of making the system "comply" with those standards was never completed, and there are programs in my original repository that—even though it has not been updated since 2000—are still in use and were installed as early as 1994!

That Standard Is So Lame!

Each system administrator in this story thought he was doing a good thing by imposing "personal standards." The first one exclaimed to me that my "old standard is so lame!" The others made equivalent comments about the schemes of all of his predecessors. Each one believed sincerely that his new "personal standard" would be the one that would make the most sense and pass the test of time.

To me, looking at it from the outside, this attitude was something like a gambling addiction, as each new player entering the game thought his new scheme would "win" where other schemes had "lost." But—as surely as most gamblers lose at the casino—every "personal standard" ended up being "lame" in the end!

Each "personalization" seemed at the time to be a "solution" but turned over time into a "problem." The real thing that is "so lame" in this story is that there are three "personal standards" for doing the same thing. There is only one mechanism by which our site could escape the cycle: by adopting standards for shared package location and then assuring compliance to those standards. "Personal standards" just do not work.

Incidental Complexity

What standards could really help the profession of system administrator? I and my students have made a multi-year study of what we call the "incidental complexity" of system administration [2]. Incidental complexity arises from making arbitrary decisions without coordination and for no particularly good reason. The main cause of incidental complexity is that many management decisions have nothing to do with final system behavior. A pa-

rameter value is "incidental" if the choice of a value is a matter of style and preference and has no conceivable effect upon system function. My students and I have found that, on average, the values for most configuration parameters are incidental.

Incidental complexity includes choices for locations of files within a system. For example, the outward appearance of a Web server has nothing to do with where files for the Web server are stored, which led one of my students to propose that this choice be taken away from the administrator by automated mechanisms [3].

Based upon experiences like these, I believe that appropriate practice standards for system administration involve identifying incidental choices and standardizing those choices. This might seem hard for an unvirtualized system, but in a properly virtualized execution environment, there are even more incidental parameters than before. I believe the following choices are now completely arbitrary for a site of sufficient size:

- Names of home directories (use SAN volumes to enforce quotas and enforce barriers between user groups).
- Locations of published Web content (use SAN volumes and standardize locations of mount points).
- IP addresses of hosts (use IPV4/NAT and/or IPV6).
- Locations of remotely installed and locally installed software packages.

Through a combination of virtualization and standards, the values of these "parameters" do not matter, and every site can choose these in exactly the same way.

The benefits of this kind of "standard" are subtle but profound. If all repositories are named the same way, a system administrator won't have to read site documentation to fix a Web server, any more than an electrician has to stop—while wiring your house—to refer to the National Electrical Code. In the same way that no electrician will work on noncompliant wiring, a system administrator won't even try to learn what has been done in a nonstandard fashion, because such work is not sufficiently externally verifiable. No more "job security," but, rather, an increase in overall system administrator efficiency and a dramatic reduction in what every practitioner has to remember.

The Linux Standard Base

One example of eliminating incidental complexity is the Linux Standard Base (LSB) [4]. This is a set of standards for layout of Linux distributions, and it specifies the locations and versions of important files and libraries in a Linux distribution. The purpose of the LSB is to assure that vendor-supplied Linux applications will run in an LSB-certified environment. There are two levels of certification, both checked by software tools. First, the environment is certified as being compliant with the LSB standard, by running a script that checks that all files and versions are present and in the proper places. Then the application itself, as a binary file, is checked for accessing library functions properly, with the correct types of arguments.

The "theory" behind the LSB is "test once," "certify once," "works everywhere." If one tests an application in a certified environment, and the application works there, and the application is itself certified, then we have high (but not quite perfect) confidence that it will work in any certified environment (where uncertainty is due to esoteric technical limits of LSB beyond the scope of this article).

The LSB serves as one example of a practical standard. One can run the certification tools on an environment or application and get a "yes" or "no" answer, so the LSB meets the definition of external verifiability. A rather long document describes compliance measures in plain English, so the LSB meets all of the criteria for a standard.

