RIK FARROW

musings

rik@usenix.org

THERE ARE TIMES WHEN WE JUST can't wait for the future to arrive, such as the coming of warmer weather. And sometimes it seems that people pine for the poorly remembered past, as if it were somehow better than what we face today. Right now, I want to talk about sysadmins and ponder whether they are looking ahead while wishing for an imagined past.

In this issue you will find the summaries for LISA '07, including the summary I wrote about John Strassner's keynote. John spoke about experiences with a project at Motorola where researchers had created a functioning example of network autonomics. This is a complex system, with many different active components all contributing to decisions that result in changes in configuration. The FOCALE architecture (see slide 23 of his presentation on the LISA '07 page [1]) has a Context Manager, a Policy Manager, and an Autonomic Manager, as well as a machine learning component, all of which are involved in controlling the creation and modifications of device configurations.

FOCALE is a working system. It actually helps to simplify a terribly complex control setup that includes seven different groups of administrators (see slide 4). John carefully began his talk by explaining the existing situation found in many telecommunications companies (think cell phone operators). He explained the limitations of the current network management, including the need for human involvement in analysis before anything can be done. And he described what he means by autonomics, going way behind the infamous four self-functions of self-configuration, self-protection, self-healing, and self-optimization made famous by IBM [2, 3]. John considers these benefits, seeing the way forward via knowledge about component systems, the context in which they operate, and an ability to learn and reason, to follow policy determined from business rules, and to adapt offered services and resources as necessary.

I thought John's talk described groundbreaking research, where a real autonomic system was working to make a network function more smoothly. But others at the conference weren't nearly as happy. The most common complaint, one that really stuck with me, was that there was "too much math" in his solution. I wondered whether the *two* equations found on slide 47 (shades of calculus!) were to blame. But then I read Alva Couch's article

(page 12) and realized that perhaps the real problem was something completely different. The real problem has to do with two things: a mindset, and being stuck in the past.

The Mindset

Alva Couch explains something I have had difficulty understanding since I first encountered the concept, way back when I was a college student. I found languages such as FORTRAN and ALGOL easy to comprehend, but LISP and APL unpleasant to use. I've recently learned that there is a "semantic wall" between these two languages, to borrow from Alva. Using a more modern example, the C language is bottom-up, or imperative. You write a sequence of commands, and they are executed in order. Functional programming languages, such as LISP or Haskell, work top-down, where the entire program is a single expression. In a functional language, the expression describes the desired result without specifying how the result is derived.

Now consider how system administration gets accomplished today. Someone requests a change to a service, and sysadmins go about changing the configuration, an imperative operation. If something breaks, the sysadmins set about uncovering the cause of the problem and adjusting the configuration to solve the problem—again, a bottom-up approach.

What John Strassner, and Alva Couch, suggest requires a mindset that is very different. Instead of acting imperatively, getting right into the nitty-gritty of configuration editing, autonomics requires a more functional, top-down approach. I believe that a lot of sysadmins will find this approach inimical to the way they have carried out their duties for their entire working careers.

And thus the past, in which we do things the way they have always been done, becomes an obstacle to a future where some things will need to be done differently. We are really not that different from people riding horse-drawn carriages in 1907 complaining about the noisy and dangerous horseless carriages.

Autonomic computing does not mean the end of understanding and editing configuration files. It will mean that this task will consume less of the sysadmin's working day. I expect autonomics, in some form, will evolve, regardless of kicking, screaming, and temper tantrums or editorializing against its adoption. And the people who develop autonomics may not be sysadmins but researchers willing to take a top-down, instead of a bottom-up, approach.

So times change: either the world becomes more complicated, or it appears more complicated because it now works differently. Remember, there are still many people living in developed countries who do not use electronic communication such as email, IM, and text messaging. Don't get left behind.

The Lineup

I had often wondered about anycasting, so I contacted ISC and found Joe Abley willing and able to describe the pros, cons, and sheer aggravation surrounding the use of anycasting in IPv4. Anycasting is not a solution that many can use, but I believe you should be aware of it.

Alva Couch follows with his article that examines mindsets, or the semantic wall I also attempted to describe in this editorial. Learning more about the

differences between imperative and functional programming languages is not the point of Alva's article; the example is just used to demonstrate what he considers the crux of building real autonomic computing systems.

Eric Langheinrich next tells us about a method for controlling access to Web content that relates to anti-spam techniques. Through the use of scripts and a scoring service, you can configure your Web server to deny content to crawlers looking for certain content, such as email addresses, to be used in later UCE.

We next have two articles related to papers presented at LISA '07. A group from the University of Arizona describes Stork, a package management system designed for use in clusters and PlanetLab. Once you have installed your distributed applications, you can consider managing those applications using Plush, the second in this set of articles.

Octave Orgeron then continues his tutorial on Solaris LDoms, with a focus on advanced topics in network and disk configuration. He is followed by Aditya Sood, who explains problems with XML signing.

We have a new columnist starting with this issue of *;login:*. Peter Galvin, longtime tutorial instructor at USENIX conferences as well as the Solaris columnist for the now-defunct *Sys Admin*, has agreed to write about Solaris for *;login:*. I am happy to help provide a new home for Pete's column and hope that many of you will continue to enjoy reading it.

And, as mentioned, we have summaries of LISA '07, as well as of four of the workshops that occurred before the main conference began.

Starting with this issue, *;login:* will include a cartoon courtesy of User-Friendly. We are thankful to David Barton for allowing us to lighten up our pages with some relevant humor.

Times are changing. But then times always change, and those changes often prove upsetting and difficult even to consider, much less accommodate. What sysadmins face today is an exploding number of computers, and computer-enabled devices, that must be managed. We need to look toward new technologies that will make managing these devices easier, even if the transition will be difficult. And I can't imagine it will be easy.

REFERENCES

[1] LISA '07 Technical Sessions: http://www.usenix.org/events/lisa07/tech/.

[2] Home page for IBM's Autonomics project: http://www.research.ibm.com/autonomic/index.html.

[3] Wikipedia page, with more links about autonomics at the bottom: http://en.wikipedia.org/wiki/Autonomic_Computing.