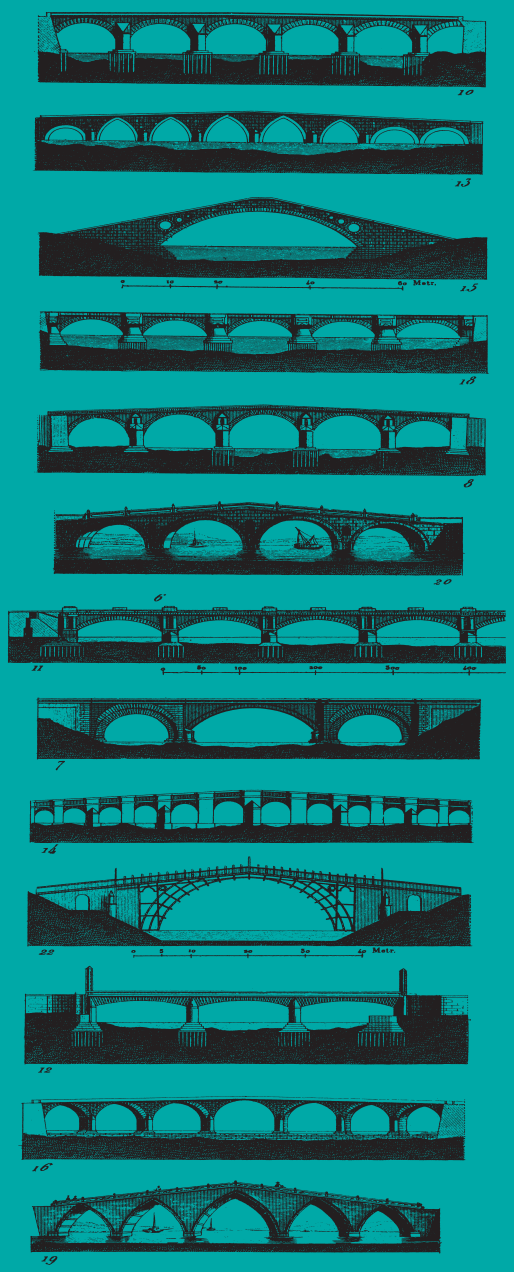
FEBRUARY 2010 VOLUME 35 NUMBER 1

;login:

THE USENIX MAGAZINE



OPINION

Musings — 2
RIK FARROW

Whither LISA? — 7
SEAN KAMATH

SYSADMIN

Configuration Management Phenomenology — 10
ALVA COUCH

Puppet and Cfengine Compared: Time and Resource Consumption — 16
JARLE BJØRGEENGEN

FILE SYSTEMS

Building and Installing a Hadoop/Mapreduce Cluster from Commodity Components: A Case Study — 26
JOCHEN L. LEIDNER AND GARY BEROSIK

The Case for Specialized File Systems, or, Fighting File System Obesity — 38
EREZ ZADOK, VASILY TARASOV, AND PRIYA SEHGAL

SECURITY

A Look at VoIP Vulnerabilities — 41
ANGELOS D. KEROMYTIS

NETWORKING

Taming the Torrent — 51
DAVID CHOFFNES AND FABIÁN E. BUSTAMANTE

SSR: Securing the Internet's Identifier Systems — 60
DAVID PISCITELLO

HARDWARE

Peculiarities of Radio Devices — 64
RUDI VAN DRUNEN

COLUMNS

Practical Perl Tools: You Never Forget Your First Date (Object) — 73
DAVID N. BLANK-EDELMAN

Pete's All Things Sun: Crossbow — 79
PETER BAER GALVIN

iVoyeur: Breaking Up Is Hard to Do — 86
DAVE JOSEPHSEN

/dev/random: Project Management—A Primer — 90
ROBERT G. FERRELL

BOOK REVIEWS

Book Reviews — 93
ELIZABETH ZWICKY ET AL.

USENIX NOTES

Nominating Committee Report, 2010 Election for the USENIX Board of Directors — 97
RÉMY EVARD

Thanks to Our Volunteers — 98
ELLIE YOUNG

CONFERENCES

Report on LISA '09: 23rd Large Installation System Administration Conference — 100

USENIX
The Advanced Computing Systems Association

USENIX Upcoming Events

3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '10)

Co-located with NSDI '10

APRIL 27, 2010, SAN JOSE, CA, USA
http://www.usenix.org/leet10
Submissions due: February 25, 2010

2010 Internet Network Management Workshop/Workshop on Research on Enterprise Networking (INM/WREN '10)

Co-located with NSDI '10

APRIL 27, 2010, SAN JOSE, CA, USA
http://www.usenix.org/inmwren10

9th International Workshop on Peer-to-Peer Systems (IPTPS '10)

Co-located with NSDI '10

APRIL 27, 2010, SAN JOSE, CA, USA
http://www.usenix.org/iptps10

7th USENIX Symposium on Networked Systems Design and Implementation (NSDI '10)

Sponsored by USENIX in cooperation with ACM SIGCOMM and ACM SIGOPS

APRIL 28–30, 2010, SAN JOSE, CA, USA
http://www.usenix.org/nsdi10

2nd USENIX Workshop on Hot Topics in Parallelism (HotPar '10)

Sponsored by USENIX in cooperation with ACM SIGMETRICS, ACM SIGSOFT, ACM SIGOPS, and ACM SIGARCH, and ACM SIGPLAN

JUNE 14–15, 2010, BERKELEY, CA, USA
http://www.usenix.org/hotpar10

8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys 2010)

Jointly sponsored by ACM SIGMOBILE and USENIX

JUNE 14–18, 2010, SAN FRANCISCO, CA, USA
http://www.sigmobile.org/mobisys/2010/

USENIX Federated Conferences Week JUNE 21–25, 2010, BOSTON, MA, USA

2010 USENIX Annual Technical Conference (USENIX ATC '10)

http://www.usenix.org/atc10

USENIX Conference on Web Application Development (WebApps '10)

http://www.usenix.org/webapps10

3rd Workshop on Online Social Networks (WOSN 2010)

http://www.usenix.org/wosn10
Paper submissions due: February 18, 2010

2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '10)

http://www.usenix.org/hotcloud10
Submissions due: March 23, 2010

2nd Workshop on Hot Topics in Storage and File Systems (HotStorage '10)

http://www.usenix.org/hotstorage10

19th USENIX Security Symposium (USENIX Security '10)

AUGUST 11–13, 2010, WASHINGTON, DC, USA
http://www.usenix.org/sec10

9th USENIX Symposium on Operating Systems Design and Implementation (OSDI '10)

Sponsored by USENIX in cooperation with ACM SIGOPS

OCTOBER 4–6, 2010, VANCOUVER, BC, CANADA
Submissions due: May 7, 2010

24th Large Installation System Administration Conference (LISA '10)

Sponsored by USENIX in cooperation with LOPSA and SNIA

NOVEMBER 7–12, 2010, SAN JOSE, CA, USA
http://www.usenix.org/lisa10
Submissions due: May 17, 2010

For a complete list of all USENIX & USENIX co-sponsored events, see http://www.usenix.org/events.

contents

;login:
VOL. 35, #1, FEBRUARY 2010

USENIX ASSOCIATION
2560 Ninth Street,
Suite 215, Berkeley,
California 94710
Phone: (510) 528-8649
FAX: (510) 548-5738

http://www.usenix.org
http://www.sage.org

;login: is the official magazine of the USENIX Association.

;login: (ISSN 1044-6397) is published bi-monthly by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

$90 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $125 per year.

Periodicals postage paid at Berkeley, CA, and additional offices.

POSTMASTER: Send address changes to *;login:*, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

OPINION

Musings — 2
RIK FARROW

Whither LISA? — 7
SEAN KAMATH

SYSADMIN

Configuration Management Phenomenology — 10
ALVA COUCH

Puppet and Cfengine Compared: Time and Resource Consumption — 16
JARLE BJØRGEENGEN

FILE SYSTEMS

Building and Installing a Hadoop/Mapreduce Cluster from Commodity Components: A Case Study — 26
JOCHEN L. LEIDNER AND GARY BEROSIK

The Case for Specialized File Systems, or, Fighting File System Obesity — 38
EREZ ZADOK, VASILY TARASOV, AND PRIYA SEHGAL

SECURITY

A Look at VoIP Vulnerabilities — 41
ANGELOS D. KEROMYTIS

NETWORKING

Taming the Torrent — 51
DAVID CHOFFNES AND FABIÁN E. BUSTAMANTE

SSR: Securing the Internet's Identifier Systems — 60
DAVID PISCITELLO

HARDWARE

Peculiarities of Radio Devices — 64
RUDI VAN DRUNEN

COLUMNS

Practical Perl Tools: You Never Forget Your First Date (Object) — 73
DAVID N. BLANK-EDELMAN

Pete's All Things Sun: Crossbow — 79
PETER BAER GALVIN

iVoyeur: Breaking Up Is Hard to Do — 86
DAVE JOSEPHSEN

/dev/random: Project Management—A Primer — 90
ROBERT G. FERRELL

BOOK REVIEWS

Book Reviews — 93
ELIZABETH ZWICKY ET AL.

USENIX NOTES

Nominating Committee Report, 2010 Election for the USENIX Board of Directors — 97
RÉMY EVARD

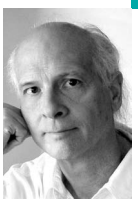
Thanks to Our Volunteers — 98
ELLIE YOUNG

CONFERENCES

Report on LISA '09: 23rd Large Installation System Administration Conference — 100

RIK FARROW

musings



Rik is the Editor of *;login:*.

rik@usenix.org

I THOUGHT I'D TAKE A DIVE INTO THE deep end for my musings this time around. Maybe Sean Kamath's "Whither LISA?" (p. 7) got me going on this, or Alva Couch's own deep dive into phenomenology in system configuration (p. 10). Or perhaps it was the re-reading of *Accelerando* [1]. Whatever it was, I've decided to imagine what life, and work, would be like for a senior sysadmin 10 years from now.

Chuck awakens from a recurrent nightmare. He is sitting in a virtual meeting room, having been "promoted" to management, when he gets asked to present his research on fulfilling subsection 11.2.c of ISO 700001:2019 on the labeling of subsapient AIs. Of course, he's been caught, metaphorically, with his pants down, as he has nothing to present on this mind-numbingly boring topic, in yet another meeting.

Chuck is visibly agitated, but manages to wake up enough to head to the refresher. A few minutes later he returns and settles back in his bed, which has made itself as well as reshaped itself into a perfectly form-fitting recliner.

Chuck gestures for a glass of vanilla-flavored LJAA (Low Jitter Adrenaline Analog) and a glass of it pops up from his lounge-side table as he prepares to view reports from the system monitoring network.

HAZE Computing

Chuck is the beneficiary of the continuous advances predicted by Moore's law [2], in that his home office has gone well beyond the Cloud Computing [3] that was all the rage back in the early 'teens. He has the latest in HAZE (Highly Adaptable Zoomorphic Engines) systems growing in his walls, with computing power that was totally unthinkable just a few years before. His every gesture gets interpreted as a command, which has made things like nose-wiping into a very conscious and deliberate activity.

The overnight monitor reports appear in Chuck's view as if they were floating in space ten feet away. The seventeen different windows all include color-coded borders indicating the general health and functioning of each subsystem. Today, several windows are bordered with an ominous-appearing

orange, and Chuck grimaces, then takes a big gulp of LJAA. Things are going to be interesting today indeed.

By focusing on a window, he can make it swish from its place in the array and zoom until it covers the other windows, which still can be seen vaguely behind it. This window deals with user satisfaction. A blink, and the user satisfaction window drills down into graphs showing the groups of users his team is responsible for. The scientist groups are blinking red, and Chuck sighs.

The scientists have been unhappy for years, ever since their treasured grids and clusters have been moved to the Physical Plant. It wasn't his fault that those systems generated enough waste heat to heat (and cool) the entire complex. As heating and cooling falls under the control of the Physical Plant, Plant gets to run grids and clusters as well, recycling their heat [4]. The scientists still get to use these systems, but the Physical Plant manager often throttles performance to match times of peak heat requirements, something that (of course!) vexes the scientists.

Chuck chuckles and gestures a note to himself to add a weight to the display of the scientist group's user satisfaction index. There is really nothing he can do about their discomfort, unless he can help them get their own HAZE-based systems. But that's a job for management.

Backup

Chuck blinks, and the user satisfaction window zips back into its position. Another steely glare and the next orange-tinged window pops into front-and-center position. This window warns Chuck that today is backup day, and the automated truck carrying the disk arrays is late. Chuck gestures up a tracking screen and can see that the truck is sitting in a recharging station. Another gesture shows him that demand for electricity is exceptionally high this morning (yet another day of unusual heat!), and that is slowing down the recharge rate for the truck.

Chuck fondly recalls the days when he did off-site backups to tapes, back in the 'oughts. These days, with petabyte storage systems being the norm, nothing less than a truck full of disks (TFOD) can be used for reliable backup storage.

Like other businesses deemed part of the crucial infrastructure by the government, Chuck is responsible for seeing that all data gets backed up routinely, shipped by automated truck, then stored in Cheyenne Mountain. Chuck is vaguely aware that Cheyenne Mountain was once the doomsday bastion for fighting nuclear wars, but those days have passed. To Chuck, Cheyenne Mountain is just a secure place for off-site backups.

Chuck gestures up a trouble ticket, tags it with the truck's position and an action item to notify him when the truck has arrived and its TFOD is online. The backup operation will have to be postponed until this evening, as it not only takes all night but also sucks up almost all available I/O bandwidth.

Virus Alert

Chuck blinks the window down, then stares up the final orange-framed window. He grimaces with annoyance as he sees that the network has a viral infection again. Ever since researchers learned how to use bacteria to build circuits, viruses have become a terrible problem. This virus is reducing net-

work bandwidth for several marketing managers' offices. Chuck calls up an anti-viral robot and sends it searching through the office's plumbing.

Chuck's own wideband connection is via his sewage pipe. It turned out to be the perfect place for genetically engineered bacteria to grow, and the sewage treatment plants provide a centralized location for connection to Internet3. He also can fall back on the much slower (1Gbs) link provided by his smartphone, if needed.

Chuck wonders why the marketing managers' network constantly has such issues. He pops a new window open to HeadSpace and grimaces as his brain gets scanned, authenticating him. He hates the ticklish feeling as the scanner passes over his cortex and doesn't believe the scan is any better than a plain old fingerprint. Then he is in.

HeadSpace

HeadSpace, the social networking site favored by uber-geeks, appears as an endless plane stretching to the vanishing point in all directions. He gestures a query, then zooms across the plane to a gathering of like-minded visitors, who appear to be arguing. At times like this, Chuck marvels at the completely immersive experience provided by his gaming implants. The brain surgery was scary, even if it was done by robo-docs, and the cost—well, at least he could write off part of it as a business expense.

A stocky dwarf avatar is shouting, "You'd be better off if you stuck with McMantic anti-viral!"

An androgynous-looking mage intones a response: "I only trust Merc-Welcome Pharmaceuticals when it comes to protecting my network's environment."

Chuck butts in, and his well-muscled warrior avatar works its own type of magic, as two meters of hulking, edgy, muscle-equipped-with-sharp-edged-instruments has a habit of doing. "Hey, has anyone heard of virals that seem to be attracted to marketing networks in particular?"

A pretty little fairy pipes up: "The creeps probably deserve it."

