JAN SCHAUMANN

teaching system administration in the cloud

Jan Schaumann is a Systems Architect at Yahoo!, a nice place to stay on the Internet. He is also a part-time instructor at Stevens Institute of Technology, where he teaches classes in system administration and in UNIX programming. Like most people, he has his own ideas as to what exactly "cloud computing" is supposed to mean.

jschauma@netmeister.org

LEARNING SYSTEM ADMINISTRATION IN an academic environment requires students to have full superuser access to multiple operating systems (OSes), but many schools' resources cannot provide this kind of access. I have successfully used Amazon's Elastic Compute Cloud (EC2) as an alternative to traditional computer labs or Live-CDs. Using EC2, free for students in most cases, allowed me to offer more flexible and valuable assignments and learning possibilities beyond the boundaries of traditional university resources. There are, however, some disadvantages as well.

System administration has long been a profession that is learned primarily by experience, where people grow into a position in order to fulfill the requirements of an organization rather than follow a career path well-defined by courses, degrees, and certifications. Up until fairly recently, formal classes in system administration as part of a Computer Science or Engineering curriculum were uncommon, but in the past few years more and more institutions have recognized the industry's need for academic courses that adequately prepare students for the multitude of responsibilities within this field.

To really have students understand and appreciate some of the most general aspects of system administration, they need to gain practical experience. They need to administer a system, to have superuser access, to have a chance to configure a system for a specific service, and to make the kind of spectacular mistakes that experienced system administrators value—if only in hindsight.

This normally conflicts with the requirements of the IT staff at the universities: students would require access to a number of different OSes when the school's system administrators strive for a certain level of homogeneity. In order to understand OS installation concepts, filesystem tuning, and other low-level principles, students need to perform these tasks themselves. Learning to debug network connectivity issues or being able to actually see the payload of captured network traffic requires access to raw sockets, which the staff responsible for the security of the campus network would certainly rather not provide.

System administrators are unusually creative and frequently able to provide solutions to help overcome any given limitation. Isolated labs can be made available to students, or various virtualization technologies may make it possible for students to set up and control OS instances outside the standard university environment. All of these incur a significant maintenance penalty and still, in the end, students normally remain confined to a sandbox of some kind, necessarily limited in their learning environment in one way or another.

When I started teaching system administration at Stevens Institute of Technology (not entirely coincidentally, at a time when I was working as a system administrator at Stevens Institute of Technology), I created the syllabus under the assumption that at the very best I could only convey parts of the many topics relating to system administration. "Aspects of System Administration" [1], therefore, covers very broad topics in each lecture, using practical assignments to let students understand and experience what was talked about.

Elastic Compute Cloud

After years of struggling to give all students equal access to the resources required for the diverse assignments and frequently having to change the assignments to be able to make use of available resources, I finally decided to do what many companies do. I outsourced the resource provision problem into "the Cloud"—specifically, to Amazon's Elastic Compute Cloud (EC2).

Having recently experimented with different cloud platforms, I knew that this approach should make it possible for me to make available to each student an unlimited number of servers, different OS instances, not in a restricted network using theoretical or contrived examples but on the Internet, where they could have full superuser access. Students wouldn't be restricted by lab hours or firewalls. And, as an instructor, I could create assignments that let students experience having full control over their systems.

As a result, at the beginning of the semester each student was required to create an account with Amazon's Web Services (AWS)—account creation itself is free of charge, although a valid credit card needs to be provided—and familiarize themselves with the concepts of cloud computing in general and Amazon's EC2 in particular. Furthermore, they needed to download and install the tools (provided by Amazon) to create, manipulate, and maintain EC2 instances from their preferred platform [2]. Thanks to Amazon's educational program [3], students were given up to $100 in compute resource credits—more than enough for all required course work.

Since EC2 OS instances are billed based on their actual usage, it should be noted that I did have to repeatedly instruct students to remember to shut down their instances after their work was done, so as to avoid using up their allocated free credits. In the end, one student managed to leave two instances running for over a week, using up over two-thirds of his free credits. However, the current price for a standard or "small" instance (32bit, 1.7GB RAM, 1 virtual core, 160GB local instance storage) of $0.085 per hour made it possible to expect students to cover any required additional compute time themselves. At the end of the semester, almost all students still had several hours of compute credits left, allowing them to further experiment with EC2 and deepen their experience for free.