Cost and Value of Standards

The LSB serves as a good case in point for considering the cost and value of standards. The cost of compliance to the LSB is actually severalfold and a bit subtle:

The LSB always lags behind current operating systems versions and distributions, because a distribution has to exist before it can be standardized, and the standardization process takes time. Thus, almost by definition, a compliant environment is "out of date" by some reckonings.

- The LSB is "a lot to remember" when managing a system. In particular, upgrading a system often inadvertently invalidates the base. There are inadequate tools for asking "what-if" questions about updates and making intelligent decisions. Compliance takes time to initially assure and time to maintain.
- The LSB does not standardize all aspects of application and system, and unstandardized aspects can cause an LSB-compliant application to fail in an LSB-compliant environment.
- But the value of compliance is a bit more subtle to itemize:
- Compliance only provides assurance that one configuration will work, but not that other configurations will not work. Thus it is possible to "do without" LSB compliance and not feel the pain.
- Compliance is thus not a guarantee of "point behavior," but of "lifecycle behavior"; it isn't "required" at any one time, but overall compliance—over time—increases software reliability, at the cost of being slightly "behind the curve" of software development.

In the same way, the National Electrical Code is important to electricians—not because houses could not be wired differently and still function—but because the standards therein ensure that, over the house's lifecycle, the wiring is unlikely to fail compliance with the Code and that any electrician who knows the Code can come to a compliant house and know what to expect and how to change it.

Certification to Standards

Adopting standards has a profound effect upon the meaning of certification. So far, certification efforts for system administrators have concentrated upon certifying skills; I have commented on the dubious value of this kind of testing and certification in a previous *;login:* article [5]. The license exam for a master electrician instead certifies knowledge of the Code; skills are tested instead in the context of apprenticeship, during which there are numerous opportunities to observe them. Likewise, an exam about system administration practice, rather than about knowledge, is much narrower and easier to create than an exam about the general knowledge required to function as a system administrator; the latter is also ideally tested during apprenticeship. In the context of practice standards, certification means that the system administrator has enough knowledge of the relevant standards to graduate from the apprenticeship and produce a compliant site.

Putting Lifecycle First

I propose, therefore, a profound change in strategy for the profession of system administration. I propose that we learn from other professionals and borrow some of their better ideas. I propose that we drop "personal standards" in favor of "professional standards" and strive for universal respect for those standards. I propose that we stop looking out for ourselves and start looking out for a profession that can take care of us as a collective group. Part of engendering that professionalism is a set of shared values that must trump the personal values of the past.

This is just part of the lifecycle of the profession. We learned in the early days that there were certain practices we use that distinguish us from amateurs. We progressed to define "best practices" as our first "standards" but realized that many of these are "personal." We progressed to understand the value of systems standards such as LSB. Now we are at a juncture where it is possible to move past personal professionalism to a definition of professionalism that is practice-wide. This requires giving up autonomy, very much as an electrician cannot function outside the bounds of the Code. What we gain, however, is something very precious, which is that the profession itself attains an enduring value that is—like a good standard—externally verifiable.

That is what "best practices" really means.

REFERENCES

[1] Alva L. Couch, "SLINK: Simple, Effective Filesystem Maintenance Abstractions for Community-Based Administration," *Proc. LISA X*, pp. 205–212, USENIX Association, 1996.

[2] Alva Couch, John Hart, Elizabeth G. Idhaw, and Dominic Kallas, "Seeking Closure in an Open World: A Behavioral Agent Approach to Configuration Management," *Proc. LISA XVII*, pp. 125–148, USENIX Association, 2003.

[3] Steven Schwartzberg and Alva Couch, "Experience Implementing an HTTP Service Closure," *Proc. LISA XVIII*, pp. 213–230, USENIX Association, 2004.

[4] The Linux Standard Base: http://www.linuxbase.org.

[5] Alva L. Couch, "Should the Root Prompt Require a Road Test?" *;login:*, Volume 32, Number 4 (August 2007).