After a few more comments about how the marketing droids get all the nice bennies, a fairly normal-looking avatar (just an impossibly handsome one) suggests that he check to see if anyone in that group has visited Thailand recently. Turns out there is a rather nasty form of STD there that might also have an effect on the bacteria the network runs on.

Chuck's avatar grunts out a thanks and vanishes. Chuck doesn't log out yet, as it is time for the morning gripe meeting with his team. He pops into existence at the opening of a cave overlooking a jungle environment. Most of his team are already present.

Chloe, appearing as a unicorn, is talking about the user satisfaction index being down. Chuck counters by pointing out that the problem comes from the scientists, as usual. Chloe timidly mentions that she down-weighted the scientists' portion of the index, and it is still in the orange. Chuck opens a window display and zooms in. Chloe is right. Even with the adjustment, many users are still not satisfied.

With some more digging, the team discovers that the naming server had a partial core failure last night, and fewer than half the processor cores are still resolving names. With only 128 cores to handle the DNSSecv2 response verification signatures, name resolving was taking twice as long as usual, up to 30 milliseconds, a noticeable difference. A warm reset of the offline

processor cores provided a temporary solution to the issue. Chuck gestures out another trouble ticket, so that someone can get to the underlying issue before the problem becomes permanent.

After the meeting concludes, Chuck decides he needs a little exercise, and joins a group of warriors on a quest. As he swings his heavy sword, electrodes built into his real-life bed/lounge stimulate his muscles. When Chuck emerges from his virtual visit, his body is gleaming in sweat. Not bad, he thinks, for fifteen vigorous minutes of mock-fighting. He feels great.

Now, back to work . . .

Lineup

Has storage really been getting large enough to require TFODs? You should read the LISA summaries in this issue or watch the video of the presentation about building petabyte-sized disk storage systems [5]. You can also check out the slides or the video of another LISA IT about water-cooled datacenters that can resell their "waste" heat for home heating (Bruno Michel, Friday). We also have summaries from the Advanced Topics, University Issues, and Government and Military System Administration workshops.

After several attempts, I finally managed to include an article about building a Hadoop cluster. Jochen Leidner and Gary Berosik have built several clusters and provide details on building their own cluster using commodity PCs. They also cover some of the reasons behind the interest in clusters.

Besides the articles by Couch and Kamath I've mentioned, we have another sysadmin-related article. Jarle Bjørgeengen was working on a project at his university where they needed to replace an older configuration management system. Björgeengen also wanted to create an experiment where he could scientifically examine performance differences between Puppet and Cfengine3. His results should surprise no one who understands how these tools work, but they are sure to stir things up anyway.

I was able to attend SOSP in Montana this fall, where I got to hear a great panel on "Rethinking File Systems." Only one panelist managed to finish an article for this issue, Erez Zadok (with co-authors Vasily Tarasov and Priya Sehgal), but I hope to have more visions of the future of file systems in future issues.

Angelos Keromytis presents research he has done into known VoIP and IMS vulnerabilities. As Keromytis points out, the RFCs for VoIP are enormous and flexible, and there are many correct but dangerous implementations. He closes his article with suggestions about what you can do to harden your VoIP systems.

Choffnes and Bustamante point out a clever way of improving BitTorrent performance. They show that it is better for participants who share the same provider to also be assigned to the same torrent, as bandwidth within a provider is usually much higher than inter-provider bandwidth.

Dave Piscitello has written an article about the new duties of ICANN. Piscitello works with ICANN, as well as having a long history of Internet-related activities, and provides accurate information about long-awaited changes.

Rudi van Drunen has written about wireless technology. Rudi provides details about how wireless works, as well as a case study in designing a medium-haul wireless network.

David Blank-Edelman takes the time to cover time in Perl, as well as useful modules that don't ship with the core. Dave Josephsen fills us in on turmoil

in Nagios, as a fork appears (or is it just marketing and maneuvering?). Robert Ferrell explains how project management works, or doesn't. And Elizabeth Zwicky opens our book reviews section with several excellent reviews.

Fifty years ago, Dick Tracy's wristwatch radio was science fiction. Now we see people walking around apparently talking to themselves all the time. Intel has followed up on their TeraGrid 80-core demonstration chip with a more exciting 48-core Simple Cloud Computing chip [3]. Perhaps my future vision of sysadmin is not that far off the mark.

REFERENCES

[1] Charles Stross, *Accelerando* (Ace, 2005), in hardcover, paperback, and free ebook: http://www.antipope.org/charlie/accelerando/.

[2] Emin Gün Sirer and Rik Farrow, "Some Lesser-Known Laws of Computer Science": http://www.usenix.org/publications/login/2007-08/pdfs/sirer.pdf.

[3] Ryan Shrout, "Intel Shows 48-core x86 Processor as Single-chip Cloud Computer," *PC Perspective*: http://www.pcper.com/article.php?aid=825.

[4] Bruno Michel, "Towards Zero-Emission Datacenters through Direct Reuse of Waste Heat," IBM Zurich Research Laboratory: http://www.usenix.org/events/lisa09/tech/slides/michel.pdf.

[5] Raymond L. Paden, "How to Build a PB Sized Disk Storage System," IBM Deep Computing: http://www.usenix.org/events/lisa09/tech/slides/paden.pdf.

SEAN KAMATH

Whither LISA?

Sean Kamath is a System Architect for PDI/Dreamworks. He has been a lurker at LISA conferences since 1992.

kamath@geekoids.com

USENIX is looking for input from the membership on how to improve LISA. Join the discussion at http://lists.usenix.org/mailman/listinfo/usenix-discuss. Find out more about how you can participate in LISA '10 on p. 118.

I'VE BEEN A LONG TIME LISA ATTENDEE, having been at every LISA since 1992 (LISA VI). I have a fond memory of my introduction to what it means to be in a room full of sysadmins when Michael Cooper presented "Overhauling Rdist for the '90s." Memorable quote: "I'm not the guy who wrote rdist, I'm just the guy who fixed it." Things have changed.

LISA has always been an intimidating conference for the new attendee, what with people who write books walking among those who buy them. In 18 years of attending the conference, I've seen it grow, shrink, re-grow, and shrink yet again. The focus has shifted from extremely technical, to human oriented, and back. But what I've really been observing for all these years has been the growth of an industry. That's something that has both upsides and downsides.

I've been mulling over patterns of development as they apply to everything from businesses to professions and even governments. As our occupation has grown from "those weird people who do things with those weird computers," we've experienced that which other occupations have experienced: legitimacy, growth, formalization, specialization, and something bordering on stagnation. Are we at stagnation yet? I hope not.

At that first conference I attended, my co-worker, Dan, and I were in one of the sessions where someone said, "It's OK to make system administration your profession." This was like an epiphany for Dan. Dan had been a technical support specialist and wanted to get more technical (TSSes were not actually all that technical, it turns out). When he made the transition to system administration, Dan had a hard time explaining to a lot of people what he did and why he chose to do it at that point in his life (in his 30s).

Remember, at that time, sysadmins were often actually classified as operators, and the position was usually thought of as a stepping-stone into true development work. All during the '90s, I met a lot of software developers who would tell me, "I used to do what you did." Implicit in this was that they'd moved onward and upward, to something respectable and better-paying. But during this time, the profession became accepted. People posted jobs for it. It was no longer a student position, or an

entry-level position for people who weren't quite up to snuff in coding or lacking in software development classes.

The flipside, the real upside, of this time was that people who were in the position generally really cared about what they did. To be in a job no one understood took a lot of self-confidence. And the people who forged that new path either knew what they were doing or moved into other fields.

The Roaring '90s

It was a heady time of system administration. The conference was a wonderland, where you could steep yourself in the environment of people who did what you did, inculcate yourself in a culture you might not have known even existed. Tools were written by people who had the same problem you had, not by companies hoping to make a buck. Support was checking the code or emailing the original developer. You gave your changes back to help others.

By the end of the '90s, we had a profession. We had certifications. We had a huge conference—even LISA-NT! We had legitimacy. Along with this was the introduction of products. No longer was the Vendor Exhibit six vendors who were mostly conference attendees who happened to work for the vendors with tables and some printouts and a few geeks standing around. We sysadmins were in demand. We'd known all along that we were needed, but now the companies knew it, too. The Internet ran on sysadmins! Having great sysadmins could make or break a company—or so everyone thought.

This led to hyper-inflation in the sysadmin field. Those of us who lived through it (and went to the conferences) will never forget the insanity. Dan and I walked into a hospitality suite one night in Chicago. We were greeted with (and I'm not making this up), "Hi! Come on in! You want to work for us!" My friend said, "How can you know you want to hire me?" which garnered the reply, "Well, you're at the conference, right? We want to hire you!" We drank some—OK, a lot of—vodka and kept our existing jobs. But we all remember that huge influx of newbie sysadmins who were like doe-eyed children in a candy store.

Of course, a huge number of inexperienced sysadmins were ripe for vendors to expand their professional services offering ("Doing the work so you don't have to learn how!"), as well as other companies either productizing open source applications or creating their own closed source versions. And with companies that had so many inexperienced sysadmins, who could blame them for purchasing things that did what sysadmins used to know, or would have learned, how to do?

The Bubble

Everyone remembers the dot-com collapse. I like to believe no small part of it was the idea that what the sysadmins had done had led people who didn't understand what we do to believe anything was possible. When you sweep the hubris aside, I think there's some truth to the idea that those tools we sysadmins wrote to help us get things done morphed into a lot of online applications, in one form or another, that made the basis of a lot of the dot-com bubble.

I know I paint a jaded picture of this time. I also know there was a fair amount of innovation and advancement in the field of computers. The complexity of environments was also exploding. This led to a legitimate need to add tools and hardware that were extremely complex and cost a lot

of money. The seeds for my discontent were sown then, and it doesn't really matter if it was through folly or genuine need.

From the ashes of the dot-com conflagration, the IT phoenix rose again. My recollection was that LISA finally did generate enough attendance that it exceeded the peak of the dot-com bubble. It took a while and was a rough time. The focus on the profession was about putting your head down and getting things done. (I think LISA '03's line summed it up best: "Effective Real World System Administration: Virtuosity, Flexibility, Ingenuity, Perseverance"—i.e., stay the course.) Attending LISA during that time brought the realization, for me at least, that our profession had survived the first two phases of development (initial acceptance, or legitimacy, and exponential growth) and had now started down the path of formalization.

While there had been efforts to create certification processes early on, those efforts were joined by companies offering training and certification on the administration of their own products. And that paved the way for the next stage: specialization. One reason it was so difficult to get a certification for "system administration" was that it was such a broad spectrum of work. Two sysadmins from two different companies might do drastically different tasks. As companies offered administration certification on their products, naturally those products had special functions. And the OS vendors also realized that as companies embraced what we did, they could have certifications not in (or at least in addition to) "system administration," but in networking, storage, and security. Much like the medical profession saw specialists as the technology matured (or the knowledge increased past the ability for one person to understand it all in their own head), so, too, did system administration. Not only was it a horizontal spread, but also vertical: junior storage admin, senior networking admin. You get the idea.

Today

So that leaves me at the current day. This is where we are. When I showed up at LISA '09 (LISA XXIII for us old-timers), I was somewhat despondent. The desire to express my frustration at the situation led to writing this column. And an interesting thing happened during the conference: I saw hope for the future. In an effort to see how other people felt about the conference, I learned that my views were not universal. Many, many folks had the same wonderment and enjoyment of the later conferences as I had of the earlier ones. I truly had become jaded. However, once again, I learned I was not alone. There was a sizable community of people who were looking for the next level, as it were. A way to get back to some of what we've lost. I've joined some people who are considering a possible new track at LISA for the future. Nothing is a done deal, but it brings me hope. And once again at LISA, I've found a community of like-minded people who are motivated to better our profession, to help others and make things work. I guess it's just what system administrators are hardwired to do.

So, yeah, LISA has changed over the years. My hope (and part of my reason for writing this column) is to call other system administrators to action, to egg them on to join me in the process of providing that forum we all would find useful. To recognize that it's our responsibility to better our profession by getting out there, writing tutorials, papers, presentations and talks, or whatever. We each need to do our part to bring the depth and breadth of our knowledge into the world. It's our choice whether we let vendors and others outside our profession drive us or whether we take the wheel ourselves and head off in the right direction.

ALVA COUCH

configuration management phenomenology

Alva Couch is an associate professor of computer science at Tufts University, where he and his students study the theory and practice of network and system administration. He served as program chair of LISA '02 and was a recipient of the 2003 SAGE Professional Service Award for contributions to the theory of system administration. He currently serves as Secretary of the USENIX Board of Directors.

couch@cs.tufts.edu

A QUANDARY IN MAPPING BEHAVIOR TO configuration is traced, in part, to a philosophical quandary rooted in the relationship between system administrator and user.

> "Talk to the bomb. Teach the bomb phenomenology."
>
> —The captain, in *Dark Star: The Spaced-Out Odyssey*

At the climax of the cult science fiction parody *Dark Star* [1], a "smart bomb" has decided to explode while still in the spaceship bay, rather than exploding on the planet that it is supposed to destroy. The spaceship crew attempt to "solve" this problem by engaging the bomb in a deep philosophical discussion of the meaning of life and exploding; they try to convince the bomb that it need not explode, because the importance of whether it explodes or not is subjective and not particularly significant in the larger picture of things.

This silly discussion somehow reminds me of the state of the art in configuration management. Tools act on the configuration as if it were the definitive representation of behavior and assume that this is enough for tools to do. Meanwhile, the behavior of a configuration management solution is monitored via mechanisms that—again—quietly assume that configuration *defines* behavior. But it might not, for many reasons. In autonomic computing parlance, the human system administrator is left to "close the loop" between configuration and behavior, and, when things go wrong, must rely upon intuition and experience to "close" this loop manually.

The situation, similar to the situation in *Dark Star*, is that the tools allow the environment to "explode," humans must intervene, and the apparent solution, as in *Dark Star*, is to "teach the tools phenomenology" by giving tools perceptive capabilities by which they can understand the effects of their actions. What does this mean, and is it even reasonable? In this article, we explore this question from several angles.

Beyond Semantics

This article might be considered the second in a series. In the first article [2], we discussed the semantic wall between high-level and low-level configuration specifications and how difficult it is to map between high and low levels of abstraction in a configuration. We commented on the difference between "specifying configuration" and "specifying behavior" as a problem of semantics.

Now, two years later, another problem looms on the horizon. Even if we manage to successfully bridge the gap between levels of configuration abstraction, the problem of bridging desired and observed behavior remains. This is not just a problem of semantics, but a deeper problem with the assumptions we make and the way we approach both configuration management and the profession. While the former problem arises from difficulties of meaning, this problem arises from the philosophy that we adopt in satisfying user needs.

Phenomenology

In a naive sense, "phenomenology" refers to the practice of relying upon one's senses to define the nature of the physical world. In like manner, I refer to phenomenology in system administration as the practice of trusting what one can observe the system actually doing, instead of trusting any abstract idea one might have of what it is supposed to do. Thus I propose that "a machine's identity is what it does," by contrast with the traditional configuration management view that "a machine's identity is how it is configured." For each configured machine (to paraphrase Sartre), I assert that "to do is to be," i.e., machines' behaviors define their natures. By contrast, a fundamental tenet of configuration management (attributed to Socrates) is that "to be is to do," i.e., machines' natures define their behaviors.