Assignments

The first assignment was for students to create an OS instance following the instructions provided in Amazon's "Getting Started with EC2 Guide" [4] and run a few simple commands. Anticipating the learning curve all students would experience, students had a full week for this assignment; once comfortable with the tools, spinning up a new instance and running the required commands would normally take minutes.

Amazon offers two main ways of interacting with your OS instances: via the Web browser and by using a set of command-line tools. Both ways utilize the well-defined and open API underneath, the programmatic use of which could well be expanded into a separate assignment or even a stand-alone class. Not very surprisingly, initially many students chose to use the GUI-oriented Web interface for this task. Subsequent assignments required the use of the CLI as a way to illustrate how flexible tools can be combined into a powerful program able to control a number of remote instances. These tools make it possible for students to create or destroy—on demand—an OS instance, create and apply firewall rules and access restrictions, and monitor the status of their instance(s). Once created, an OS instance comes up and can almost immediately be accessed via one of the ssh keys associated with the student's AWS account.

As an instructor, I was surprised at the ease with which students adapted to using the cloud. The turnaround time of just a few minutes from requesting a new OS instance to being up and running and ready to log in still seemed fast to me, while students quickly accepted this as a convenience to be relied on. Only a few students struggled initially with the concepts of different firewall rules and ssh keys that one can associate with one's OS instances. The overall quick adoption of the new tools allowed me to be much more flexible in creation of assignments, many of which were adapted, changed, or scaled during the course of the semester as the capabilities of the resources and the students' own experience became clearer.

One assignment, for example, required students to compare a number of different package management systems on a few different OS flavors (each system's tools native to the OS as well as a single cross-platform system [5]). The lessons learned included not only an appreciation of the differences (and similarities) between software package management systems and an understanding of the intricate complexities of software dependencies, but also nicely illustrated how different libraries provided by different OSes can influence the compilation process of the same set of sources in, shall we say, "interesting" ways. Without root access on four different UNIX flavors (for this exercise alone), these lessons would have been abstract and theoretical, if understood at all.

Another assignment's objective was for students to learn how to use tcpdump(1) to observe and analyze DNS-related traffic. Students had to set up one instance as a DNS resolver and another as a client pointing to the server. While running tcpdump(1) on both hosts, they could then observe how one query from the client caused the server to first contact one of the root servers, then one of the nameservers responsible for the TLD, for example, before returning the results; a subsequent query was observed to not leave the DNS server, as it had cached the results. The detailed analysis of the tcpdump(1) output—graphical analysis tools such as Wireshark were explicitly forbidden—caused the students to really understand how to read network packets, what kind of lookups are done, and when a server replies with what kind of information. Previous lectures on the topic of the DNS hierarchy proved to me that experience is necessary to illustrate clearly to students

what happens behind the curtain for almost any and every network connection initiated on a host. It became clear that only the actual inspection of network packets on a DNS resolver as they're generated really reached the students.

From an instructor's perspective, it quickly became clear to me that the availability of a fresh, new OS instance to test each student's project as well as my own sample solutions was invaluable, but something more was lurking on the horizon. The fast turnaround time makes it feasible to spin up an instance to, for example, quickly test a program for cross-platform compatibility without having to run your own (energy-wasting and mostly idle) server farm for this purpose.

Cloud Limitations

From the above, it should be obvious that I'm quite enthusiastic about using Amazon's EC2 service in order to teach system administration adequately. Most of the practical assignments would not have been possible (to the same degree or at all) without the resources provided in this way; all of the assignments in the cloud have produced significantly better understanding on the students' part than previously assigned similar exercises. However, as much as our industry would make it seem, "the Cloud" is no panacea: there are limitations and downsides.

A slight disappointment was that at the time of this writing, Amazon did not offer IPv6 networking capabilities in EC2, something I had hoped to be able to expose students to in practice. Our school did not provide IPv6 connectivity itself, either. Using a tunnel broker would require the use of so-called "elastic IPs" for the students' instances and might thus be possible, although unfortunately we did not have the time to pursue this in the last semester.

In previous iterations of my class, I made students perform an OS installation from scratch, by hand (i.e., without the use of either a GUI or a menu-driven install program). This has always been a great opportunity for students to learn to understand the concepts of disk partitions, filesystem creation, boot loaders and boot order, OS set installation and initial configuration that previously had only been discussed in class and normally are hidden from the end user by the installer.

