Teaching System Administration

Steve VanDevender
University of Oregon

Abstract

For the past twelve years I have taught a one-term college-level class introducing students to the discipline of system administration. I discuss how the class was created, the considerations that went into designing the class structure and assignments, student outcomes, how the class has evolved over time, and other observations on teaching. Links to detailed course materials and other resources are provided.

1. Introduction

In 2000 I began teaching an 8-week class titled ''Introduction to System Administration'' once each year during the summer session for the University of Oregon Computer and Information Science department. This class has been popular with students and also attended by University staff and community members who were already working as system administrators as a training opportunity.

Designing and running this class was a challenge since I initially lacked formal training or experience in teaching a college-level class. Careful thought and some experimentation went into developing a set of assignments that emphasized the combination of technical and non-technical topics suitable for undergraduates who might have no previous system administration experience.

By describing my experiences with creating and teaching this class, I also hope to inspire others with an interest in education to get involved, while also giving them some practical suggestions and a realistic idea of the challenges.

2. How did I get into this?

I attended the first System Administration Education workshop at LISA '99. At the time it looked like an interesting alternative to taking a tutorial. It turned out to be even more interesting when the attendees, a mixture of experienced educators, people interested in developing training for their workplaces, and inexperienced people like me, had widely varying views on education techniques and what should be in a system administration curriculum. Some of this came from a lack of a widely agreed-upon definition of system administration itself. Later workshops that I have attended continue to struggle with these issues, partly because the technology of system administration and its role in organizations and society continues to change rapidly.

At the workshop Mark Burgess was also passing around a draft of a system administration textbook [Burgess] he was writing. I got a chance to skim through it and was pleased that unlike many other general books on system administration, it really was structured more like a textbook than a collection of how-to topics.

Although I didn't go in to the workshop with the ambition to become an educator, I came out at least hoping to encourage my local computer science department to develop a class in system administration. When I first suggested this to the head of the department, her response was ''But that would be so . . . practical.''[1] This was not that surprising given the theoretical bent of our computer science department.

Later a colleague and I were invited by the professor teaching their operating systems class to speak to her students, and at the end of the class I asked her if the department would be interested in developing a system administration class. She said ''That would be great! Will you teach it?'' My stunned reaction was a qualified ''yes'', and that I would have to clear it with my manager. As my position description actually included teaching as an option, my manager gladly approved and I was shortly contacted by the summer session coordinator to start setting up the details of the class for the 2000 summer session.

3. Developing the class

My computer science department gave me a great deal of independence in developing the class and its content. The professor who championed the creation of

[1] She later claimed she was kidding. I don't remember that being obvious at the time.

the class offered her advice to avoid being too ambitious (which I agreed was wise), but otherwise there were few requirements or even recommendations for content nor was I required to submit material for review or approval.

The department was able to provide me a small lab of computers that could be dedicated to my students while my class was in progress, which was a major influence on my assignment design.

I had a number of obvious constraints that also affected my design:

- The class lasted only eight weeks. Although summer session classes had four or five lecture days a week, comparable to an 11-week 4-credit-hour class with three lecture days a week during the regular school year, the amount of material I could cover was limited and I would have to focus on basic, essential topics. I titled the class ''Introduction to System Administration'' to try to establish the right expectations.
- That first year there were 10 lab computers available for up to 30 registered students, so students would have to share computers. Since I wanted to emphasize themes of communication and teamwork, I could actually use this to my advantage.
- The lab computers didn't necessarily come with usable operating system installations (they were used by other classes than mine) and would have to be have installations created.
- Lab access was limited, so I wanted to make the computers remotely accessible as soon as possible. On the other hand I didn't want to put a lot of vulnerable machines on the network, so some basic security issues had to be covered early, even though security is often considered an advanced topic.
- I did not want the class to be a system administration ''boot camp'', based partly on the advice not to be too ambitions. While I wanted class work to accurately reflect real system administration in some ways, I also wanted the class to be accessible to students without previous experience and who weren't necessarily interested in system administration as a career.
- I would be responsible for all assignment grading and other management of course work. Even if I could have obtained a teaching assistant, I wasn't sure having one would be useful given the amount of extra coordination required. Therefore I had a strong incentive to keep assignment evaluation as simple as possible.

I also had several notions of teaching philosophy based on my own experience as a student.

