Musings

RIK FARROW

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

In 1983, I wrote one of my first attempts at fiction: a day in the life of a system administrator. My intent was to provide examples of what system administrators of UNIX systems typically did: boot the system, check logs, solve problems with printers or full disks, update software, and back up the system. These tasks didn't require a full-time position as there was just one system to manage. That was my vision at the time, and you can see my story at [1], scanned from the documentation I wrote.

When I started writing my sysadmin book, I did more research. My first sally into the field had been based on documentation I was asked to write for systems that were being sold to organizations that needed a way to have multiple people share that same system, using terminals. I visited UC Berkeley, where they had four VAXen running BSD, and talked to sysadmins there, who had no idea they were acting as sysadmins. They were students tasked with freeing up disk space, unclogging print queues, backing up, adding or deleting users, in other words, pretty much what I had imagined sysadmins would be doing. My vision of the world of sysadmin seemed well-aligned with reality.

For many years after that initial research in 1985, the world of the sysadmin remained mostly unchanged. Sysadmins were very often chosen from the ranks of Liberal Arts majors (although no college education was required) because they expressed an interest and willingness to work with UNIX systems or, perhaps, because they were drafted unwillingly to work on computers, especially anything as arcane as UNIX appeared to be. Not that UNIX had the corner on weirdness. Based on my own experience by that time, UNIX was refreshingly consistent and easy to use compared to the Microsoft or IBM systems of the day.

Things had begun to change by the late '80s, and that change was marked by the creation of the LISA conference. LISA, or Large Installation System Administration, was founded as a conference to help those who had so many systems to manage that the older methods of doing so, sitting at each console and typing away, were no longer sufficient. Over time, the need for automation led to infrastructure as code, beginning with tools to manage configurations like CFEngine and NIS [2].

Over the next decade, workstations multiplied like bacteria, covering desktops at most organizations. These fell into two classes: UNIX systems, because they supported networking, and PCs running Novell Netware, as Windows didn't support either file sharing or networking until 1996. Novell featured centralized administration, while those managing fleets of UNIX workstations had to get creative, and often did so by building their own set of tools. You could say that we had whole networks of pets (as opposed to cattle [3]) in the '90s, and each set of pets was ruled by idiosyncratic tools, largely unportable to other organizations.

Sea Change

What changed everything was yet another startup: Google. Larry Page and Sergey Brin wanted to create an efficient way to index the Internet. Companies had built indexes mostly manually up to that time, meaning that you searched through these indexes hoping they might lead you to the information or product you were really searching for. Page and Brin had

another idea and that was to actually canvas the web, collect all the pages they could find, then index and rank those pages based on the data the pages contained and the links that referred to those pages.

Besides needing serious network bandwidth, Page and Brin also needed a lot of computing horsepower, and that was seriously expensive. Their approach was to divide up the task so that a collection of computers could do the job, creating a form of parallelism that was fairly new at the time and terribly common today.

As Google grew, so did their clusters of computers. The service provided by those clusters also grew over time, so different clusters would be providing different services. But the clusters themselves were designed to be pretty interchangeable from the start. And managing those clusters of Linux systems had also moved very far away from being the system administrator of a bunch of desktops and a handful of servers. Google had invented cattle.

Ben Treynor Sloss, VP of Engineering at Google, invented the term Site Reliability Engineer (SRE). Ben had started out as a developer who got moved into operations in 2003, and decided to run his operations team as he would a developer team. Ben also came up with other important concepts, such as the error budget. That is, if your service-level objective (SLO) is 98.6%, that remaining 1.4% is your error budget: the amount of time your team has for updates or handling service outages.

There's much more to being an SRE, and one of the most important concepts, in my opinion, is eliminating toil. Toil is repetitive work that can be automated away, and SREs are supposed to spend no more than 50% of their time on operations so they can spend the rest of the working time on automation. As Sloss has mentioned, as you grow, your operations may scale exponentially. But your operations staff cannot scale exponentially. You must automate.

Even as SRE concepts became more popular, there has been a lot of pushback: not all organizations are going to be like Google, Facebook, and LinkedIn, to name a few. But what are most organizations today doing with their computing infrastructure? They are moving to the cloud, and if they expect to scale up their operations, they too will need to behave more like organizations with SREs.