Although it might seem that I am splitting meaningless philosophical hairs, there is a world of difference between these definitions that strikes at the core of the assumptions underlying configuration management as a practice. We often comfortably and tacitly assume that the way a machine is configured defines its behavior. I beg to differ for a multitude of reasons. The reason that this assumption is false is more than a simple problem of semantics. Behaviors arise from sources other than the configuration.

Using Phenomenology

Phenomenology is not a new idea for system administrators; we use it every day. In tuning a configuration or troubleshooting a problem, we engage in controlled (or perhaps not-so-controlled) experimentation. We are intimately familiar with many cases in which what something does correlates poorly with what we think it is and—implicitly—we quietly modify our idea of what it is, accordingly.

My evidence is, however, that we do not go far enough in believing our senses. Our behavior is based upon hidden assumptions—deeply embedded in practice—that influence and sometimes cloud our thinking. One way to bring those assumptions out into the open is to consider how we philosophically approach the problem of system administration.

Verification and Validation

There is a subtle difference between what current configuration management tools do and what we tacitly assume that they do, which is similar to the difference between "verifying" and "validating" a software product in software engineering. According to software engineer Barry Boehm, the process of "verification" answers the question, "Are we building the product right?" This is the way most configuration management tools work. A configuration is "verified" if it accords with the system documentation. "Validation," by contrast, answers the question, "Are we building the right product?" A sys-

tem is validated if it is doing what users need it to do (and—implicitly—not doing things they do *not* want it to do) [3].

In human terms, verification involves making sure that we have obeyed the documentation for a product in trying to manage it, while validation involves ensuring that the documentation is itself correct and definitive about the relationships between configuration and behavior. Current practice engages in the former and assumes that verification implies validation (so that explicit validation is optional rather than required). And this is almost always a bad assumption to make.

Consider, for example, that a non-functional email server can be broken in two basic ways. First, the configuration can remain unverified, e.g., the configuration tool fails to modify it properly. It is more common, however, for the configuration to be verified but not validated, e.g., the configuration looks as it should, but the system still fails to forward email. The latter is a validation problem.

In going around the table at LISA, I found that almost everyone has some story of getting burned as a result of incorrectly assuming that the documentation is correct. The simplest example is that of a manual page that describes the wrong syntax for a file, but there are much more subtle variants. As software is revised, the manual pages need not keep up with it, so that one is often reading older descriptions of newer software.

Closed-World Assumptions

The assumption that verification implies validation is just one example of a "closed-world assumption" that arises in system administration practice. Verification is necessary but not sufficient for validation. Verification is only sufficient when "what you ask a system to do" is always "what it does." This is an implicit closed-world assumption that all influences upon the managed system are known and accounted for. In other words, the kind of thinking that this represents might be paraphrased as "to be is to do."

There are many cases in which this implicit assumption fails to hold: when the managed software has a bug that affects behavior, for example, or when there are hidden unmanaged influences, such as a forgotten configuration file that can adversely affect behavior. In the worst case, a security breach can change all the rules and even replace the managed application with another unknown and hostile one.

Closed-world assumptions pervade our practice. We often implicitly assume that configuration completely determines behavior, and that a specific configuration tool completely controls configuration and thus behavior. I say "implicitly" because there is no conscious action on our part to assume anything, but the assumption quietly lurks in how we use our data!

Consider, for example, how we currently document a site's function. Usually, some description of the configuration suffices: either a description of how each machine is configured or some network-wide, tool-readable description. This seems innocent enough, until we consider that it is often the *only* documentation of site function. At a deeper philosophical level, a configuration description cannot be more than a statement of *intent* rather than fact. Anything we do outside its closed-world assumption is (implicitly) not documented.

Open-World Assumptions

Phenomenology, by contrast, implicitly adopts an open-world assumption that more or less any behavior can arise as a result of configuring a system. A system is what it does. The configuration might result in appropriate behavior, but it might not. Verification does not imply validation. In other words, we might think of an open-world assumption as equivalent to the philosophical stance "to do is to be."

One's philosophical stance can have a profound impact upon one's everyday practice. If one really considers validation as separate from verification, then there is no way to "prove" correct system function. As in software testing, one can never fully test a system, and the only solid evidence one can gather is that something is *not* working. But this philosophical stance also clarifies some of our thinking about behavior. By throwing away a tacit and common assumption that has been proven false countless times, we are freed to reason more clearly about configuration, behavior, and contingencies.

I consider it almost a tautology that the job of a system administrator is to "close an open world," i.e., to provide some concept of predictability in an otherwise unpredictable environment [4]. One starts with an "open world" (e.g., the Internet) and makes some adjustments to make that world "usable." Along the way, one forms "closures," islands of predictability in an otherwise unpredictable universe, where what you think you are telling something to do is what it actually does, i.e., verification implies validation [5]!

The Value of Philosophy

So far, this discussion probably seems abstract and impractical. What, you might ask, is the value of a philosophical stance? Isn't system administration what we do, and not how we think about it? I claim that simply refusing to "believe" that verification implies validation has profound implications for practice.

Particularly, if we refuse to blindly believe that verification implies validation, there is always a validation step after configuration management. That step involves observing behavior, and effective testing (manual or automatic) becomes a central part of system administration and our tools. Tools learn to observe the world as well as to configure it.

But less tangible benefits include the ability to ask new questions that our prior beliefs had sidelined. We must eventually ask, "What is validation?" and, more importantly, "What behavior is actually desired?"

Monitoring Is Not Validation

One might think that log monitoring is a form of validation; after all, monitoring does measure behavior rather than configuration. But monitoring records *symptoms* of behavior, not the behavior itself, and it is possible for a system with the proper symptoms to be behaving improperly.

Consider the common problem of a log message saying that an undelivered email was delivered. This can happen in many ways: for example, the file system on which the message is to be stored can fail *after* delivery. Symptoms can only be definitively related to causes if there is again an implicit "closed-world assumption" that the monitoring data is complete enough to represent what actually happened. In the above case, that assumption is equivalent to the assumption that "the disk does not fail," which is clearly ridiculous. Expected log entries are again necessary but not sufficient for

proper operation; there are many cases in which the log is correct but behavior is wrong.

Monitoring *is* validation if an appropriate closed-world assumption holds. Thus, monitoring is sufficient if we have already verified (by some other mechanism) that a closed-world assumption is reasonable. But monitoring, by itself, cannot substantiate a closed-world assumption.

Real validation involves more explicit testing than most of us do. Are email messages really being delivered? Are services responding properly? This includes checking on the actual function of services, and not solely relying upon logs of past behavior.

What Is Behavior?

To achieve validation we must first understand what behaviors are desirable. Describing behavior might seem a daunting task, but we are aided by two simple ideas. First, user-level behavior is much easier to describe than the configuration that assures it. Behavior is a much higher-level thing to describe than configuration. A behavioral description can be written to be relatively portable and reusable for many sites, while configuration contains the (often hopelessly non-portable) methods for assuring that behavior. Configuration—because it is "how" and not "what"—contains details that have nothing to do with behavior. Second, most user needs are met by a set of well-known behaviors. Behavioral expectations are largely homogeneous over the whole Internet and thus more reusable from site to site than configuration details, which by contrast are highly heterogeneous.

Note that a so-called "high-level configuration system" as first proposed by Anderson [6] is *not* a description of behavior but, rather, an abstract (and hopefully more portable) definition of *configuration*. A "high-level" configuration language still describes "what a system should be" instead of "what a system should do." Any linkage between these two is again a closed-world assumption.

Facing Social Forces

Given that the system administrator has to use phenomenology on a daily basis, one might ask why implicit closed-world assumptions are so easy for us to accept. I believe the roots of our closed-world assumptions are social rather than scientific.

One social reason that it is "convenient" to sweep "behavior" under the rug is that we remain unaware, on average, of exactly how our systems behave. Users make changes, and thus behavior changes. There is "behavioral drift" (and even "behavioral rot") based upon independent actions of individuals, especially in a desktop environment. But at a deeper level, the system behaviors that users "need" are different from what they might "want." And facing *that* quandary, and the quandary of whether to give users what they want or what they need, remains "the elephant in the room" whenever we discuss behavior.

Our job is "closing open worlds." The typical user wants to be able to do "everything." And we can't close *that* world.

I think this social reason is the real force underlying our confusion between configuration and behavior, and between verification and validation. It is "convenient" and "comfortable" to assume that configuration determines behavior—and, implicitly, that verification implies validation—because

otherwise we have some very difficult social questions to answer about what behavior "should" be. We can hide behind what tools do and escape the "should," by adopting a convenient closed-world assumption!

Do-Be-Do-Be-Do!

The old joke (which I first learned from scribblings on the MIT Math Department men's room wall) is that the response to Socrates' "To be is to do" and Sartre's "To do is to be" is Sinatra's "Do-Be-Do-Be-Do"! I think that Sinatra better describes current configuration management practice than Socrates or Sartre does. We make closed-world assumptions in enforcing and monitoring behavior, and open-world assumptions in troubleshooting. I believe that for the practice to evolve, we have to stop conveniently fabricating closed worlds where they cannot exist. But to do this, we must acknowledge and directly deal with the social forces that brought about our current philosophy.

Facing the social forces is uncomfortable, and the fuzzy relationship between user and system administrator can become even fuzzier when we try to document it. Users ask for "everything," implicitly or explicitly, and we find it difficult to say no. It is more comfortable sometimes to live in ignorance of user expectations and hope in return that users live in ignorance of our true limitations!

But I also believe that facing this "elephant"—and coming up with ways to precisely specify and guarantee system behavior—is crucial to the ongoing evolution of the profession. Without that step, system administration appears to undertake the theoretically impossible task of closing *every* world the user's heart desires. Making the task clearer to the user involves casting out our own closed-world assumptions in a first step toward encouraging users to cast out theirs.

Only then can we truly be partners with users and replace attempting the impossible with cooperating on the possible. To do this is to be.

REFERENCES

[1] For some reviews of *Dark Star*, see http://www.flixster.com/movie/dark-star.

[2] Alva L. Couch, "From x=1 to (setf x 1): What Does Configuration Management Mean?," *;login:*, vol. 33, no. 1, February 2008.

[3] Barry W. Boehm, *Software Risk Management* (IEEE Computer Society Press, 1989), p. 205.

[4] Alva L. Couch et al., "Seeking Closure in an Open World: A Behavioral Agent Approach to Configuration Management," Proc. LISA 2003.

[5] Mark Burgess and Alva Couch, "Modeling Next-Generation Configuration Management Tools," *Proceedings of LISA '06: 20th Large Installation System Administration Conference* (USENIX Association, 2006).

[6] Paul Anderson, "Toward a High-Level Machine Configuration System," *Proceedings of LISA VII: 7th USENIX System Administration Conference* (USENIX Association, 1994).

JARLE BJØRGEENGEN

Puppet and Cfengine compared: time and resource consumption

Jarle Bjørgeengen has been working with UNIXes, storage, and HA clusters for about a decade. He is now a master's student at Oslo University College, and works for the UNIX group of the central IT department (USIT) of the University of Oslo.

jarle.bjorgeengen@usit.uio.no

DURING THE MASTER'S PROGRAM AT Oslo University College, I was required to define and carry out a set of scientific experiments. As both graduate student and member of a project evaluating configuration management tools for the University of Oslo's IT department [9], I was looking for an experiment that would serve both of these roles. The project was about evaluating Cfengine [3] and Puppet [5] against our existing script-based solution.

My experiment needed to fulfill the criteria of scientific measurability and bringing a new and valuable approach to the decision-making process. I chose to compare time and resource consumption in Puppet and Cfengine 3 tools when carrying out identical configuration checks and actions.

This would be useful information to have for a tool to be used on a large scale, since the time and resources used limit the scope of managed configuration during a certain period. Time usage for the state compliance verification process is of particular interest, since it will affect the frequency and/or scope of the configuration verified.

Equipment and Tool Setup

The experiment was run on a PC with the following specifications:

- CentOS 5.2 , 2.6.18-92.1.22.el i686
- MSI MS-6380E (VIA KT333 based) motherboard
- 1024 MB RAM
- AMD Athlon XP2200 (1.8GHz / 266MHz FSB) 256KB cache
- Quantum fireball 7200 rpm / 30GB / 58169 Cyl / 16 Head / 63 sectors

In both tools it is the configuration agent that does the job of converging to desired state. In Puppet a server component is mandatory, since it is the server that provides the agent with its configuration. The latest stable version of Puppet at the time the experiment was done was version 0.24.7 and that is the version used in the experiment.

Cfengine's configuration agent is independent of a server component, but can be configured to be used with a server component if desired. Cfengine version 3.0.1b3 was downloaded and was compiled according to the installation part of the reference manual [2].

Installing Puppet is easy on CentOS 5 using the EPEL [1] yum repository.

```
# rpm -Uvh http://download.fedora.redhat.com/pub/epel/5/i386/epel-release-5-3.noarch.rpm
# yum install puppet
# yum install puppet-server
# service puppetmaster start
# chkconfig puppetmaster on
```

This also automatically resolves and installs the dependencies needed for Puppet to work (ruby-libs, ruby, facter, ruby-shadow, augeas-libs, ruby-augeas).

Installing Cfengine 3 took a bit more effort:

Dependencies:

```
# yum install db4-devel
# yum install byacc
# yum install openssl-devel
# yum install flex
# yum install gcc
```

Cfengine:

```
# wget  http://www.cfengine.org/downloads/cfengine-3.0.1b3.tar.gz
# tar zxvf cfengine-3.0.1b3.tar.gz
# cd cfengine-3.0.1b3
# ./configure && make && make install && cp /usr/local/sbin/cf-* /var/cfengine/bin/
```

Methodology

The workload chosen for the measurements reflects comparable activities of a configuration management tool. There are three main categories of work:

- File permissions
- File contents
- /etc/hosts entries

The host entries measurement differs a little bit from the file permission and file content workloads. This is because, in Puppet, host entries means a special type of resource, which ends up as file edits to the /etc/hosts file in the end. In Cfengine this is just one form of file-editing operation directly in the configuration language. This difference might make the /etc/hosts entries workload less directly comparable than the file permission and content workloads.

Each type of work was measured in two ways:

1. With a known deviation from the tool's configuration applied up front. Hence the tool will need to converge the deviation to compliance.
2. With the tool's configuration applied up front. Hence the tool will do verification only.

This expands to six measurements for each tool, and each measurement was repeated 40 times. All measurements and application of preconditions for each measurement were done in a shell script which ran the same measurements 40 times.

All workload combinations measured are summarized in the following table:

No.	Type	Converge or Verify	Tool	configfile
1	Permissions	Converge	cf3	cf3-perms.cf
2	Permissions	Converge	puppet	puppet-perms.cf
3	Content	Converge	cf3	cf3-content.cf
4	Content	Converge	puppet	puppet-content.cf
5	Hosts records	Converge	cf3	cf3-hosts.cf
6	Hosts records	Converge	puppet	puppet-hosts.cf
7	Permissions	Verify	cf3	cf3-perms.cf
8	Permissions	Verify	puppet	puppet-perms.cf
9	Content	Verify	cf3	cf3-content.cf
10	Content	Verify	puppet	puppet-content.cf
11	Hosts records	Verify	cf3	cf3-hosts.cf
12	Hosts records	Verify	puppet	puppet-hosts.cf

Each amount of work was scaled up such that it was possible to actually measure some resource and time consumption for both tools. For the file permissions and file content tests, a file tree with known content and permissions was created under a test directory */var/tmp/file_tests*.

