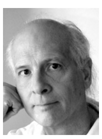
OPINION

Musings

RIK FARROW



Rik is the editor of *;login:*.
rik@usenix.org

I just finished reading about how humans managed to survive, just barely, a robot revolution [1]. A quick and easy read, it was just what I needed after immersing myself in file systems and the cloud. And fortunately, after a grim beginning, the book focuses more on the puny humans' successes than on their failures.

We already live in a world where robots are becoming more common. Some examples, such as factory robots, are obvious. But others, such as Google's self-driving Prius [2], don't look at all like robots. But then, neither do factory robots look at all like Robbie the Robot, nor do the robots that help to care for old folks that are gaining acceptance in Japan.

Thinking about robots, not the virus-infected bloodthirsty type, got me to thinking about automation and autonomous systems. For several years, autonomic systems were the rage in some parts of the sysadmin world, while being anathema in others. Many sysadmins felt that bringing autonomy into their networks would make the sysadmin redundant. That won't happen, any more than the fifth-generation programming languages [3] managed to do away with programmers back in the 1980s.

Automation

If we go back to the 1950s, we have early examples of computerized automation that go well beyond card-punch readers. ADP, for example, was selling Software as a Service (SaaS) by 1957, with their payroll-processing business. ADP quickly expanded to offer other services, all SaaS.

To stick to something more in the world of system management, I began managing multiple systems in 1985 and used UUCP [4] to share data between these systems. Each system was installed independently, configured independently, and updated independently. UUCP was a serial line protocol, and I had wired the office where I consulted with lots of telephone wire terminating with RS-232 connectors. Any change, such as adding a user or a new host, involved manually editing files on each existing host.

After Ethernet arrived, at a blistering 10 megabits/second, we had real networking at last, but still no automatic administration. We could use Sun's Network Information Service, but only on the Suns.

But as TCP/IP networking evolved, we began to see what might not at first look like automation, but certainly worked that way. DNS is a great example, as it replaced fetching and installing the /etc/hosts file with a system where updates appeared almost immediately. Having MX records really helped deal a death blow to UUCP's bangpath addressing, where you needed to know every host along the path if you wanted to send email to someone.

DHCP is another wonderful example of network automation. Instead of having to configure each system with an IP address, DHCP does that for you, and more. Note that IPv6 does support automatic address assignment, but fails to provide the other key features that DHCP currently serves (such as host and domain name, and the addresses of key network services).

In 1994, Sun Microsystems introduced Jumpstart: instead of installing each system, you used network boot and network installation of computers. We now have similar systems, such as Kickstart, for installing Linux systems.

None of this automation put sysadmins out of work, as far as I know.

The Cloud

While ADP got an early start as a "cloud" provider, almost 50 years before the buzzword became popular, the cloud is a real game changer today. I attended the High Performance Transaction Systems (HPTS) workshop in October (see the reports in this issue) and heard many fascinating talks about how the ability to host your applications with various cloud providers is transformative. Yes, this too has to do with automation.

Instead of gangs of sysadmins installing system images, we now have Web interfaces that allow us to point-and-click our way through the process. There are more elaborate versions of this, such as the offerings of VMware, which allow you to build and control your own clouds. There are also custom interfaces, built on top of cloud providers' interfaces, that make installing your system images, complete with your own applications and data, as easy as filling in Web forms.

Adrian Cockcroft, during HPTS, told the fascinating story of how Netflix has migrated away from datacenters using SQL databases into the cloud [5]. Adrian described several reasons for the migration: reliability, flexibility, and cost. That a cloud-hosted service could be more reliable than a traditional datacenter service surprised me. But if you read more about the Netflix story, you too will begin to understand how this works.

The NoSQL (not-only SQL [6]) databases also support a movement to both clouds, and to farms of low-cost servers. NoSQL-style databases are good when you need to scale beyond what can be done with a single server, no matter how many processors, gigabytes of RAM, and SANS you throw at the problem. HPTS also had a debate contrasting scaling up (the SQL database model) with scaling out (the farm of servers running one of many varieties of NoSQL).

Robots and Jobs

Today, people roam the hot and noisy aisles between racks in datacenters, replacing hard disks, power supplies, and entire systems. Datacenters are noisy environments, and not the nicest places to work. As James Hamilton (Amazon) said during his HPTS talk, if your datacenters look nice enough that you invite people to see them, you have paid too much for your datacenter [7]. It seems to be that datacenter system wranglers might someday be replaced with robots.

Robots would also make much safer drivers, instead of the multi-taskers we have today who reserve most of their attention for the person on the far end of a cell phone connection.

The Lineup

I was also able to attend the SNIA Storage Developers Conference [8] this September, where I learned some amazing things that I might have missed had I not gone. Even though SDC is long over, I did ask several people to write about some of the topics covered that I felt would be relevant to USENIX members.

First up, Josef Bacik (Red Hat) explains Btrfs. I had first heard about Btrfs (pronounced "butter FS") four years ago, during a Linux File System workshop running alongside FAST. At the time, I thought that Btrfs would be the Linux clone of Sun's ZFS, but, if anything, Btrfs is a new file system that resembles Sun's, but with simpler administration and the features of ZFS. When I learned that the "Btr" in Btrfs referred to B-trees (as used in Mac file systems), I asked Josef to include an explanation of B-trees as well. Knowing that Btrfs uses B-trees for almost everything makes developing Btrfs easier and administering it easy, and it helps you decide which of your file-system needs best match Btrfs instead of ext4 or XFS.

I didn't meet Rob Chansler at SDC, but he too has a filesystem-related article. Rob has worked on HDFS development at Yahoo!, and wanted to share his experience with very large HDFS installations. HDFS is a distributed file system designed to work with Hadoop, and Yahoo! is not just the initial developer of Hadoop but one of its larger users as well.