The Lineup

We open this issue with three articles based on papers presented at OSDI '18. There were many more papers at OSDI of course [4], but I picked these because I liked them and thought they would be of broad interest to USENIX members.

Cody Cutler, Frans Kaashoek, and Robert Morris wrote an experimental operating system using Go. Their original goal had been to see whether language features would be an aid in OS writing, but the project pivoted toward seeing whether a high-level language, one with garbage collection, could run popular applications as fast as Linux could.

The next two articles include open source projects that support running services in clouds or clusters. Ana Klimovic et al. created Pocket as a means for ephemeral storage. Services that need to quickly store short-lived objects, such as Spark, are poorly served by the current mix of cloud storage, like S3. Pocket manages a range of storage services that are both faster and cheaper to use than current offerings.

Jon Gjengset et al. wrote Noria, a database server with an SQL front end that is not only faster than existing servers, like MySQL, but also supports much higher loads. Noria is a dataflow processing system that creates graphs where the vertices are operators and edges carry updates. Noria is slower to write but much faster for reads, and fits best when applications have read-heavy mixes.

While at LISA18, I heard several people talking about boring tech. That sounds, well, boring, but it's actually about keeping your architecture as simple as possible. I met Dave Mangot, who had presented "Familiar Smells" [5] and stirred up a fair bit of controversy. Dave agreed to explain his points about how important it is to architect your systems and services so that they are as simple as possible.

Laura Nolan volunteered to write a column about operational debt. You probably have heard of technical debt. Laura compares technical debt to credit card debt, but likens operational debt to a mortgage. Operational debt has to do with having failed to automate as much of operations as possible and instead having to waste most of one's time on toil.

Sergey Babkin offered to write about his experience interviewing people for mid-level programming positions. Sergey's thesis is that when it comes to solving the programming problems that often are used during interviews, he sees people using two different approaches. Each approach has its strengths, but they are best used together rather than in isolation.

Nisha Talagala, Bharath Ramsundar, and Swami Sundararaman wrote about the new, one-day OpML conference happening in May 2019. With the huge surge in interest in machine learning (ML), they discovered that just as there is a need for AI specialists, there's also a growing need for people who can run ML at scale. ML involves not just AI, but also working with vast amounts of data as well as other production issues.

Peter Norton explores issues with `zip`, a function used to create iterables in Python. Peter had been stretching his skills using the Advent of Code and, while solving one of the problems, uncovered a weakness in how `zip` works. His workaround, `SnitchZip`, is simple, but it won't be replacing `zip`.

EDITORIAL

Musings

David Blank-Edelman is retiring his column after having written it 66 times. At least that's what his final Perl program has discovered. We thank David for being so generous with his time over the last 12 years, and sharing his approaches to problem-solving with a Perl flair.

Mac McEniry shows us how to execute commands from within a Go program. Mac breaks down the potential usage into groups, depending on input and output, and whether to wait, forget, pipe-in, check out, or replace data.

Dave Josephsen expands on the monitoring system for mail processing at Sparkpost. In Part 3, Dave focuses on detecting backed-up queues within Fluentd, as well as staying with his theme on rivers and flow. With the sewers of Paris backing up, the flows aren't so fragrant.

Dan Geer and Scott Guthery examine the patterns of patents granted that relate to cybersecurity. Over time, the preponderance of new patents has shifted from the US and Europe to Asia, even as the general topics of security-related patents has changed over a period of decades.

Robert Ferrell discusses the reality of ubiquitous computing. With everything from online doorbells to toilets, Robert still manages to leave out automotive systems like OnStar that tell GM every time you accelerate or stop too quickly. But like the systems Robert describes, all of the data gathered is for the use of our corporate overlords.

Mark Lamourine has written three book reviews, covering managing Kubernetes, learning Git, and using "gamestorming." I reviewed *The Site Reliability Workbook,* the follow up to *Site Reliability Engineering*.

In USENIX Notes, I interview Liz Markel, the new Community Engagement Manager. You will be seeing Liz, often with her camera handy, at future USENIX events.

System administration has morphed from managing single servers to riding herd on fleets of cattle. While there will always be pets, especially in organizations that are naturally disposed to being fiefdoms, like many universities, the world has changed. To be honest, I think many people are glad that they don't have to design their own systems for fleet management and that the tooling has become so much more powerful over time.