The test file tree was made each time as follows:

```
TESTDIR=/var/tmp/file_tests
if [[ -d $TESTDIR ]]
then
   rm -rf $TESTDIR
fi
for i in `seq 1 10`
do
   mkdir -p $TESTDIR/dir_$i
   for j in `seq 1 10`
   do
      echo "Some testline" > $TESTDIR/dir_$i/file_$j
   done
done
```

Run as root, this creates directories with ownership root:root and mode 755 and files with ownership root:root and mode 644.

For the file permissions, the tools' work was to converge from the default permissions to ownership root:bin and mode 755 for all files and directories.

Here's the Puppet configuration for applying file permissions (puppet-perms.cf):

```
class fix_perms {
   file { "/var/tmp/file_tests":
      owner => "root",
      group => "bin",
      mode  => 0755,
      recurse => inf,
      backup => false,
   }
}
```

```
node localhost {
  include fix_perms
}
```

And here's the Cfengine configuration for applying file permissions (cf3-perms.cf):

```
body common control
{
bundlesequence  => { "perms" };
}

bundle agent perms
{

files:
   "/var/tmp/file_tests/"
   pathtype => "literal",
   perms => passthrough("0755","root","bin"),
   depth_search => recurse("inf");
}

body depth_search recurse(d)
{
   depth => "$(d)";
}

body perms passthrough(m,o,g)
{
   mode  => "$(m)";
   owners  => { "$(o)" };
   groups  => { "$(g)" } ;
}
```

For file content, the tools' work was to ensure some particular content in the same files.

Here's the Puppet configuration for applying file content (puppet-content.cf):

```
class fix_contents {
   file { "/var/tmp/file_tests":
      content  => "Trallala\n",
      recurse => inf,
      backup => false,
   }
}

node localhost {
   include fix_contents
}
```

And here's the Cfengine configuration for applying file content (cf3-content.cf):

```
body common control
{
bundlesequence  => { "content" };
}

bundle agent content
{

files:
```

```
    "/var/tmp/file_tests/.*/.*"
    edit_defaults => no_backup,
    edit_line => en_liten_trall;
}

bundle edit_line en_liten_trall
{
    delete_lines:
    ".*";
    insert_lines:  "Trallala";
}

body edit_defaults no_backup
{
    edit_backup => "false";
}
```

For host entries, the tools' work was to ensure that all specified host-to-IP mappings were present in /etc/hosts.

Here's the Puppet configuration for applying host entries (puppet-hosts.cf):

```
class my_hosts {
    host { "private1.localdomain.com":
    ip => "10.0.0.1",
    ensure => present,
}

host { "private2.localdomain.com":
    ip => "10.0.0.2",
    ensure => present,
}

(...)

host { "private254.localdomain.com":
    ip => "10.0.0.254",
    ensure => present,
    }
}

node localhost {
    include my_hosts
}
```

And here's the Cfengine configuration for applying host entries (cf3-hosts.cf):

```
body common control
{
    bundlesequence  => { "hosts" };
}

bundle agent hosts
{
        vars:

     "my_hosts" slist => {
         "10.0.0.1 private1.localdomain.com",

         (...)

    "10.0.0.253 private253.localdomain.com",
    "10.0.0.254 private254.localdomain.com"
```

```
};

files:

  "/etc/hosts"
  edit_defaults => no_backup,
  edit_line => host_ensure("@(hosts.my_hosts)");
}

bundle edit_line host_ensure(record)
{
        insert_lines:

"$(record)";

}

body edit_defaults no_backup
{
  edit_backup => "false";
}
```

When measuring verification time, the script converged configuration of the tool before taking the measurements. This way it is unlikely that the pre-condition is anything other than the desired state in the tool configuration; hence the measured values when doing this are verification only.

Application of the configurations with Puppet was done like this:

```
/usr/sbin/puppetd --no-daemonize --onetime
```

Application of the configurations with Cfengine was done like this:

```
/var/cfengine/bin/cf-agent --no-lock
```

To have as close to equal starting points as possible for the tests, all block device buffers were dropped before each test by using:

```
sysctl -w vm.dropcaches = 3
```

The Scientific Method

When measuring alternatives there will be uncertainty in the measurements. The uncertainty is defined as error, or noise. The total error was quantified using repeated measurements and statistical methods for finding the confidence intervals of the differences between alternatives. Confidence intervals of the differences comes with a probability of the true value being inside the interval. For the confidence interval to have any utility, the probability that the true value is inside the interval must be high. Higher probability widens the confidence interval, and vice versa. A commonly chosen value of this probability is 0.95.

If confidence intervals of the two alternatives overlap, it is impossible to say that the difference is not caused by random fluctuations. If they don't overlap, there is no evidence to suggest that there is *not* a statistically significant difference. The chosen probability quantifies the certainty of being right in assuming there is a true difference [10, p. 43].

Plotting the values shows random variations, but the true standard deviation of the underlying population is not known. The student's *t* distribution takes this into account and is commonly used for detecting statistically significant differences between two alternatives [8]. The student's *t* distribution, or more precisely the *t*-test, was used to identify any statistically significant differences in the experiment results.

The tool "gnu time" [4] (version 1.7 release 27.2.2) was used for measuring time and resource consumption of each operation. For each measurement, three metrics were logged:

- Total time used
- Number of CPU seconds (system + user)
- Number of involuntary context switches

The values were logged in semicolon-separated files, one for each $measurement-$tool containing 40 lines of data. Each column of measurements then represents one metric (time, CPU, or CSWITCH) measurement repeated 40 times. These vectors were used for calculating sample means and confidence intervals using the free statistic program R [6].

For all 12x3 metrics, the *t*-test functions of *R* were used to compare pairs of vectors of 40 numbers, each representing the measured values of each tool, respectively. The outcome of each comparison is the sample mean of the difference between vectors and its confidence interval given a certain probability that the true value is within the interval.

The probability used for the tests was 0.99, meaning there is 99% probability that the true value of the sample mean of the differences is within the confidence intervals produced by the *t*-tests.

All *t*-tests were done as follows:

```
diff = t-test(puppet_vector,cfengine_vector,conf.level = 0.99)
```

and the return values are fetched out as follows in *R*:

```
c(diff$estimate[1],diff$estimate[2],diff$conf.int[1],diff$estimate[1] -
diff$estimate[2] , diff$conf.int[2])
```

The five values produced from the two statements above correspond to columns 3–7 in the results table.

Results

The output of the eighteen *t*-tests in *R* is summarized in the following table.

Table legend:

1. Type of workload: Permission/Content/Host Converge/Verify
2. Resource/time measurement
3. Sample mean value for Puppet
4. Sample mean value for Cfengine
5. Start of the confidence interval of the sample mean difference (C1)
6. Sample mean difference
7. End of the confidence interval of the sample mean difference (C2)

Workload	Measurement	Puppet mean	Cf. mean	C1	Mean difference	C2
Permissions conv.	Execution time	14.80s	1.18s	13.54s	13.63s	13.72s
Content conv.e	Execution time	14.85s	1.43s	13.24s	13.42s	13.6s
Hosts conv.	Execution time	28.44s	1.21s	26.19s	27.23s	28.27s
Permissions ver.	Execution time	14.28s	1.25s	12.82s	13.04s	13.25s
Content ver.	Execution time	14.32s	1.23s	12.92s	13.09s	13.27s
Hosts ver.	Execution time	21.52s	1.11s	20.30s	20.41s	20.53s
Permissions conv.	cpu seconds	11.03s	0.24s	10.77s	10.79s	10.81s
Content conv.	cpu seconds	11.03s	0.36s	10.64s	10.67s	10.69s
Hosts conv.	cpu seconds	10.50s	0.32s	10.16s	10.18s	10.19s
Permissions ver.	cpu seconds	11.04s	0.23s	10.78s	10.8s	10.8s
Content ver.	cpu seconds	11.03s	0.34s	10.67s	10.69s	10.72s
Hosts ver.	cpu seconds	17.97s	0.32s	17.61s	17.65s	17.68s
Permissions conv.	forced cswitch	1861	150	1595	1711	1827
Content conv.	forced cswitch	1918	159	1644	1759	1874
Hosts conv.	forced cswitch	3194	155	2929	3039	3148
Permissions ver.	forced cswitch	1933	151	1675	1782	1889
Content ver.	forced cswitch	1967	160	1708	1808	1907
Hosts ver.	forced cswitch	3186	159	2913	3027	3142

The following graphs are produced by the package Sciplot [7] in R. The error bars show the 99% confidence interval of each sample mean.

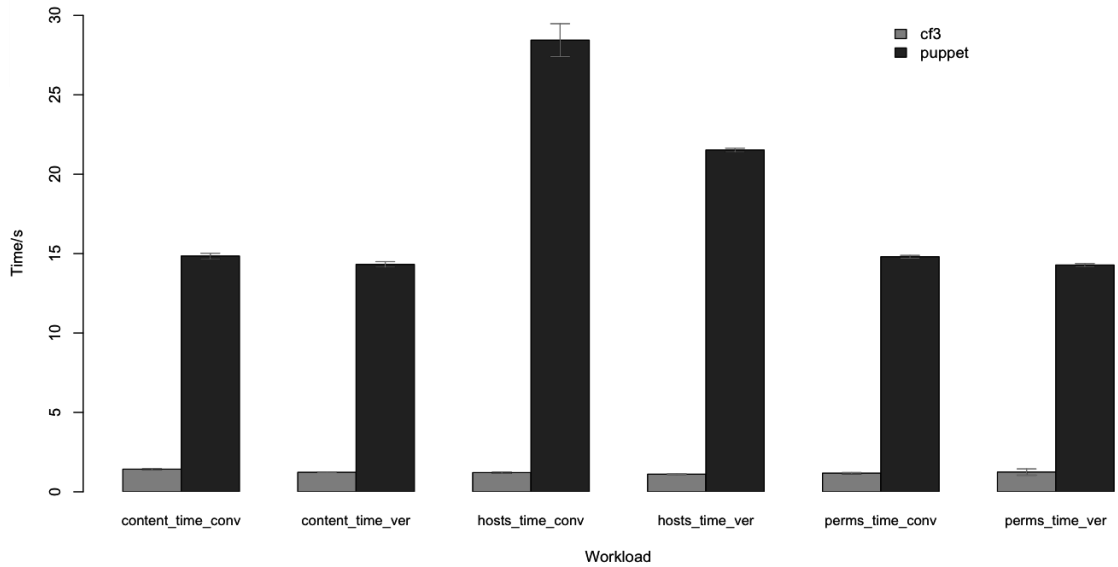


FIGURE 1: TIME USAGE FOR THE SIX TASKS

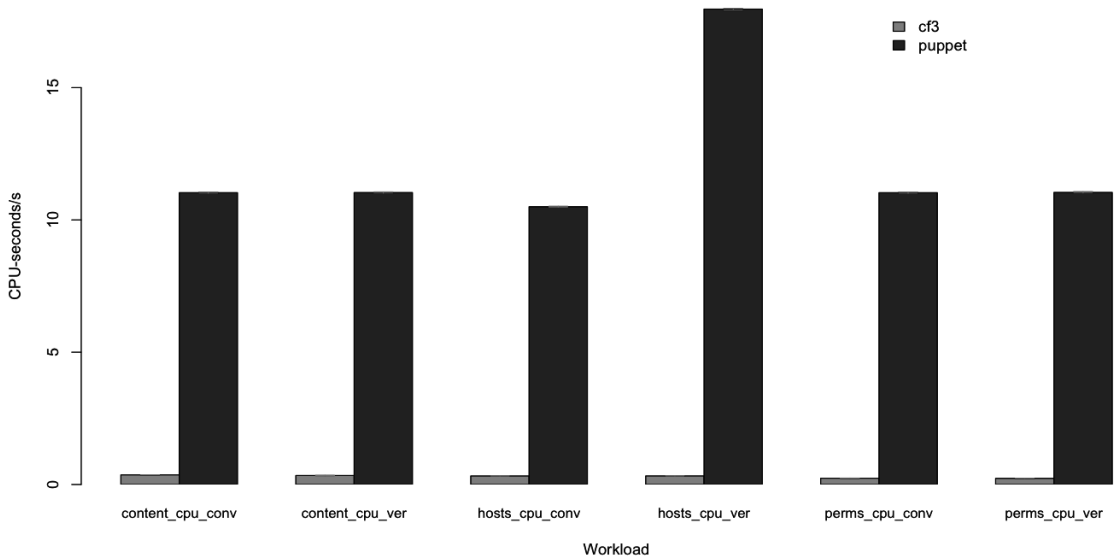


FIGURE 2: CPU SECONDS USED

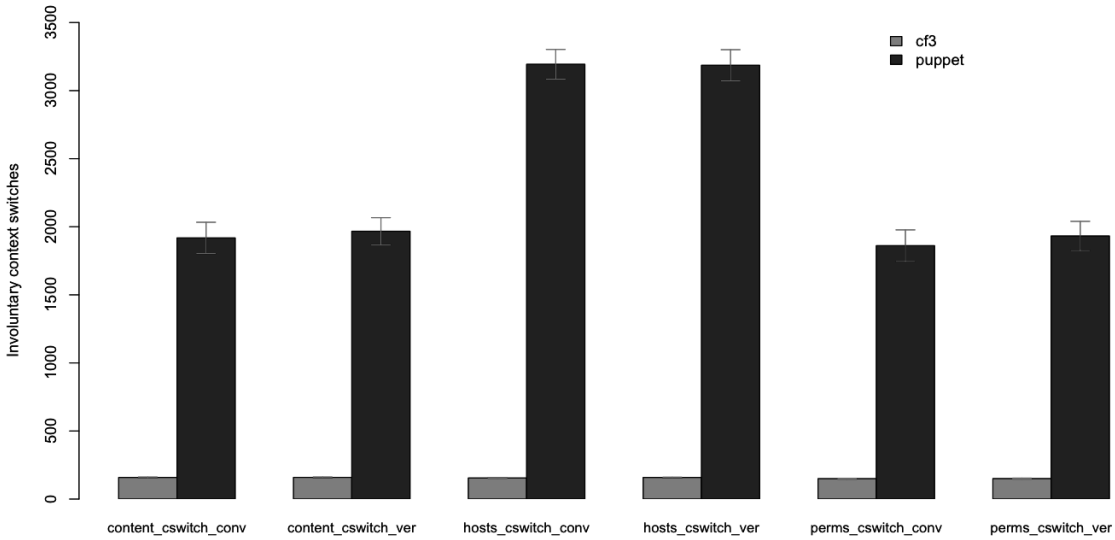


FIGURE 3: NUMBER OF INVOLUNTARY CONTEXT SWITCHES

Conclusion

The results show that Puppet uses considerably more time and resources than Cfengine3 for all the measurements included in the experiment. Time and resource usage is important, particularly in the verification phase. Verification is done every time the agent runs, regardless of compliance. The maximum frequency of verifications will be affected by time usage for each verification. The scope of the verified configuration in the experiment is small compared to what can be the case in real production environments. Clearly, time usage of verifications will limit the frequency of verifications when the scope increases.

Generally it is also desirable to have low resource consumption on administrative processes that run regularly, both from an environmental and from a capacity point of view.

The experiment shows that usage of Puppet involves a major tradeoff with respect to time and resource consumption compared to Cfengine3 for the operations that were measured.