I encountered Chris Hertel in the hallway at SDC, shortly before he was scheduled to speak. I had met Chris many years before in Minneapolis, where I was teaching Linux security. While catching up with Chris, I learned that he is a Samba developer, had written the book on the SMB protocols, and had become familiar with SMBv2. SMB, Microsoft's file sharing protocol, had pretty much stagnated for many years. SMBv2 has changed that, with many times the performance, and many of the features one expects to find in modern file-sharing services. Chris not only explains the advantages of using SMBv2 but also shows Microsoft in a new perspective which I think will surprise you. It certainly did me, as did the live demonstration of SMBv2.2 beta between a bunch of disks over bonded channels with automated failover.

Alex McDonald (NetApp) finally provided something that I had long searched for: an explanation of NFSv4, along with motivation for moving to it from NFSv3. NFSv4 has been around for years now, but adoption has been glacial. Too many people have spent too many years learning how to deal with the eccentricities of NFSv3 and are not anxious to try something new. As you read Alex's article, you will realize that some of the features appearing in beta in SMBv2.2 have long been a part of NFSv4. NFSv4 makes managing NFS simpler, its security better, and its performance much faster, as well as adding support for HPC. Alex's article also contains a link to his SDC white paper on migrating to NFSv4 from v3.

I also met Stuart Kendrick at SDC, although he was not there as a presenter. Stuart also learned about the improved performance of SMBv2, and this appears in his article. But the focus of his article is not SMB, but discovering where the problem might lie in client-server transactions. Stuart describes, and provides scripts for, slicing up client-server pie, a nice visible method for seeing whether it is the client, the server, or the network, that is responsible for bad performance. Note that Stuart has written for *;login:* before.

I met Adrian Cockcroft (Netflix) at HPTS. Adrian was very excited about Netflix's move from big iron in datacenters to Amazon's AWS and EBS cloud services. I talked with him during the reception, listened to his HPTS talk, read his other online slide presentations, and read his Netflix blog. And I still had more questions, which Adrian took some time out of his crazy-busy schedule to answer.

David Blank-Edelman provided me with some timely help. David discusses Perl modules for manipulating spreadsheets and CSVs (Comman Separated Values). I was very happy about this, as someone had just dumped an Excel spreadsheet in my lap, suggesting that I extract some useful data from it. Somehow I have never learned much about spreadsheets, including how to extract just two columns from many rows. David also explains, for the Perl-inspired and spreadsheet-averse, how you can write to spreadsheets as well—just what I needed.

Dave Josephsen continues with the explanation of Graphite that he started in the December 2011 issue. Graphite is one of a trio of tools for monitoring a large number of systems, and one that Dave waxes enthusiastic about because it fulfills many of the items on his long list of features desired in distributed monitoring systems. Dave explains how to integrate Graphite with Nagios and Ganglia, as well as linking Graphite up to a number of other packages useful in analyzing monitoring data. Very cool.

Dave Beazley presents his first *;login:* column, with a reprise of his earlier *;login:* article that covers Python 3. Dave has many years of experience writing and teaching Python, and has plans to include columns about data analysis, libraries, the language itself, and system programming. In this column, Dave focuses on features of Python 3 that he particularly appreciates, contrasting them with how they might have appeared in Python 2.

Robert Ferrell muses about the process of writing columns, especially when he can't find his muse.

Elizabeth Zwicky reviews a pile of books, covering skills for software architects, five big data titles, and programming Pig. Sam Stover covers two books, one about MongoDB with Python, and the second a different take on *Privacy and Big Data*, a book Elizabeth also reviewed.

We conclude this issue with reports from the HPTS workshop. We worked on producing these reports because we felt that many of the issues addressed in this workshop would be relevant to USENIX members.

While I was searching for a link to Wilson's book about the robot revolution, I learned that in a year or so Steven Spielberg will have a movie out based on the book. I am sure the movie will have lots of cool special effects, and likely much better character development than Wilson's book had (after all, Wilson got his PhD in robotics), but it really was a fun read, even if the robots in it were not just murderous, but a little like Nazis from the 1940s.

We will certainly be living in a world with many more robots, ones that will perform work that human beings generally dislike doing. But that also begs the question: once robots have replaced factory workers, stoop labor, seamstresses, and even system wranglers, how are all those people going to make a living? When the problem was *just* factory workers, with many of those jobs going to poorer nations, the white collar workers weren't too worried. But those days may soon be over. Or, to put it another way, when was the last time you called a business and a human answered the phone? The robots are taking over.

Good thing most of them don't have guns [9].

References

[1] Daniel Wilson, *Robopocalypse* (Doubleday,. 2011).

[2] Google's self-driving Prius: http://www.greencarreports.com/news/1067485_how-googles-self-driving-car-works.

[3] Fifth-generation programming language: https://secure.wikimedia.org/wikipedia/en/wiki/Fifth-generation_programming_language.

[4] UUCP Project: http://www.uucp.org/info.html.

[5] Adrian Cockcroft, "Migrating Netflix from Datacenter Oracle to Global Cassandra" : http://www.slideshare.net/adrianco/migrating-netflix-from-oracle-to-global-cassandra.

[6] Greg Burd, "NoSQL," *;login:*, vol. 36, no. 5: http://db.usenix.org/publications/login/2011-10/openpdfs/Burd.pdf.

[7] Photo tour of the Facebook datacenter: http://scobleizer.com/2011/04/16/photo-tour-of-facebooks-new-datacenter/.

[8] SNIA SDC 2011: http://www.snia.org/events/storage-developer2011.

[9] Samsung SGR-A1: https://secure.wikimedia.org/wikipedia/en/wiki/Samsung_SGR-A1.