- I had hated assignments that were poorly specified and had unclear evaluation criteria.
- I knew that different students had different preferred modes of learning, like reading, interacting with teachers and other students, and hands-on work. Therefore I provided information in each of these modes: textbook readings, lecture material and classroom discussion, and hands-on assignments.
- I wanted to create a class that provided underlying principles and a framework for knowledge, rather than just training students how to perform specific technical tasks.
- I wanted to emphasize important non-technical aspects of system administration such as effective documentation, communication, and service to a user community.

4. Class description

Complete on-line materials for my class for the years 2004 onward are available in [VanD] including the syllabus, full assignment descriptions, and lecture material. Here I will focus on the considerations that went into designing the assignments and other course content, and how the assignments have developed over time based on my experiences.

The overall sequence of my system administration class involves students working together in small groups to install and maintain computer systems and perform a variety of basic system administration tasks. I also want to allow students an opportunity to pursue a topic of their own interest and learn how to manage their own system administration projects, so the last two weeks are dedicated to a final project that student groups get to design and implement themselves with guidance from me.

4.1. Week 1: Class setup

The university's registration policies allow students to drop classes without fee penalties or add classes without special permission from the instructor through the first week of the term, so I have found that enrollment isn't really stable until the second week. I use that first week to provide general background for the class, introduce students to the lab, and to have students form groups. Usually students are good about doing this on their own so I don't have to assign

students to groups.

Because group work is required in the class I explain my expectations for how groups should work. Groups should try to divide work evenly among members as much as possible. Someone who is expecting to be absent for an extended period should let their group and me know, and work out how to handle that absence with their group. I took an idea from the LISA '99 education workshop for evaluating group contribution, and ask group members to privately send me estimates of the relative contributions of all group members with each assignment. This tends to highlight cases where someone really wasn't contributing equitably since the other group members tend to agree when that happens. However, if there does appear to be a problem, I won't assign unequal credit to group members for assignments until I can meet with the entire group and discuss the situation with them, to make sure it really is a problem with someone not contributing, and not some other sort of internal friction.

4.2. Week 2: System Installation

Before anything else, the computers used by groups need an installed operating system. I decided to try a dangerous experiment the first time I ran the class and allowed students to choose their OS distribution themselves, as long as it was freely available for educational use. I hoped that this would give people in the class indirect experience with a variety of OS distributions to see that all could be made to work. This turned out to be successful and I've continued to give students that choice, although it requires me to maintain basic knowledge of the various Linux and BSD distributions and their peculiarities. This typically results in a mixture of different Linux distributions (Ubuntu, Debian, and Fedora being the most popular recently) and one or two FreeBSD or OpenBSD installations.

However, the real goal of this assignment is to introduce students to creating repeatable, documented procedures and some basic concepts of version control. One group member does an initial installation and writes an installation document for it, and every other member of the group is required to do at least one OS installation using and revising documentation created by the one before, using RCS to record each revision to the install document. I chose RCS as the version control software because it is widely available, demonstrates basic version control concepts clearly, and, unlike some more modern version control systems such as CVS or Subversion, requires no extra infrastructure to use and can be applied to individual files as needed. Later assignments also require use of RCS for change tracking.

The material each group hands in is the RCS file containing their installation document and its revision history, which I grade based on whether each group member contributed at least one revision, and on the comprehensibility and completeness of the documentation.

4.3. Week 3: Network installation and updates

The tasks in this assignment are determined by a practical considerations for use of the lab. Limited lab hours provide students with little time or flexibility for physical access to their lab computers, but if their computers are on the network they can work on later assignments remotely at any time and place with network access. However, it's necessary to minimize the vulnerability profile of the computers before they are put on the network, since usually security updates aren't conveniently available to them until after that has been done. That means disabling all non-essential network services, which for our purposes leaves only `sshd`.

In order to determine how to disable network services on their systems, students need to know how those services are started. The first part of this assignment involves investigating how all the running processes on their computer are started by exploring their `init` configuration and scripts (or corresponding `upstart` or `systemd` configuration in more recent OSes), which usually directly indicates how an unwanted network service daemon can be disabled. This also gives students a motivation to dig in to their systems and understand how system initialization works.

Before the class started I removed the network cables for the lab computers. Once students believe they have disabled all network services other than `sshd`, I personally audit their computer to ensure that it is safe to put on the network, and give them a network cable at that time. This also gives me an opportunity to help them create and test their network configuration. The amount of effort required to ensure secure installations has decreased over time, as distributions have converged on secure-by-default policies so the number of services that are running that need to be disabled has decreased, and `sshd` is almost always installed by default or quickly installed as a package. However, some distributions also have become more dependent on network-based installation, so sometimes only a very basic installation can be made from CD or other media, and cannot be fully completed until the

computer is on the network.