Of course, there are many factors to consider when choosing configuration management tools. The differences of time and resource consumption might be ignorable to some. The results of this experiment serve as a supplement, broadening the understanding of the differences between these two popular configuration management tools.

REFERENCES

[1] http://fedoraproject.org/wiki/EPEL.

[2] http://www.cfengine.org/manuals/cf3-reference.html.

[3] http://cfengine.com/pages/whatIsCfengine.

[4] http://www.gnu.org/software/time/.

[5] http://reductivelabs.com/products/puppet/.

[6] http://www.r-project.org.

[7] http://cran.r-project.org/web/packages/sciplot/index.html.

[8] http://en.wikipedia.org/wiki/Student's_t-distribution.

[9] http://www.usit.uio.no.

[10] D.J. Lilja, *Measuring Computer Performance: A Practitioner's Guide* (Cambridge University Press, 2000).

JOCHEN L. LEIDNER AND
GARY BEROSIK

building and installing a Hadoop/MapReduce cluster from commodity components: a case study

Jochen L. Leidner, Ph.D., is a research scientist in the corporate Research and Development group at Thomson Reuters and a director at Linguit Ltd. He holds a doctorate degree in Informatics from the University of Edinburgh, where he has been a Royal Society Enterprise Fellow in Electronic Markets and a postdoctoral researcher, and he has master's degrees in Computational Linguistics, English Language and Literature, and Computer Speech, Text and Internet Technology. His research interests include natural language processing, search engines, statistical data mining, and software engineering. Jochen is a member of ACM, ACL, and SIGIR and has co-authored over 20 peer-reviewed papers and several patent applications.

leidner@acm.org

Gary Berosik is a lead software engineer at Thomson Reuters Research and Development and an adjunct faculty member in the Graduate Programs in Software at the University of St. Thomas in St. Paul, MN. His interests include software engineering, parallel/grid/cloud processing, statistical machine learning algorithms, learning-support technologies, agent-based architectures, and technologies supporting business intelligence and information analytics.

gary.berosik@thomsonreuters.com

WE DESCRIBE A STRAIGHTFORWARD way to build, install, and operate a compute cluster from commodity hardware. A compute cluster is a *utility* that allows you to perform larger-scale computations than are possible with individual PCs. We use commodity components to keep the price down and to ensure easy availability of initial setup and replacement parts, and we use Apache Hadoop as middleware for distributed data storage and parallel computing.

Background

At the time of writing, single desktop computers and even mobile devices have become faster than the supercomputers of the past. At the same time, storage capacities of disk drives have been increasing by multiple orders of magnitude. As a result of mass production, prices have decreased and the number of users of such commodity machines has increased. Meanwhile, pervasive networking has become available and has led to the distribution and sharing of data and, consequently, distributed communication, creation, consumption, and collaboration. Perhaps paradoxically, the ever-increasing amount of digital content that is the result of more powerful machine storage and networking is intensifying the demand for information and for making sense of activities, preferences, and trends. The analysis of large networks such as the World Wide Web is such a daunting task that it can only be carried out on a network of machines.

In the 1990s, Larry Page, Sergey Brin, and others at Stanford University used a large number of commodity machines in a research project that attempted to crawl a copy of the entire Web and analyze its content and hyperlink graph structure. The Web quickly grew, becoming too large for human-edited directories (e.g., Yahoo) to efficiently and effectively point people at the information they were looking for. In response, Digital Equipment Corporation (DEC) proposed the creation of a keyword index of all Web pages, motivated by their desire to show the power of their 64-bit Alpha processor. This effort became known as the AltaVista search engine. Later, the aforementioned Stanford group developed a more sophisticated search engine named BackRub, later renamed Google.

Today, Google is a search and advertising company, but is able to deliver its innovative services only due to massive investments in the large-scale

distributed storage and processing capability developed in-house. This capability is provided by a large number of PCs, the Google File System (GFS), a redundant cluster file system, and MapReduce, parallel data processing middleware. More recently, the Apache Hadoop project has developed a reimplementation of parts of GFS and MapReduce, and many groups have subsequently embraced this technology, permitting them to do things that they could not do on single machines.

MapReduce/Hadoop Concepts

The key innovation of Hadoop [3], modeled after Google's MapReduce [11], is to eliminate synchronization problems by imposing a programming model that makes it possible to automatically address synchronization issues under the hood. This distributed programming model was inspired by functional languages like LISP and SML. In functional programming, new data is created from old data without modifying state, and this model is more suitable for parallelization, as synchronization issues are less often an issue. Each MapReduce task comprises two phases: (a) a Map step iterates over a set of data elements (called splits or slices), creating a key-value pair representation, and, optionally, carrying out other computations on each element; and (b) a Reduce step that transforms the set of data elements produced by the Map step into a single data element.

A naive example would be to compute the sum of squares of a time series. A vector of input numbers, [3, 1, 2, 19, 3] could each be squared independently from one another by a Map task, resulting in the intermediate tuple vector [(1, 9), (1, 1), (1, 4), (1, 361), (1, 9)]. The keys are artificially the same for all tuples to ensure that each tuple gets processed by the same Reducer, as there is an implicit sort step, not shown in Figure 1, that happens after all Mappers have completed but before the Reducer starts. Then the Reduce step could carry out the summation, aggregating the data vector to the single scalar 384. Jobs submitted by the user to the Hadoop/MapReduce system get broken down to a set of tasks (i.e., there may be many parallel Mappers working on slices of the data: see Figure 1). The key innovation of HDFS, a redundant distributed file system modeled after Google's GFS [10], is to optimize storage and streaming access to large files on commodity hardware. This is achieved by means of *n*-time replication (e.g., *n*=3 means every file is held on three machines at any one time, and if one of them dies, another copy will be created to make up for the loss). HDFS blocks on disk are large (typically 64 MB). The command-line program hadoop can take UNIX-style commands as arguments to list, copy, remove, etc., files as well as to upload from the local file system to HDFS and back.

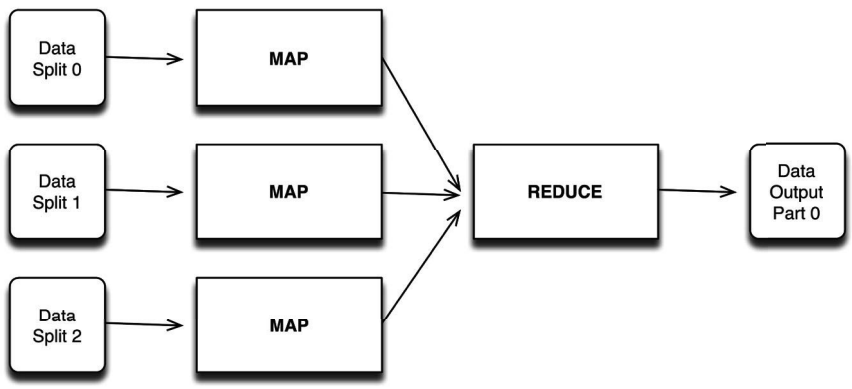


FIGURE 1: MAPREDUCE PROCESSING MODEL

Procurement

We describe how to build such a cluster next. We choose the GNU/Linux operating system because it is very efficient, scalable, stable, secure, is available in source code without licensing impediments, and has a large user base, which ensures rapid responses to support questions. We select the Ubuntu distribution of the operating system because it has good support and a convenient package management system and is offered in a server edition that contains only server essentials (Ubuntu Server). At the time of writing, release 9.07 was current. Little, if anything, of the described approach depends on this particular version. You will need $5,000–$8,000 for a 10-node installation and two person-days of your time. You will also need the following components:

- **Master**: We need a desktop PC running Linux as a master machine. In most cases, no master purchase will be necessary, because an existing high-end PC will be available. In one installation, we picked a Dell Optiplex (2x1 TB HDD configured as RAID, 24" TFT display), which was already available. Cost: $0 (if you don't have a main machine yet, I'd suggest ordering a Mac Pro or a Dell for $3,000).
- **Switch**: Using a professional-grade Gigabit switch (e.g., the auto-configuring Netgear ProSafe 24) means we can save valuable time for setup; it has 24 ports, giving us room for expansion. Gigabit Ethernet is important (and now affordable), as there is going to be a lot of traffic between nodes.
- **Network cabling**: Get one CAT6 patch cable per node to connect it with the switch. Cost is $18 + $3*n*, where *n* = number of nodes; when ordering in bulk, plan on $40 for a 10-node cluster.
- **Nodes**: We should plan for at least three slave-node PCs in order to have an advantage over a single, powerful desktop PC or server and to deploy the various core Hadoop processes. Since we are drawing from commodity components, we should pick an attractive package deal rather than waste time on customizing a node. Criteria are price, CPU speed, number of cores, RAM, number of hard disk slots, energy consumption, cooling, and noise. The hard disk drive size is not a criterion, because they are going to be replaced (cheap commodity PC deals include only very small disk drives). The amount of RAM is important, but since it can be cheaply replaced what matters more is the potential rather than the existing memory size: 2 GB should be considered the absolute minimum, 4 GB/node RAM is recommended (16 GB would be ideal, but at the time of writing is unlikely to be found in consumer machines). For the cluster shown in Figure 2, Acer X2 nodes with a dual-core Athlon X2 64-bit CPU (1.2 GHz), 3 GB RAM, 320 GB HDD were selected because they offer 4,800 bogo-MIPS/core for around $400. Cost is $400*n*; plan on $4,000 for 10 nodes.
- **Hard disk drives** (1 TB or higher): Determine the type of hard disk drive with the lowest $/TB cost. High-capacity drives (at the time of writing, 2 TB) are overly expensive, and the average size of drives used in commodity PCs as you buy them is too small, so the drives they come with (e.g., 320 GB) have to be replaced. Cost is $100n: plan on $1,000 for 10 nodes.
- **Enclosure** (shelf or rack): The nodes, switch, etc., need to live somewhere. A 42U 19" rack is the standard for data centers, however it may prove an unreasonable choice for several reasons: first, the cost of a new rack could easily exceed the total of the hardware expenses for the cluster itself, and, second, since the nodes are commodity machines as opposed to "professional" 19" servers, they may be hard to fix, so the main advantage of the 19" rack may be lost on them.
  Alternatives are cheap IKEA shelves made from wood or metal or anything similar. Finding a solution with wheels to keep the cluster mobile avoids

having to disassemble it, should the need for relocating it arise. Cost: approximately $400 (but if you are a system administrator, it is highly likely that you already have a spare rack or shelf somewhere).

- **Socket multiplier**: Use a fused socket multiplier to cope with the plugs of all nodes and the master. Cost: $20.

FIGURE 2. HYDRA, A MINIATURE CLUSTER WITH ONE MASTER PC AND THREE NODES, IS THE FIRST AUTHOR'S PRIVATE CLUSTER FOR HOME OFFICE USE. THIS IS ONE OF A SERIES OF INSTALLATIONS OF VARYING SIZES WE HAVE BEEN WORKING ON.

Physical Setup and Assembly

It is important to pick a suitable location for the cluster up front. Power consumption for a cluster of reasonable size will be considerable. For instance, 10 nodes of Acer X1700 at 220 watts each amount to 2200/120 = 18.4 amperes, which is just under the limit of what the circuit breaker of a typical household or small office will be able to carry. In addition, consider the significant heat generation that a large cluster can create.

Upgrade RAM and insert HDD drives. If you use a closed rack that comes with a door (recommended to keep the noise level down, but even more expensive than its open siblings), you may consider removing all cases of the node PCs altogether to improve ventilation (this "naked" configuration was pioneered by Google).

Operating System Installation

Download Ubuntu Server 9.07 or higher and burn a medium; connect all nodes with the switch using the CAT6 cables; connect all nodes with the power source. Connect the master PC's screen to the first node to be installed, insert the Ubuntu CD to boot, and install Ubuntu Server. Set up the HDD partitions as follows:

- **boot** (1 GB) not mounted, ext3, bootable, primary (important: this one should be at the beginning of the disk, or you may run into BIOS boot problems, e.g., "grub error 2")
- **root** (50 GB) mounted as /, ext3, logical

- **main** (0.94 TB) mounted as /var (var is a standard convention indicating variable data, i.e., a high amount of input/output is to be expected), ext3, logical
- **swap** (2 GB) not mounted, type swap, logical using manual partitioning (last option)
- Don't use logical volume management (LVM) or encryption.

Set the time zone to UTC/GMT and install software packages:

- Ubuntu server (always included, no action required)
- LAMP
- OpenSSH

You will also have to give the computer a name (e.g., the cluster in Figure 2 is called Hydra, so its nodes are called hydra1, hydra2, . . .). After the installation you should be able to re-boot and to issue:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get install sun-java6-jdk xorg gdm xfce4 xemacs21
```

Since our example cluster is experimental, we wanted to be able to run X11 on the nodes; every package from xorg onwards is not necessary for a production cluster. Depending on the tasks that we anticipate using the cluster for, we may want to consider installing additional packages. One of us does a lot of Web research, data crawling, analytics, and statistical machine learning, so it makes sense to get some crawlers, the R statistics system, and libraries for numeric computing:

```
sudo apt-get install wget curl lynx r-base r-base-dev python-numpy python-scipy
```

At this point, our first node is operational as far as the operating system is concerned, but in order to make combined used of its nodes as a single quasi-utility we still need to install Apache Hadoop and then replicate the setup to the other nodes.

Example Hadoop Installation on a Small Cluster

After the installation of the operating system, we can now turn to the setup of Apache Hadoop as our middleware for redundant storage and parallel processing. When we are done, we should have three processes running on our three nodes, as follows:

```
hydra1  Acer X1700  Master     NameNode;DataNode;JobTracker
hydra2  Acer X1700  Slave      SecondaryNameNode;DataNode;TaskTracker
hydra3  Acer X1700  Slave      DataNode;TaskTracker
```

Note that the use of the word Master here pertains to Hadoop and is distinct from the cluster master PC (which has the keyboard and screen attached).

1. Ensure that Java is set up. The latest Hadoop releases depend on Java 6.x or later. Download/install/test the latest 6.x Java release if this is not already set up.

It is recommended to follow the procedures for single node cluster setup as described in the online article by Michael G. Noll on running Hadoop in Ubuntu Linux environments [1].

We suggest you do this on each machine of a multi-node cluster to help verify the operational status of Hadoop on each node before continuing to set up a multi-node configuration. The steps below show examples of following these instructions.

2. Change to user hadoop. Add a hadoop group and hadoop user for that group.

```
<your-user-name>@hydra1:~$ sudo addgroup hadoop
<your-user-name>@hydra1:~$ sudo adduser --ingroup hadoop hadoop
```

3. Add the following exports for proxy and JAVA_HOME to the .bashrc file for both your user and the new hadoop user:

```
export http_proxy=<yourProxy>:<proxyPort>
export JAVA_HOME=<yourJavaHomePath>
```

4. Configure and test SSH operation on all nodes (required by Hadoop).

As the hadoop user, on the master node, create the RSA key:

```
hadoop@hydra1:~$ ssh-keygen -t rsa -P ""
```

Copy or append the new key to the authorized_keys file in the .ssh directory:

```
hadoop@hydra1:~$ cat /home/hadoop/.ssh/id_rsa.pub >> ~hadoop/.ssh/
    authorized_keys
```

Try to connect to the local machine with the hadoop user. Respond with a yes when prompted to "continue connecting."