References

[1] A Day in the Life of a System Administrator: https://rikfarrow.com/sysadminday.html.

[2] O. Kirch and T. Dawson, "The Network Information System," Chapter 13 in *Linux Network Administrator's Guide, 2nd Edition* (O'Reilly Media, 2000): https://www.oreilly.com/openbook/linag2/book/ch13.html.

[3] R. Bias, "The History of Pets vs Cattle and How to Use the Analogy Properly," September 29, 2016: http://cloudscaling.com/blog/cloud-computing/the-history-of-pets-vs-cattle/.

[4] OSDI '18 Technical Sessions: https://www.usenix.org/conference/osdi18/technical-sessions.

[5] D. Mangot, "Familiar Smells I've Detected in Your Systems Engineering Organization and How to Fix Them," LISA18: https://www.usenix.org/conference/lisa18/presentation/mangot.

28TH USENIX SECURITY SYMPOSIUM

SANTA CLARA, CA, USA

Co-located Workshops

WOOT '19

13th USENIX Workshop on Offensive Technologies
August 12–13, 2019
Submissions due May 29, 2019
www.usenix.org/woot19

WOOT '19 aims to present a broad picture of offense and its contributions, bringing together researchers and practitioners in all areas of computer security. Offensive security has changed from a hobby to an industry. No longer an exercise for isolated enthusiasts, offensive security is today a large-scale operation managed by organized, capitalized actors. Meanwhile, the landscape has shifted: software used by millions is built by start-ups less than a year old, delivered on mobile phones and surveilled by national signals intelligence agencies. In the field's infancy, offensive security research was conducted separately by industry, independent hackers, or in academia. Collaboration between these groups could be difficult. Since 2007, the USENIX Workshop on Offensive Technologies (WOOT) has aimed to bring those communities together.

CSET '19

12th USENIX Workshop on Cyber Security Experimentation and Test
August 12, 2019
Submissions due May 21, 2019
www.usenix.org/cset19

CSET '19 invites submissions on cyber security evaluation, experimentation, measurement, metrics, data, simulations, and testbeds. The science of cyber security poses significant challenges. For example, experiments must recreate relevant, realistic features in order to be meaningful, yet identifying those features and modeling them is very difficult. Repeatability and measurement accuracy are essential in any scientific experiment, yet hard to achieve in practice. Few security-relevant datasets are publicly available for research use and little is understood about what "good datasets" look like. Finally, cyber security experiments and performance evaluations carry significant risks if not properly contained and controlled, yet often require some degree of interaction with the larger world in order to be useful.

ScAINet '19

2019 USENIX Security and AI Networking Conference
August 12, 2019
Talk proposals due March 28, 2019
www.usenix.org/scainet19

ScAINet '19 will be a single track symposium of cutting edge and thought-inspiring talks covering a wide range of topics in ML/AI by and for security. The format will be similar to Enigma but with a focus on security and AI. Our goal is to clearly explain emerging challenges, threats, and defenses at the intersection of machine learning and cybersecurity, and to build a rich and vibrant community which brings academia and industry together under the same roof. We view diversity as a key enabler for this goal and actively work to ensure that the ScAINet community encourages and welcomes participation from all employment sectors, racial and ethnic backgrounds, nationalities, and genders.

FOCI '19

9th USENIX Workshop on Free and Open Communications on the Internet
August 13, 2019
Submissions due May 23, 2019
www.usenix.org/foci19

FOCI '19 will bring together researchers and practitioners from technology, law, and policy who are working on means to study, detect, or circumvent practices that inhibit free and open communications on the Internet.

HotSec '19

2019 USENIX Summit on Hot Topics in Security
August 13, 2019
Lightning talk submissions due Monday, June 10, 2019
www.usenix.org/hotsec19

HotSec '19 aims to bring together researchers across computer security disciplines to discuss the state of the art, with emphasis on future directions and emerging areas. HotSec is not your traditional security workshop! The day will consist of sessions of lightning talks on emerging work and positions in security, followed by discussion among attendees. Lightning talks are 5 MINUTES in duration—time limit strictly enforced with a gong! The format provides a way for lots of individuals to share ideas with others in a quick and more informal way, which will inspire breakout discussion for the remainder of the day.

Registration will open in May 2019.