After this students have to investigate which security updates are available and relevant for their OS, install them, and document which ones were installed, to assure me that they actually were. They are also required to state a security update policy that they will follow for the rest of the class, such as checking for and installing updates once a week.

Although personally auditing group computers keeps me fairly busy during this week, it has paid off in that I have had very few security problems. Those few that have occurred have been issues like weak account passwords or insecure application installations for final projects. The early emphasis on security also helps encourage students to think about it as a fundamental part of everything else they do.

4.4. Week 4: Network services

The main tasks for this assignment are to install current versions of Sendmail and Apache compiled from source. This is possibly the most technically difficult assignment of the class. When I first developed this assignment, I had certain reasons for requiring source installs. Installing software from source distributions was more common in 2000. Because everyone was using different OSes, some might have current packaged versions of these programs that could be installed with minimal effort, while others would have to build them from source anyway, so making everyone build them from source meant everyone spent a similar amount of effort.

Things have changed a lot since then. More distributions now have reasonably current prepackaged versions of Sendmail and Apache with workable default configurations. Software management using packages is now much more stable and popular. However, one reason I continue to require building these packages from source is to make students confront the issues of software installation and configuration. Installing a package with a working default configuration is usually easy. Doing from-source installation gets students to understand what package managers actually do behind the scenes. Binaries and configuration files have to be placed properly and the configuration has to be examined and appropriate customizations applied.

Since students need a C compiler and the right set of development libraries in place, most still have to also install a number of packages. Apart from Sendmail and Apache themselves, I encourage students to install packages for everything else they need.

The lecture material for this assignment, besides providing background on how software is built from source code, also explains the SMTP and HTTP protocols, and issues relating to service management. A lecture on the problem of email spam has been popular.

Evaluation of this assignment is practical and results-oriented. Groups have to demonstrate that their computer can send and receive email and serve web pages from the document root and from their own user accounts. They also have to provide the primary Sendmail and Apache configuration files, RCS logs tracking any customizations, and their installation notes.

4.5. Week 5: Account management

For this assignment, students have to create accounts for everyone in the class, including me, on their systems. The methods they may use to accomplish this goal are left unspecified, although the two most common approaches are to either have each account holder pick a login name and initial password, or to use a list of all class members to assign precreated login names and passwords.

By not dictating any preferred method of organizing account creation, I usually see a number of approaches. Sometimes this also results in creative or naive approaches that can be problematic. Once, given a preprinted slip that said my password was `$password$`, I asked aloud in class whether that group had assigned everyone in the class the same password (they gave me a rather embarrassed look — it turned out they had). Another group asked everyone to PGP-mail them a requested username and password, and helpfully included a `---PGP PRIVATE KEY---` block in the mail they sent to everyone.

The first time I handed out this assignment, I had somewhat unwisely failed to establish how I was going to evaluate it beforehand. What I ultimately decided to do was evaluate each group based on the proportion of their users who had requested accounts who were able to successfully log in to their accounts on the group's system. One of the problems with this was getting everyone to both put in the effort to request accounts on all the other computers and report whether their accounts worked. I do some cross-checking to determine whether the reason someone didn't get an account was because they didn't request it, didn't work with a group to report and resolve problems with it, or because the group responsible for creating it didn't put in the effort to get it working. Noticing that there were consistent problems each year with a few people not providing reports about the status of the accounts they requested, I eventually made a policy that any

individual who did not send me a report of which accounts they had requested and which they had working would not get credit for the assignment.

While it might be argued my chosen evaluation method could be unfair, although I am careful to watch for signs that anyone might be gaming the results, it is the best way I have been able to think of to emphasize that interaction with a user community should be based on providing good service. Based on my early experiences with this assignment I have since been careful to emphasize that the difficult part of this assignment is communicating effectively with everyone else in the class and being responsive to reported problems. Also, I advise that a simpler approach like pregenerating accounts tends to work better than a complicated approach like trying to write a web-based signup system. Since each person in the class has to interact with all the other groups to get accounts, everyone gets a sense of the tradeoffs of the different approaches to handling the assignment.