```
hadoop@hydra1:~$ ssh localhost
```

5. Download Hadoop 0.19.2

As of this writing, there are known instability issues with the 0.20 release, so release 0.19.2 is used for this installation. For example, download from:

```
http://newverhost.com/pub/hadoop/core/hadoop-0.19.2/hadoop-0.19.2.tar.gz
```

(Note: there are also API differences between 0.19 and 0.20+.) Now, with administrator permissions, install this release in the desired directory location. The example installation steps below assume the original download was to the directory location: /home/<your-user-name>/Desktop. Note that *these steps must be performed as root*.

Uncompress the Hadoop release to the desired location:

```
root@hydra1:/usr/local#  tar xzf /home/<your-user-name>/Desktop/
hadoop-0.19.2.tar.gz
```

Rename the release as desired, and change ownership of all the release contents to permit use by the hadoop user:

```
root@hydra1:/usr/local# mkdir /var/hadoop
root@hydra1:/usr/local# ln –s /var/hadoop hadoop
root@hydra1:/usr/local# mv hadoop-0.19.2/* hadoop/*
root@hydra1:/usr/local# chown -R hadoop:hadoop hadoop
```

6. Add an export for HADOOP_HOME to the .bashrc file for both your user and the hadoop user:

```
export HADOOP_HOME=<yourHadoopHomePath>
```

7. Edit the file /usr/local/hadoop/conf/hadoop-env.sh by uncommenting the export for JAVA_HOME and setting it to the correct value:

```
export JAVA_HOME=<yourJavaHomePath>
```

8. Edit the file /usr/local/hadoop/conf/hadoop-site.xml to contain single node test configuration settings. Adjust values to suit your own configuration needs. There are *many* possible default configuration parameter settings

that can be adjusted. See the /usr/local/hadoop/conf/hadoop-defaults.xml file for more information about the complete set of adjustable parameters.

```
<configuration>

<property>
  <name>hadoop.tmp.dir</name>
  <value>/usr/local/hadoop/tmp/datastore/hadoop-${user.name}</value>
  <description>
  A base location for other temp datastore directories.
  </description>
</property>

<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
  <description>
  The name of the default file system.
  A URI whose scheme and authority determine the filesystem implementation.
  The URI's scheme determines the config property (fs.SCHEME.impl) naming
  the filesystem implementation class. The URI's authority is used to
  determine the host, port, etc. for a file system.
  </description>
</property>

<property>
  <name>mapred.job.tracker</name>
  <value>localhost:54311</value>
  <description>
  The host and port that the MapReduce job tracker runs at.
  If "local," then jobs are run in-process as a single map and reduce task.
  </description>
</property>

<property>
  <name>dfs.replication</name>
  <value>1</value>
  <description>
  Default block replication.
  The actual number of replications can be specified when the file is created.
  The default is used if replication is not specified at create time.
  </description>
</property>

</configuration>
```

Note that dfs.replication specifies the number of copies of each file that is kept on the cluster by HDFS's redundancy mechanism. For a single "pseudo-cluster" setup, we set this to 1 until that node is operational, then we must change it (e.g., back to its default of 3).

9. With administrator privileges, create the hadoop tmp/datastore directory for your user and the hadoop user and change ownership to allow use by the hadoop and your user:

```
root@hydra1:/usr/local/hadoop# mkdir –p tmp/datastore/hadoop-hadoop
root@hydra1:/usr/local/hadoop# chown -R hadoop:hadoop tmp/datastore/
    hadoop-hadoop
root@hydra1:/usr/local/hadoop#mkdirtmp/datastore/hadoop-<your-user-name>
root@hydra1:/usr/local/hadoop# chown -R <your-user-name>:<your-user-
    name>   tmp/datastore/hadoop-<your-user-name>
```

10. Test Hadoop execution in single-node mode, using HDFS.

As the hadoop user, format the NameNode:

```
hadoop@hydra1:~$ $HADOOP_HOME/bin/hadoop namenode -format
```

Start the (single-node) cluster:

```
hadoop@hydra1:~$ $HADOOP_HOME/bin/start-all.sh
```

Verify that the expected Hadoop processes are running using Java's jps:

```
hadoop@hydra1:~$ jps
27069 JobTracker                26641 NameNode
26729 DataNode                  29425 Jps
26923 SecondaryNameNode         27259 TaskTracker
```

With administrator permission, use netstat to verify that Hadoop is listening on the expected/configured ports. For example:

```
root@hydra1:~# netstat -plten | grep java | grep 127.0.0.1
tcp6  0  0 127.0.0.1:54310  :::*          LISTEN  1001   590635 26641/java
tcp6  0  0 127.0.0.1:54311  :::*          LISTEN  1001   594563 27069/java
tcp6  0  0 127.0.0.1:51633  :::*          LISTEN  1001   601104 27259/java
```

11. Set up and run a Hadoop example test application to verify operability.

Adjust the <HadoopHome>/conf/hadoop-env.sh file to set JAVA_HOME, HADOOP_HOME, and a reasonable CLASSPATH (if desired) for Hadoop executions:

```
# == file "hadoop-env.sh" ==
export JAVA_HOME=/usr/lib/jvm/java-6-sun/jre
export HADOOP_HOME=/usr/local/hadoop
export CLASSPATH=$HADOOP_HOME/hadoop-0.19.2-core.jar:$HADOOP_
    HOME/hadoop-0.19.2-examples.jar:$HADOOP_HOME/hadoop-0.19.2-test.
    jar:$HADOOP_HOME/hadoop-0.19.2-tools.jar:$CLASSPATH:$classpath
```

Run the test example program. The following example executes the pi program included in the distributed Hadoop examples. Use the source command to ensure that the settings are kept by the executing shell process.

```
hadoop@hydra1:~$ source hadoop-env.sh
hadoop@hydra1:~$ $HADOOP_HOME/bin/hadoop jar          $HADOOP
    _HOME/hadoop-0.19.2-examples.jar pi 2 10
```

The output should look similar to the following:

```
Number of Maps = 2 Samples per Map = 10
Wrote input for Map #0
Wrote input for Map #1
Starting Job
09/10/22 13:17:50 INFO mapred.FileInputFormat: Total input paths to process : 2
09/10/22 13:17:50 INFO mapred.JobClient:  Running job:
    job_200910221225_0001
09/10/22 13:17:51 INFO mapred.JobClient:    map 0% reduce 0%
09/10/22 13:18:00 INFO mapred.JobClient:    map 50% reduce 0%
09/10/22 13:18:03 INFO mapred.JobClient:    map 100% reduce 0%
09/10/22 13:18:10 INFO mapred.JobClient:    map 100% reduce 100%
09/10/22 13:18:11 INFO mapred.JobClient:  Job complete:
    job_200910221225_0001
09/10/22 13:18:11 INFO mapred.JobClient:  Counters: 16
09/10/22 13:18:11 INFO mapred.JobClient:    File Systems
09/10/22 13:18:11 INFO mapred.JobClient:      HDFS bytes read=236
```

```
09/10/22 13:18:11 INFO mapred.JobClient:     HDFS bytes written=212
09/10/22 13:18:11 INFO mapred.JobClient:     Local bytes read=78
09/10/22 13:18:11 INFO mapred.JobClient:     Local bytes written=218
09/10/22 13:18:11 INFO mapred.JobClient:   Job Counters
09/10/22 13:18:11 INFO mapred.JobClient:     Launched reduce tasks=1
09/10/22 13:18:11 INFO mapred.JobClient:     Launched map tasks=2
09/10/22 13:18:11 INFO mapred.JobClient:     Data-local map tasks=2
09/10/22 13:18:11 INFO mapred.JobClient   Map-Reduce Framework
09/10/22 13:18:11 INFO mapred.JobClient:     Reduce input groups=2
09/10/22 13:18:11 INFO mapred.JobClient:     Combine output records=0
09/10/22 13:18:11 INFO mapred.JobClient:     Map input records=2
09/10/22 13:18:11 INFO mapred.JobClient:     Reduce output records=0
09/10/22 13:18:11 INFO mapred.JobClient:     Map output bytes=64
09/10/22 13:18:11 INFO mapred.JobClient:     Map input bytes=48
09/10/22 13:18:11 INFO mapred.JobClient:     Combine input records=0
09/10/22 13:18:11 INFO mapred.JobClient:     Map output records=4
09/10/22 13:18:11 INFO mapred.JobClient:     Reduce input records=4
Job Finished in 21.342 seconds
Estimated value of PI is 3.2
```

Congratulations! At this point you have a simple, single-node Hadoop environment up and running!

12. Shut down the Hadoop processes in the single-node cluster:

```
hadoop@hydra1:~$ $HADOOP_HOME/bin/stop-all.sh
```

The output should look similar to the following:

```
stopping jobtracker
localhost: stopping tasktracker
stopping namenode
localhost: stopping datanode
localhost: stopping secondarynamenode
```

To configure the Hadoop middleware to handle a *multi*-node cluster, we recommend you follow the procedures for setting up multi-node clusters described in an online article by Michael G. Noll [2].

You now face the issue of having to install the whole node's environment (Linux, packages, Hadoop) from one node to the remaining nodes in a near-identical way. For smaller clusters this can be done manually. For larger clusters with nodes that possibly have different hardware specifications, stronger tools need to be used to define machine classes, separate configurations for each class, and assist in the distribution of these configurations to the appropriate node machines. In these settings, various sources suggest the use of configuration management tools like Puppet [4], Cfengine [5], or Bcfg2 [6]. More concretely, there are several solutions to this, depending on your experience and number of nodes:

1. Burn an ISO image with your setup and use this with the remaining nodes.
2. Insert the empty hard disk drives as secondary drives in the master PC temporarily in order to copy over the entire disk using the dd(1) command.
3. Clone the disk over a network connection using dd(1) and netcat (nc(1)) as outlined by [7].
4. Install the other nodes manually (estimated time: about 30 min/node).

Method 3 is superior for large clusters, but method 4 is fast enough for smaller clusters. Remember that the hostname must be unique, so you may

have to set it manually after cloning the node setups by manually invoking the hostname(1) command for each node.

In order to automate the installation completely, [9] recommends using static IP addresses for the nodes, setting the hostname by keeping a file hostnames.new that contains the node names and their static IP addresses, and then generating a set of node-specific kick-start files from a master template (here called "anaconda-ks.cfg," with NODE_HOSTNAME and NODE_STATIC_IP being placeholders) as follows:

