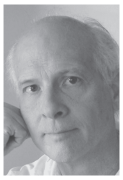
OPINION

Musings

RIK FARROW



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

It's a question of balance [1]. At least, that's how I see things as I look over the collection of articles in this month's issue.

We all need balance: balance in our lives, between work and play, between waking and sleeping, exercise and rest, eating what we like and staying healthy. Computer systems require balance too.

Leading off, Jeremy Elson and Ed Nightingale describe Flat Datacenter Storage, based on their OSDI 2012 paper. I liked Jeremy's presentation in Hollywood, and asked him to reprise that presentation as an article. But as much as I liked his presentation, I also liked the balance presented in the design of FDS.

At first glance, FDS is just another distributed storage system, like HDFS. But if you look closer, you can see evidence of the balance that is the theme of this issue. FDS uses hashes to distribute *tracts* (extents) evenly throughout servers. Metadata is also distributed evenly, via a similar mechanism. But the real balancing act of FDS has to do with the design of the network. Instead of hooking up servers to top-of-rack switches, and these switches to other switches or routers, the designers of FDS sought a balance between network bandwidth at both clients and servers and the capability of a client or server to consume or produce data. They spent extra money on network infrastructure so that there are no bottlenecks to slow down performance.

And did this level of balance make a difference? FDS can replace a lost drive containing a modest amount of data (92 GB) in 6.2 seconds if the cluster includes 1,000 disks. And using the MinuteSort benchmark [2], the FDS replaced Hadoop as the fastest at generic sorting and TritonSort at specialized (Formula 1) sorting.

I think there are lessons to be learned with FDS, which is really why I asked Jeremy and Ed to write for this issue.

How Hot Is Too Hot?

El-Sayed et al. take on a different type of balance. Most datacenters (DCs) choose a conservative setpoint for the cold side of the aisles. This is based on conservative information provided by server vendors. After all, the goal of a DC is to process information, not lose servers and disks because of too high temperatures.

Through the analysis of vast amounts of data, and their own tests using a heat chamber, these researchers found that the balance between too cold (spending more on cooling) and too hot (having to replace failed components) can shift toward warmer DCs. As cooling DCs represents a large proportion of their energy budgets, being able to adjust the setpoint higher saves energy, money, and perhaps the planet as well.

Along the way, El-Sayed et al. disprove some of the long-held assumptions about the effects of temperature on the reliability of disks and memory as temperature levels increase.

Michael Adam, a Samba developer, provides us with deep insight into where Samba is today and its path forward. The Samba project has gone from leading Microsoft, with the first version of clustered SMB storage, to playing catch-up today. Fitting this into my theme of balance is a bit more difficult, but when you consider that Samba had advanced the state of SMB storage, and is now playing catch up, I can see this as an attempt to recover the balance between Samba and SMB.

I interviewed Allen Wittenauer of LinkedIn, both a Hadoop committer and large scale cluster operator. I had been curious about how one goes about managing Hadoop clusters, and Allen was just the person to talk to. Allen provided information about both how to monitor a Hadoop cluster and how to manage and improve the performance of Hadoop tasks. So where's the balance? It turns out that there is a balance here too, between taking the macro overview and drilling down to see what is happening on individual systems.

Jeanvoine et al. present Kadeploy3, a system for installing OS images on grids. They have been developing this system for years, and have successfully tested it on grids with over a thousand nodes. For balance, they describe how they decided on the best way to utilize the network bandwidth for the quickest installation.

Thomas Barr and Scott Rixner present their Python development toolchain for ARM Cortex-M3 embedded controllers. With their open source toolkit, developing software for embedded systems becomes almost as easy as Python programming: you get to use Python, you have access to the special libraries provided by the provider of the system-on-chip, and you also get instant feedback. Barr and Rixner do mention balance, in the sense of the comparison of the expert embedded systems programmer versus the rest of us. With tools like Owl (their development toolchain), we don't need to be experts to get work done, while experts are still needed to create the libraries required by Owl.

Out of Balance

So much for balance. Or perhaps I could say that for balance, the rest of the issue doesn't follow the balance theme?

David Blank-Edelman continues his journey of showing us how to use various Web-based APIs, focusing this time on a service that can send and receive texts, make voice calls, and collect dial button presses as part of a survey, or automated call center.

David Beazley set out to scare his readers by diving deeper into the arcana of the Python import mechanism. In an earlier column, David explained how you can set the import path, but this time he shows us tricks for changing what actually happens when you attempt to import a module into your Python program. Not to worry, concludes David, as this will be rationalized with version 3.3.

Dave Josephsen continues his coverage of XI, the GUI and database extension to Nagios. Dave explains the various ways a group of sysadmins can hose each other's work when using different methods of administering Nagios when only one uses XI.

Robert Ferrell shares a blast from the past. Robert depicts a UNIX "expert" with advice that might send shivers, perhaps of annoyance, down your spine.

Elizabeth Zwicky reviewed five books for this issue, including a second edition of a book on regular expressions, two Python books (data analysis and Python for kids), a book on managing programmers, and finally another kids programming book. Coincidently, Mark Lamourine reviewed the same book, *Super Scratch Programming Adventure!*, and I'm including both of them, as the perspectives of the authors, and the ages of their testers, differ. Mark also reviewed a book on assembly language programming for ARM, helping to balance out the focus on Python. Trey Darley reviewed two books, one on insider threats, and the other on Internet protocols.

We also have three sets of summaries from OSDI in this issue: the conference itself, and two workshops, MAD and HotPower.

As the editor of *;login:*, I always strive for a balance between materials that I believe are of most interest to the greatest number of readers, and articles that have something to teach. To me, the process of research is one of learning, composed of new ideas, implementation, and trial and error, and that's something that we can all learn from.

References

[1] A Question of Balance: http://en.wikipedia.org/wiki/A_Question_of_Balance.

[2] Sort Benchmark: http://sortbenchmark.org/.