The other component of this assignment is having groups write a basic use policy for their system, as a way of encouraging students to think about issues in policy creation and enforcement. I don't expect students to write legally airtight acceptable use policies, but I do want them to understand common kinds of inappropriate computer use and what they might have to do as sysadmins to monitor and enforce compliance with policies.

4.6. Week 6: Logging, access control, scripting

This assignment gives students some basic experience with analyzing log data, experimenting with access control for their computers, and writing a simple log processing script that is to be run from cron. This is combined with lecture material on more advanced security topics.

Evaluation of this assignment is also very practical and simple. Students must provide certain specified log events, such as the pairs of Sendmail log lines showing incoming and outgoing mail being delivered, and logs showing that access control was used to permit or deny access to a service. They also have to write a simple script that mails them selected events from their system logs once a day when run via cron.

This assignment is the least complicated one. It's something of a break before the final project.

4.7. Weeks 7-8: Final project

In order to give students an opportunity to explore a system administration topic of their choice and learn some important principles of project planning, they get to design their final project themselves. I require that their design address five of the things I consider most important when doing any system administration work and planning projects:

(1) Concrete goals

(2) Specific benefits to their user community

(3) Security considerations

(4) Estimated effort of implementation and maintenance

(5) Documentation for implementation and use

A project proposal that specifies their concrete goals and outlines questions that cover the other topics is due early in week 7 to encourage some advance planning. I strongly recommend they discuss their ideas with me so I can make sure their project is realistic, since they sometimes propose extremely ambitious projects thinking that those will be better received. However, I tell everyone that projects will be evaluated based on how well they think about and address those topics, and not on how ambitious or complicated it is. The completed project, including a write-up discussing the five topics above based on the group's experience in implementing their project, is due at the end of week 8.

Examples of some notable final projects include:

- Replacing Sendmail with qmail, while preserving compatible user functionality.
- Building a honeypot system and successfully investigating the behavior of someone who broke into the honeypot environment.
- Building a NAT gateway system that provided dynamic IP allocation and firewalling for clients behind the NAT boundary.

4.8. Other elements of the class

The weekly assignments are each worth 10 points, while the final project is worth 30 in total. I originally did not grade the final project proposal separately from the final hand-in, but found it was hard to get students to cough up the proposal. Once I remembered my own student days, I realized that from a student's point of view, "if it's not worth points, it's not worth doing", made the proposal worth 10 points, and got a lot more handed in complete and on time.

Although I did not do this officially in the first few years of the class, later I held in-class discussions

for four of the assignments (week 2, 3, and 5, and the final project) each worth 5 points. This gives students an opportunity to share information with each other, and also conveniently rounds out the point total for the class to 100, making grading computation slightly easier.

In week 7 I hold an activity called "System Emergency Day!" (another idea I got from an early LISA education workshop) where I cause simulated outages to student systems by various means, such as doing `chmod 700 /`, renaming an important but non-critical system shared library, disabling some important system service at boot time, or modifying network configuration settings. All are chosen to impair system functionality without causing data loss and without requiring complicated effort to fix (at least if the correct underlying cause is found). Students then have to diagnose the problem, often with some hints from me to help keep them on track and keep the exercise from running too long. I have found that students enjoy this activity greatly, and some have even suggested doing more of this during the class, although the amount of effort required by me and the students is such that I usually spread it over two days so that I only have to track a few student groups through the exercise each day.

5. Observations and ideas for the future

Developing some 30 hours of lecture material (50 minutes a day, 4 days a week for 8 weeks) was my biggest challenge the first year, especially since initially I didn't have much intuition for how much material fit into a 50-minute class period. Having at least an idea of what needed to be covered as background for each assignment, I created an outline of everything I wanted to talk about, and got an idea of what fit as I went along. It's still an imprecise art, since student questions and in-class discussions affect how much time is available to cover each day's material, and I still often have to postpone or rearrange things as I go.

Later I expanded the outlines into readable prose that I posted as lecture notes on the class web site, to make them more usable to students as reference material or for catching up on missed lectures. I also use these as slides in class, although there is some question about whether it's useful to present that kind of prose to people while speaking. I have also tried to freshen and update the lecture material to reflect changes in current practices and my own thinking about certain system administration issues.

While I initially required Burgess's textbook [Burgess] (although I did not directly use its exercises), over time I found most students tended to be reluctant to buy it, especially as textbook prices have increased. I still recommend it and also provide information on the course web site [VanD] about a few other textbooks for reference and background, but do not require its purchase any more.