```
for i in $(cat ~/hostnames.new) ; do \
  cat anaconda-ks.cfg | sed s/NODE_HOSTNAME/$i/g | sed s/NODE_STATIC_
   IP/$(grep $i /etc/hosts | awk '{print $1}')/g > ks-$i.cfg ; \
done
```

In this approach, the operating system is booted over the network using the node-specific kick-start file.

There is yet another mode of operation to install Hadoop on more than one node very conveniently: Cloudera Inc., a cluster/cloud computing startup, which recently hired Hadoop architect Doug Cutting, permits you to enter your desired cluster configuration on a Web interface (my.cloudera.com), which automatically creates customized installers (e.g., *.rpm packages) that contain all the cluster configuration information.

Operating the Cluster

Now that your Hadoop cluster is fully operational, we recommend you try out the word count example from the Apache Hadoop tutorial [8], which shows you how the UNIX wc(1) command can be distributed across a cluster.

In general, to use Hadoop, you can choose several modes of operation:

1. Use its native Java API. To do this, you will have to write mapper and reducer classes that extend org.apache.hadoop.mapred.MapReduceBase and implement Mapper<S> and Reducer<T>, respectively (S and T are type signatures).

2. Use Hadoop Streaming. If you already have a set of command-line tools such as taggers, parsers, or classifiers that you would like to utilize, you can invoke them as mappers and reducers, or you can write mappers and reducers in Python, Perl, etc. This is an excellent way to prototype a new system pipeline, but anecdotal evidence suggests the startup cost of scripting language interpreters/compilers may be prohibitive for production use (recoding a Python/Perl program in C++ for use in Streaming or using the native Java API may be faster by a large factor).

3. Use Hadoop's C++ interface. There is a C++ wrapper library, Hadoop Pipes, which uses socket communication to talk to the Hadoop daemon processes directly (avoiding JNI).

Besides the MapReduce parallel processing functionality and the HDFS distributed redundant file system, there are some other useful sub-projects in Apache Hadoop: **Avro** is a system for serialization of data. It is schema-based and uses a fast, compact binary format. **Chukwa** is a sub-system for managing and monitoring the collection and updating of distributed log files. **HBase** is a free open-source database for Hadoop modeled after Google's Bigtable. Facebook's contribution is **Hive**, which comprises a toolkit and library for processing text and logfiles and which contains an interactive environment, a query compiler and evaluation engine for HQL, an SQL-like query language, compiler, driver, and execution engine, as well as a

metastore called SerDe (short for Serialization and Deserialization). **Pig** is a versatile scripting language contributed by Yahoo that permits easy iteration over data records/tuples, sorting, joins, and counting, besides user-defined functions. Users with exposure to SQL will quickly pick up Pig idioms, and a nice property of the interpreter, which is implemented in Java, is that it can execute scripts both inside and outside the Hadoop/HDFS infrastructure. **ZooKeeper** is a centralized service for a couple of things that make distributed computing challenging, namely, maintaining configuration information, naming services, providing distributed synchronization, and providing a notion of groups. For example, the distributed synchronization primitives offered permit the implementation of a distributed queue. The Pig distribution contains some tutorial examples on mining Web search engine queries that we highly recommend; running these will give you an idea of the power of the MapReduce paradigm and its free Hadoop implementation.

Next, you can think of applications in your own areas of interest and express them in terms of mappers and reducers to execute them on your cluster. We are particularly interested in machine learning, information retrieval (indexing, retrieval, clustering), graph analysis of social networks, and data mining from log files. Check out trendingtopics.org for an example data mining application in the area of automatic trend analysis, which was built with Hadoop. Whether you want to sort petabytes of customer records in "record" time (a Hadoop cluster currently holds the record for a sorting benchmark) or crunch logfiles to find hidden statistical relationships in your data, Hadoop is your friend (and you are not alone, as Yahoo, Facebook, Twitter, etc., are heavily relying on it as well). If you seek further inspiration, we recommend the book *Beautiful Data* [13].

Summary and Conclusion

We have described a successfully completed project to build a cluster computing utility from commodity parts. The cluster is affordable (<$5,000), can be built incrementally, and is more powerful than servers that were priced over a quarter million dollars just a few years ago. Hadoop provides powerful OS middleware for large-scale batch processing such as the automatic analysis of large document collections. We expect that in the future, enterprise versions of commodity operating systems will incorporate some of these capabilities, but we hope the above introduction can serve to give the interested reader a head start (for more detailed, recipe-style instructions targeting a non-system administrator audience, also consult [12]).

Happy Hadooping!

REFERENCES

[1] Michael G. Noll, "Running Hadoop on Ubuntu Linux (Single-Node Cluster)": http://www.michael-noll.com/wiki/Running_Hadoop_On_Ubuntu_Linux_(Single-Node_Cluster).

[2] Michael G. Noll, "Running Hadoop on Ubuntu Linux (Multi-Node Cluster)": http://www.michael-noll.com/wiki/Running_Hadoop_On_Ubuntu_Linux_(Multi-Node_Cluster).

[3] Tom White, *Hadoop: The Definitive Guide* (O'Reilly/Yahoo! Press, 2009).

[4] Puppet online information: http://reductivelabs.com/trac/puppet/wiki/DocumentationStart.

[5] Cfengine online information: http://www.cfengine.org/manuals/cf3-reference.html.

[6] Bcfg2 online information: http://trac.mcs.anl.gov/projects/bcfg2/wiki/UsingBcfg2.

[7] Gite Vivek, "Copy hard disk or partition image to another system using a network and netcat (nc)": http://www.cyberciti.biz/tips/howto-copy-compressed-drive-image-over-network.html.

[8] Apache Hadoop Map/Reduce Tutorial online information: http://hadoop.apache.org/common/docs/current/mapred_tutorial.html.

[9] Installing CentOS on a cluster via NFS online information: http://biowiki.org/InstallingCentOSOnClusterViaNFS.

[10] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, "The Google File System," *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003),* Bolton Landing, NY: 29-43.

[11] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI '04)*: 137–150.

[12] Jochen L. Leidner and Gary Berosik (2009), "Building and Installing a Hadoop/MapReduce Cluster from Commodity Components Technical Report": http://arxiv.org/ftp/arxiv/papers/0911/0911.5438.pdf.

[13] Toby Segaran and Jeff Hammermacher, *Beautiful Data: The Stories Behind Elegant Data Solutions* (O'Reilly, 2009).

Disclaimer. All opinions expressed in this article are the authors' and do not reflect any official opinion or endorsement by the Thomson Reuters Corporation.

EREZ ZADOK, VASILY TARASOV, AND PRIYA SEHGAL

the case for specialized file systems, or, fighting file system obesity

Erez Zadok is an Associate Professor of Computer Science at Stony Brook University. His research interests involve file systems and operating systems, performance benchmarking and tuning, and green computing.

ezk@cs.sunysb.edu

Vasily Tarasov is a third year PhD student in the Computer Science Department of Stony Brook University. His scientific interests include operating systems, computer architectures, software engineering, green technologies, and services science.

vtaras@cs.sunysb.edu

Priya Sehgal is a Master's student at the Computer Science Department of Stony Brook University. Her research interests include operating systems, green computing, and computer architecture.

psehgal@fsl.cs.sunysb.edu

THE COMPLEXITY OF MODERN FILE systems has increased drastically in recent decades, and it keeps increasing. The ext2 file system in the Linux kernel has over 8,000 lines of code (LoC), ext3 doubles this, and ext4 doubles it again. A working development version of btrfs file system already has over 52,000 LoC, XFS is over 77,000 LoC, and other network-based file systems easily exceed 100,000 LoC. We believe that the amount of functionality provided in modern file systems is overkill for many of the usage scenarios, and this often hurts performance, energy efficiency, and even reliability [1]. Instead of creating gigantic general-purpose file systems that are hard to develop, debug, maintain, and tune for specific workloads, we propose to develop minimalistic file systems, each tuned for a particular case.

The growth of complexity is mainly caused by the expanding functionality integrated in a file system. In fact, the list of the features supported by modern file systems is impressive: journaling, B-tree-based search for objects, flexible data extents, access control lists (ACLs), extended attributes, encryption, checksumming, etc. ReiserFS allows programmers to write plugins for it; large file systems such as zfs and btrfs integrate complex storage pool management and deduplication. Features such as access-permission checks, hardlinks and symlinks, unlimited file name length and file size, as well as arbitrary directory depths, are no longer considered extra features: any self-respecting file system *must* support them. But should this "must" really be so strict?

Too Many Features

The large variety of features supported by modern file systems is, in part, the desire of file system developers to satisfy as many end users as possible. Depending on the specific situation, different characteristics are required from a file system. In emergency cases, reliability is the most important factor; for storing military data, security is crucial; enterprise servers require high performance; and in mobile platforms, energy efficiency plays an important role. When all corresponding features go into one file system, the final user obtains not

only the functionality *they* require, but also all the functionality that *other* users may need. The number of tunable parameters of a file system grows proportionally to the functionality of a file system. Ext2 alone allows users to specify over 10 format options and over five mount options, resulting in at least a 10×5 = 50 parameter space; often, many of these options are not mutually exclusive, making the parameter space exponential (e.g., as large as 2^{50} in ext's case). It is extremely difficult for the end user to find an optional point in this space where performance is best. Our experiments show that the default format and mount parameters (often considered by the users as universally best) are up to 50% suboptimal and in some cases nearly an order of magnitude worse than a carefully tuned system [2].

From the developers' point of view it is hard to support, maintain, and develop large file systems. Integration of new features takes a lot of time: one needs to ensure that new functionality interoperates correctly with all other features that are already implemented in the file system. Consequently, the amount of effort spent on adding each new feature grows exponentially. The number of different code paths in a large file system is huge, leading to an exponential number of states to explore, which considerably complicates debugging and performance analysis. New developers spend a lot of time understanding the details of a complex file system before they can fix bugs or change file system behavior in some way.

Most of the users do not need all of the functionality incorporated in a modern file system at once. Actually, in certain cases only minimal file system functionality is enough. We held discussions with scientists who sought our help in designing efficient HDF-based file formats for complex images [3]. These scientists have diverse backgrounds—in neutron and X-ray imaging, molecular and structural biology, optical microscopy, macro-molecular imaging, 3D cryo-electron microscopy, and astrophysics—and use various clusters, with a range of file systems installed, analyzing terabyte-sized data sets on a daily basis. It was surprising to find out that they do not care about even basic features available in modern file systems. They do not use hardlinks, softlinks, or ACLs. The sequence of open-unlink-close (which is painful to implement in a file system) as well as directory renaming are very rare in their environments. They do not use deep directories: most files often reside in one flat directory or a shallow hierarchy. Files typically have known names of fixed length. The input and output file sizes in an experiment are often known in advance. Reliability features (e.g., journaling) are usually not crucial, because lost data can be regenerated easily by rerunning an experiment; for long-running experiments, periodic checkpointing is performed at the application level. With all this in mind, many scientists do not have a preferred file system, because most present file systems provide all the bare features the scientists require.

Simpler File Systems

We looked at all the difficulties related to developing and using the functionality in "obese" file systems, as well as the lack of necessity for the full set of features they offer. We propose creating minimalistic file systems with the functionality incorporated only on an as-needed basis. In this case the code size of a file system can be much smaller, which allows programmers to develop the file system quickly and then support it with less effort. Additionally, such file systems can be tuned more tightly for specific workloads, and without creating a myriad of parameters to confuse the end user. Note that inmany cases (e.g., the aforementioned scientists), users already know the target usage of the file system and the characteristics of the workloads

they are running. In our recent work we showed that careful tuning of existing file systems can increase their performance and power efficiency by as much as a factor of nine [2]. Developing a specialized file system would increase these numbers even more.

Creating a new file system is not as hard as one might think. To demonstrate this, we conducted an experiment within the graduate Operating System class at Stony Brook University. Four teams of 2–3 first-year MS students developed a *very simple real file system* (VSRFS). The functionality was limited, but varied from group to group: fixed/variable number of files and file sizes, no directories vs. simple directories, support of extended attributes, time-stamp storing, etc. It took only 3–4 calendar weeks for the students to create a working file system, with code sizes of 1000–2000 LoC. We therefore hypothesize that file system development time is not linear with respect to the code size and that it is easier to develop many small file systems instead of a few larger, feature-rich file systems. To facilitate filesystem development more, one can take advantage of templates technology similar to the one used in FiST for automatic generation of stackable file systems [4]. Another alternative is to design file systems to be modular: minimal sets of features could be loaded on demand based on workload characteristics.

In conclusion, file systems have become kitchen sinks in recent years; they integrate many hard-to-implement features that many do not use. This fact complicates the development of file systems and makes them less efficient for specific usage. We think that the adoption of small custom file systems is a feasible alternative that facilitates development and increases the efficiency of future file systems.

REFERENCES

[1] V. Prabhakaran, N. Agrawal, L.N. Bairavasundaram, H.S. Gunawi, A.C. Arpaci-Dusseau, and R.H. Arpaci-Dusseau. "IRON File Systems," *Proceedings of the 20th ACM Symposium on Operating Systems Principles*, SOSP '05 (ACM Press, 2005), pp. 206–220.

[2] P. Sehgal, V. Tarasov, and E. Zadok, "Evaluating Performance and Energy in File System Server Workloads Extensions," *Proceedings of FAST '10: 8th USENIX Conference on File and Storage Technologies* (USENIX Association, 2010), forthcoming.

[3] The HDF Group, Hierarchical Data Format, 2009: www.hdfgroup.org.

[4] E. Zadok and J. Nieh, "FiST: A Language for Stackable File Systems," *Proceedings of the 2000 USENIX Annual Technical Conference* (USENIX Association, 2000), pp. 55–70.

ANGELOS D. KEROMYTIS

a look at VoIP vulnerabilities

Angelos Keromytis is an associate professor with the Department of Computer Science at Columbia University and head of the Network Security Lab. He is interested in all aspects of systems and network security. He received his Ph.D. from the University of Pennsylvania and his B.Sc. from the University of Crete.

VOICE OVER IP (VOIP) AND INTERNET Multimedia Subsystem (IMS) technologies offer higher flexibility than traditional telephony infrastructures and the potential for lower cost through equipment consolidation and new business models. In this article, I examine the current state of affairs on VoIP/IMS security through a survey of all the 221 known/disclosed security vulnerabilities in the Common Vulnerabilities and Exposures (CVE) database and in IETF RFCs/drafts. My key finding is that the higher complexity of VoIP/IMS systems leads to a variety of attack vectors, many of them caused by unforeseen and unexpected component interactions. A second finding is that what people seem to worry about in VoIP (traffic interception and impersonation) bears no resemblance to the distribution of vulnerabilities actually disclosed. The article concludes with some practical suggestions for securing VoIP systems.

VoIP/IMS refers to a class of products that enable advanced communication services over data networks. While voice is a key aspect in such products, video and other capabilities (e.g., collaborative editing, whiteboard sharing, calendaring) are supported. The key advantages of VoIP are flexibility and low cost. The former derives from the (generally) open architectures and software-based implementation, while the latter is due to new business models, equipment and network-link consolidation, and ubiquitous high-speed broadband connectivity.

As a result, VoIP has seen rapid uptake in both the enterprise and consumer markets. An increasing number of enterprises are replacing their internal phone switches with VoIP-based implementations, both to introduce new features and to eliminate redundant equipment. Consumers have embraced a slew of technologies with different features and costs, including P2P calling, Internet-to-phone network bridging, and wireless VoIP. These new technologies and business models are being promoted by a new generation of startup companies that are challenging the traditional status quo in telephony and personal telecommunications. As a result, a number of PSTN providers have already completed or are in the process of transitioning from circuit-switched networks to VoIP-friendly packet-switched backbones. Finally, as the com-