The virtualization of the OS and abstraction of the instance creation process made it impossible for students to execute these steps. However, a different opportunity awaits: if time and students' prerequisites permit, it ought to be possible to allow them to construct a custom Amazon Machine Image (AMI), an exercise that would convey valuable experience and understanding of rapid deployment systems and abstraction of system components in a virtualized compute world.

Finally, a big challenge lies in the fact that an OS instance has a limited life—when it is shut down, it ceases to exist, and any data residing on the instance is lost. Hence, for the most part, students configured their hosts but "lost" all their work when they shut it down or rebooted it. This can be overcome by making use of Amazon's Elastic Block Storage, which offers off-instance, persistent storage, but while not particularly difficult to use, this is another hurdle and part of the learning curve when using "the Cloud." More than once students reported that they had completed their assignment but had to redo it after accidentally shutting down a host. Frustrating as that was for the students, it was satisfying for me to hear that they also reported that the second time around was much easier, implying that they had internalized some of the lessons as intended.

Each of these problems and limitations does, it should be pointed out, offer an opportunity for students to expand their understanding of cloud computing, of how services, OSes, and host instances are seen in a virtualized world, and how the workarounds or solutions create other interesting possibilities.

As noted above, system administrators are creative. Hence, it comes as no surprise that we should be able to determine viable alternatives to using one company's commercial service. In the end, as so often, it boils down to being a trade-off: each alternative has a cost, just as the original solution does. Will the cost (explicit or implicit) be worth the (perceived or actual) gain?

At the beginning of this article, I mentioned some of the disadvantages of more traditional solutions: isolated labs create an environment that can only approximate real networks; restricting assignments to resources already available on campus necessarily removes some of the flexibility the instructor has in choosing them; and developing or providing more diverse or flexible laboratories or restricted services incurs the cost of many work-hours on the IT staff.

If a school is willing to invest in making available a virtualized environment to students by using Xen, VMware, or any of the other technologies, then it ought to be possible to offer many of the same benefits as Amazon's EC2 environment. However, the maintenance overhead of such facilities is non-negligible and is still likely subject to network access control restrictions (inbound and/or outbound). At the same time, ad hoc creation of OS instances by students themselves as needed requires a high degree of automated virtualization maintenance, which may not always be available.

Based on this, I do believe that outsourcing the infrastructure maintenance is an appealing solution. At the same time, I consider it important to expose students to emerging technologies and what already has become, to some degree anyway, an industry standard solution: teaching system administration in the cloud includes lessons on system administration *of* the cloud. The experience gained this way is not restricted to one vendor's product, either: not only are the underlying concepts illustrated by the practical experience useful when approaching other cloud-specific solutions, there exist AWS-compatible open source implementations such as Eucalyptus [6]; many companies even base their own cloud strategy on these solutions and are actively looking to hire new talent with experience in this area.

Conclusion

"Learning by experience," valuable and irreplaceable as it is, requires two things: time and opportunity. We cannot provide students with more time—learning to appreciate true scalability can only come with years of experience in a number of different environments. By sowing the seeds of a different, deeper understanding of the practical concepts of system administration, by letting students gain hands-on experience actually creating and maintaining Software as a Service (SAAS), by exposing them to scalable and elastic Infrastructure as a Service (IAAS) and letting them see for themselves how, underneath, such services are maintained or created, we can offer the opportunity to learn system administration in a new and very different way.

Teaching system administration in (and of) the cloud will be a core element of my own class in the future. Whether that will happen in Amazon's EC2 implementation or through other resources is largely irrelevant. As usual, it's the underlying principles that count, and in many ways throughout the last

semester the teacher was as much a disciple as the students, enjoying the process along the way.

REFERENCES

[1] Course Web page: http://www.cs.stevens.edu/~jschauma/615A/.

[2] Sample command-line tool invocation: http://www.usenix.org/publications/login/2010-10/pdfs/cli.html.

[3] Amazon's AWS in Education program: http://aws.amazon.com/education/.

[4] Amazon's EC2 Getting Started Guide: http://docs.amazonWebservices.com/AWSEC2/2009-11-30/GettingStartedGuide/.

[5] pkgsrc: http://www.pkgsrc.org.

[6] Eucalyptus: http://www.eucalyptus.com/products/overview.