Enrollment for the class has tended to reflect overall enrollment in the computer science department. Demand for the class was highest during the first four years, when it often filled up during registration. The higher maximum class sizes in those years were an attempt to meet demand, although the year with 38 enrolled students was especially difficult.

year	enrolled	maximum	groups
2000	23	30	9
2001	23	30	9
2002	38	40	11
2003	28	30	11
2004	14	20	7
2005	14	20	7
2006	17	20	8
2007	12	20	5
2008	11	20	11
2009	14	20	6
2010	17	20	7
2011	10	20	5
2012	17	20	7

During the busiest years groups were often four people, and I got consistent feedback that many of those groups felt too big and that dividing assignment work among that many people left them without as much to do. As enrollment decreased I have recommended people form groups of two or at most three which seems to provide the best division of labor. In 2008 with only 11 students I somewhat relaxed the group work requirements, but found that it was hard to get the amount of collaboration I wanted to see when people were largely working on their own.

I haven't changed the overall structure of the class much since I created it. There have been changes to address problems students experienced with the assignments, but I've been reluctant to make major changes to a largely successful plan. However, I do feel some of the material is becoming a bit dated, and I have considered whether to try to bring in more modern topics like virtualization or configuration management. When thinking about how to integrate those, I've been reluctant to try to squeeze more into what is already a busy class, and also felt that some advanced topics

would be out of place in an introductory class, like trying to teach calculus to pre-algebra students. Adding a sequel class seems like the best way to teach those more advanced topics.

Having a dedicated lab for the class has greatly influenced its structure and the types of assignments I offer. Without a dedicated lab I would probably have to create something like a virtual lab, where students could load up their own virtual machine images so there would be no (or at least less) dedicated hardware. However, this would require restructuring many of the assignments that assume systems will be online all of the time (in particular the Sendmail/Apache and account creation assignments).

6. Conclusion

The development of system adminstration as a profession requires bringing new people into it, and having more and better opportunities for people to learn about the profession will recruit people beyond the self-identified and self-taught. Ideally there should be collegiate degree programs focused on system administration, but that still requires finding people to develop a curriculum and and create and teach the classes.

There's also some debate over how much and how well system administration can be taught. Being an effective system administrator involves independent thought and creativity, qualities that are typically thought of as hard to teach. Most system administrators are still self-taught, and most haven't reflected on how they learned important ideas and how those ideas could be taught to others.

Elizabeth Zwicky's talk from LISA 2006, ''Teaching Problem-Solving: You Can and You Should'' [Zwicky], makes a strong case that problem-solving can be taught, particularly in the context of system administration, and provides excellent advice for the would-be teacher. I'd say that a listener can generally replace ''problem-solving'' with ''system administration'' in her talk and her points remain valid. Perhaps her most important point is to believe that system administration skills can be taught, and that education experts don't find that idea controversial. Basic skills can be taught in well-understood ways, and more sophisticated skills can be developed by giving students a safe environment for experimentation and the motivation to learn through a progression of more difficult problems.

The benefits of teaching aren't just for students. It's sometimes said that you don't really understand something until you can teach it to someone else. Learning how to teach will help you understand your own profession better. Many of the skills that are essential to teaching are useful in any professional context, such as patience, good communication and interpersonal skills, and the ability to supervise and evaluate the work of others effectively.

Some of the circumstances that made it possible for me to get involved in teaching were just luck, such as having a job description that allowed for part-time teaching and finding an eager faculty sponsor to champion my idea for a class within our computer science department. Because my class is an elective, students take it because they have a genuine enthusiasm in the subject, and are much more willing to allow for the experimentation I found necessary to refine the class. However, none of this would have happened without my having the idea to try to teach in the first place. By showing you how it was possible to develop a small but successful class from scratch, I hope some of you will have the same idea and find your own opportunities.

References

[Burgess] Mark Burgess, *Principles of Network and System Administration*, 2nd ed., John Wiley & Sons Ltd, 2004.

[VanD] Steve VanDevender, ''CIS 399: Introduction to System Administration'' web site, `http://www.cs.uoregon.edu/Classes/index.php?course=cis399sysadmin`. Retrieved September 2012.

[Zwicky] Elizabeth Zwicky, ''Teaching Problem-Solving: You Can and You Should'', LISA 2006 Proceedings, `http://www.usenix.org/conference/lisa-06/teaching-problem-solving-you-can-and-you-should`. Retrieved September 2012.