mercial and consumer sectors go, so do governments and militaries due to cost reduction concerns and the general dependence on commercial off-the-shelf (COTS) equipment for the majority of their IT needs.

Higher complexity is often the price we pay for more flexibility. You can find more details about the complexity found in VoIP in the paper this article is based on [1]. In brief, several factors contribute to architectural, protocol, implementation, and operational complexity:

- The number and complexity of the various features integrated in a product are perhaps the single largest source of complexity. For example, voice and video transmission typically allow for a variety of codecs that may be used in almost-arbitrary combinations.
- Openness and modularity, generally considered desirable traits, allow for a number of independent implementations and products. Each of these comes with its own parameters and design choices. Interoperability concerns and customer feedback then lead to an ever-growing baseline of supported features for all products. A compounding factor to increasing complexity for many of the open VoIP protocols is the "design-by-committee" syndrome, which typically leads to larger, more inclusive specifications than would be the case in the closed, proprietary wireline telephony network from 20 years ago.
- Because VoIP systems are meant to operate in a variety of environments, business settings, and network conditions, they must be highly configurable, increasing complexity. Of particular concern are unforeseen feature interactions and other emergent properties. These have often led to exposed systems through misconfiguration (or poorly understood configuration), as in the case of fraudsters who broke into Internet-accessible VoIP PBXs and routed long-distance calls through them at the expense of the PBX owners; this specific instance was estimated to have cost upwards of $5 million [5, 6]. Another case, enabled by the use of default passwords, was estimated to have cost $55 million [2].
- Finally, VoIP is intended to work over a public data network such as the Internet, or an enterprise/operator network that uses the same underlying technology. As a result, there is a substantial amount of (strictly speaking) non-VoIP infrastructure that is critical for the correct operation of the system, including DHCP, DNS, TFTP/BOOTP, NAT (and NAT traversal protocols such as STUN), NTP, SNMP, routing, the Web (HTTP, TLS/SSL, etc.), and many others. Even a "perfectly secure" VoIP system can be compromised by subverting elements of this infrastructure.

Because of this complexity, manifesting both in terms of configuration options and size of the code base for VoIP implementations, VoIP systems represent a large attack surface. Over time, we should expect to encounter security problems arising from design flaws (e.g., exploitable protocol weaknesses), undesirable feature interactions (e.g., combinations of components that enable new attacks or facilitate known attacks), unforeseen dependencies (e.g., compromised paths through seemingly unrelated protocols), weak configurations, and, not least, implementation flaws.

In trying to understand the threat space against VoIP, my approach is to place known vulnerabilities within a structured framework. While a single taxonomy is not likely to be definitive, using several different viewpoints and mapping the vulnerability space along several axes may reveal trends and areas that merit further analysis. As a starting point, I use the taxonomy provided by the Voice over IP Security Alliance (VoIPSA), available at http://www.voipsa.org/. VoIPSA is a vendor-neutral, not-for-profit organization composed of VoIP and security vendors, organizations, and individuals with an interest in securing VoIP protocols, products, and installations. The classification identifies six broad areas of concern: (1) social threats, (2) traffic

eavesdropping, interception, and modification threats, (3) denial of service (DoS), (4) service abuse, (5) physical access threats, and (6) interruption of services threats. Due to the nature of the vulnerabilities discussed, only the first four categories are relevant to our discussion. I also place the surveyed vulnerabilities within the traditional threat space of confidentiality, integrity, availability (CIA), and consider whether the vulnerabilities exploit bugs in the protocol, implementation, or system configuration.

Many of the vulnerabilities center on the Session Initiation Protocol (SIP), so it is worth highlighting some of its features and discussing its overall complexity. SIP is a protocol standardized by the Internet Engineering Task Force (IETF) and is designed to support the setup of bi-directional communication sessions, including, but not limited to, VoIP calls. It is similar in some ways to HTTP in that it is text-based, has a request-response structure, and even uses a mechanism based on the HTTP Digest Authentication for user authentication. However, it is an inherently stateful protocol that supports interaction with multiple network components (e.g., middleboxes such as PSTN bridges). While its finite state machine is seemingly simple, in practice it has become quite large and complicated—an observation supported by the fact that the main SIP document is the second largest RFC ever (after the encyclopedic "Internet Security Glossary," RFC 4949). Figure 1 shows the number of SIP-related RFCs (and the number of total bytes in these) per year (until May 2009), and a size comparison of the main SIP RFC with respect to the TCP RFC, the five main MIME RFCs, the two Secure MIME (S/MIME) RFCs, and the four main IPsec RFCs. These graphs should provide a quantitative, if indirect, indication of the complexity of SIP.

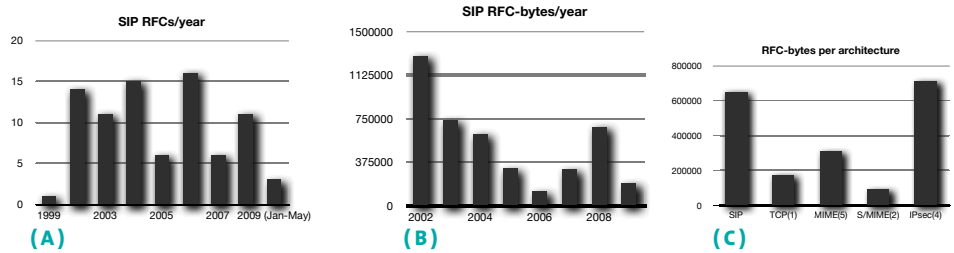


FIGURE 1: BREAKDOWN OF SIP-RELATED RFCS AND THEIR SIZES

For a complete reference to the vulnerabilities surveyed, see an online table at [3].

Overview of VoIP Vulnerabilities

Threats against VoIP system availability by exploiting implementation weaknesses are fairly common. Some implementations were shown to be vulnerable to crashes or hanging when given empty, malformed, or large numbers of SIP INVITE (or other) messages. It is worth noting that the same vulnerability may be present across similar protocols on the same platform and product due to code sharing and internal software structure, or to systems that need to understand VoIP protocols but are not nominally part of a VoIP system. The reason for the disproportionately large number of DoS vulnerabilities is due to the ease with which such failure can be diagnosed, especially when the bug is discovered through automated testing tools (e.g., fuzzers). Many of these vulnerabilities may be more serious than a simple crash and could possibly lead to remote code injection.

Unexpected interactions between different technologies used in VoIP systems can also lead to vulnerabilities. In some cases cross-site scripting (XSS) attacks were demonstrated against the administrator- and customer-facing management interface (which was Web-based) by injecting malicious

JavaScript in certain SIP messages, often through SQL injection vulnerabilities. The same vulnerability could also be used to commit toll fraud by targeting the underlying database. XSS attacks that are not Web-oriented have also been demonstrated, with one of the oldest VoIP-related vulnerabilities permitting shell command execution. Another Web-oriented attack vector is Cross Site Request Forgery (CSRF), whereby users visiting a malicious page can be induced to automatically (without user intervention, and often without any observable indications) perform some action on the Web servers (in this case, VoIP Web-based management interface) that their browser is already authenticated to.

The complexity of the SIP finite state machine has sometimes led to poor implementations. One vulnerability allowed attackers to convince a phone receiving a call to silently complete the call, which allowed the adversary to eavesdrop on the device's surroundings! The same vulnerability could be used to deny call reception at the target, since the device was already marked as busy. In other cases, it is unclear to developers what the use of a specific protocol field may be, in which case they may silently ignore it. Occasionally, such information is critical for the security of the protocol exchange, and omitting or not checking it allows adversaries to perform attacks such as man-in-the-middle or traffic interception, or to bypass authentication checks.

Since SIP devices are primarily software-driven, they are vulnerable to the same classes of vulnerabilities as other software. For example, buffer overflows are possible even against SIP "hardphones" and much more so for softphones, allowing adversaries to gain complete control of the device. Such vulnerabilities typically arise from a combination of poor (non-defensive) programming practices, insufficient testing, and the use of languages, such as C and C++, that support unsafe operations. Sometimes these vulnerabilities appear in software that is not directly used in VoIP but must be VoIP-aware, e.g., firewalls or protocol analyzers. It is also worth noting that these are not the only types of vulnerabilities that can lead to remote code execution. Other input validation failures can allow attackers to download arbitrary files from a user's machine or to place calls by supplying specially encoded URIs or other parameters.

Undocumented on-by-default features are another source of vulnerabilities. These are often remnants from testing and debugging during development that were not disabled when a product shipped. As a result, they often offer privileged access to services and data on a device that would not be otherwise available. One particularly interesting vulnerability allowed an attacker to place outgoing calls through the Web management interface.

A significant class of vulnerabilities in VoIP devices revolves around default configurations, in particular default usernames and passwords. Lists of default accounts are easy to find on the Internet via search engine. Users often do not change these settings; ironically, this seems to be particularly so for administrative accounts, which are rarely (if ever) used in the home/SOHO environment. Other default settings involve NTP servers and DNS servers.

Call interception vulnerabilities are a big concern with VoIP, given the plethora of tools for decoding video and audio streams and the ease of eavesdropping on network traffic, especially on the local subnet. Sometimes such vulnerabilities arise from strange protocol interactions and implementation decisions. For example, caching the location (address) of a VoIP phone based on the IP address used during boot time (using TFTP) seems a reasonable approach; however, since the boot and VoIP stacks are not necessarily tightly integrated, interaction with one protocol can have adverse effects (e.g., changing the perceived location of the phone) in the other protocol. Other

instances of such vulnerabilities involve improper/insufficient credential checking by the registrar or proxy or by the SNMP agent on the VoIP device, which can lead to traffic interception and user impersonation.

The integration of several capabilities in VoIP products, e.g., a Web server used for the management interface, can lead to vulnerabilities being imported to the VoIP environment that would not otherwise apply. In the specific example of an integrated Web server, directory traversal bugs or similar problems (such as lack of proper authentication in the Web interface) can allow adversaries to read arbitrary files or other information from the device. SIP components integrated with firewalls may also interact in undesirable ways. Improper handling of registration requests may allow attackers to receive messages intended for other users. Other such examples include failure to authenticate server certificates in wireless environments, enabling man-in-the-middle and eavesdropping attacks.

Some of the most serious non-implementation types of vulnerabilities are those where the specification permits exploitable behavior. For example, certain vendors permit the actual URI in a SIP INVITE call and the URI used in the Digest Authentication to differ; while arguably allowed by the spec, this enables toll fraud via credential reuse.

While rare, protocol-level vulnerabilities also exist. These represent either outright bugs in the specification or unforeseen interaction between different protocols or protocol components. For large, complicated protocols such as SIP and H.323, where components (code, messages, etc.) are semantically overloaded and reused, it is perhaps not surprising that such emergent properties exist. One good example is the relay attack in the SIP Digest Authentication [4], whereby an adversary can reuse another party's credentials to obtain unauthorized access to SIP or PSTN services (such as calling a premium or international phone line). This attack, depicted in Figure 2, is possible because authentication may be requested in response to an INVITE message at any time during a call, and the responder may issue an INVITE message during a call either automatically (because of timer expirations) or through a user action (e.g., placing the caller on hold to do a call transfer).

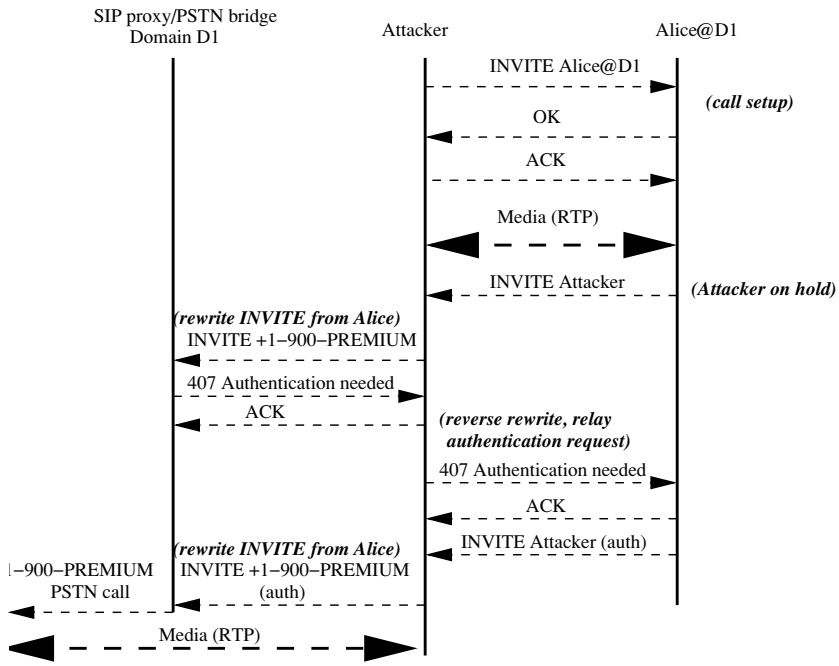


FIGURE 2: SIP RELAY ATTACK

DISCUSSION

I examined 221 vulnerabilities, 219 of which were disclosed in CVE and two as Internet drafts or RFCs. Figure 3 shows the reported number of vulnerabilities per year, up until approximately November 2009. The good news is that there appears to be a large drop in the number of *reported* vulnerabilities in the past two years. The reasons for this drop (and whether it will revert or not) are not known, which is reason enough for caution.

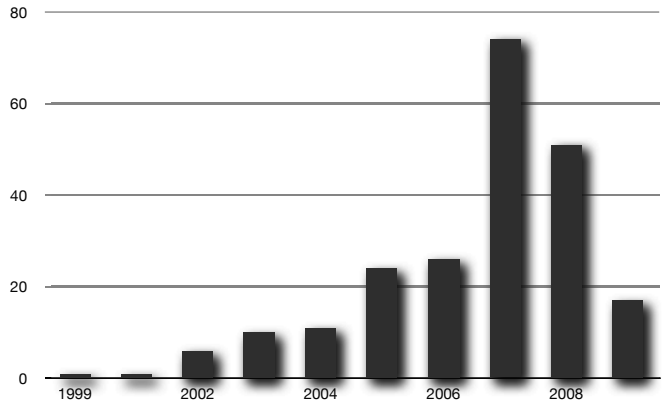


FIGURE 3: VULNERABILITIES PER YEAR

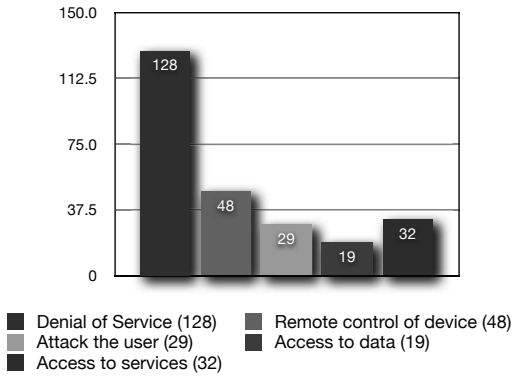


FIGURE 4: BREAKDOWN BY EFFECT

Looking at the vulnerabilities surveyed, a few patterns emerge. An informal classification of vulnerability effects is shown in Figure 4. Most categories are self-explanatory; "attack the user" refers to vulnerabilities that permit attackers to affect the user/administrator of a device, without necessarily compromising the system or getting access to its data or services. XSS attacks and traffic eavesdropping attacks fall in this category, whereas attacks that compromise state (data) resident on the system fall in the "access to data" category.

Half of the problems lead to a DoS in either an end-device (phone, softphone) or a server (proxy, registrar, etc.). This is not surprising, since DoS is easily diagnosed. In many cases, the problem was discovered by automated testing, such as protocol or software fuzzing; software failures are relatively easy to determine in such settings. Some of these vulnerabilities could in fact turn out to be more serious, e.g., a memory corruption leading to a crash could be exploitable in a code injection attack. The second largest class of vulnerabilities allow an adversary to control the device, whether by code injection, default passwords and services, or authentication failures. Note that a few of the vulnerabilities (approximately 10%) were counted more than once in this classification.

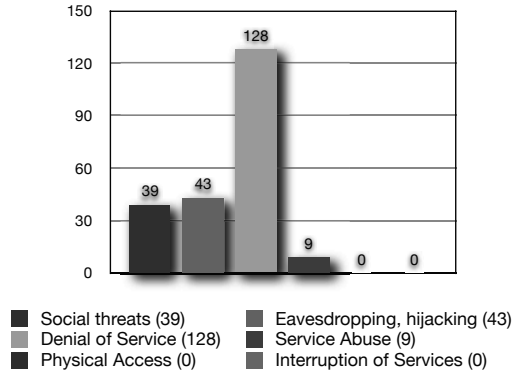


FIGURE 5: VULNERABILITY BREAKDOWN USING THE VOIPSA TAXONOMY

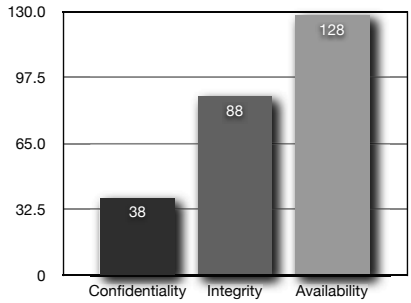


FIGURE 6: VULNERABILITY BREAKDOWN BASED ON (CONFIDENTIALITY, INTEGRITY, AVAILABILITY) CLASSIFICATION

The same pattern with respect to the predominance of DoS vulnerabilities holds when looking at the breakdown according to the VoIPSA taxonomy, shown in Figure 5. It should not be surprising that, given the nature of the vulnerabilities disclosed in CVE, there is no data on physical access and (accidental) interruption of services vulnerabilities. Furthermore, while "Access to services" was a non-negligible component in the previous breakdown, it represents only 4% here. The reason for this apparent discrepancy is in the different definitions of service: the specific element in the VoIPSA taxonomy refers to VoIP-specific abuse, whereas my informal definition covers lower-level system components which may not be usable in, for example, placing fraudulent calls. Another observation is that, while the VoIPSA taxonomy covers a broad spectrum of concerns for VoIP system designers and operators, its categories are perhaps too broad (and, in some cases, imprecise) to help characterize the types of bugs examined.

The vulnerability breakdown according to the traditional (Confidentiality, Integrity, Availability) security concerns again reflects the predominance of DoS threats against VoIP systems, as seen in Figure 6. However, Integrity violations (e.g., system compromise) are a sizable component of the threat space, while Confidentiality violations constitute only 15% of disclosed vulnerabilities. This represents an inversion of the perceived threats by users and administrators who, anecdotal evidence suggests, typically worry about such issues as call interception and eavesdropping.

Figure 7 shows the breakdown based on source of vulnerability. The overwhelming majority of reported problems arise from implementation issues, which should not be surprising given the nature of bug disclosure. Problems arising from configuration represented 11% of the total space, including such items as privileged services left on and default username/passwords. However, note that the true picture (i.e., what actually happens with deployed systems) is probably different in that configuration problems are

most likely undercounted: such problems are often site-specific and are not reported to bug-disclosure databases when discovered. On the other hand, implementation and protocol problems are prime candidates for disclosure. What is surprising is the presence of protocol vulnerabilities; one would expect that such problems would have been discovered and issued during protocol development, specification, and standardization. Their mere existence indicates high protocol complexity.

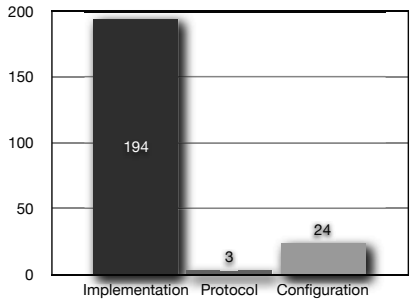


FIGURE 7: VULNERABILITY BREAKDOWN BASED ON SOURCE (IMPLEMENTATION, CONFIGURATION, PROTOCOL)

FIGURE 8: VULNERABILITIES PER PLATFORM

Finally, Figure 8 shows the breakdown of vulnerabilities based on the affected type of platform. In a few cases, typically when a bug was found in a software library, the vulnerability could be exploited in both clients and servers. Otherwise, vulnerabilities are equally distributed between the two primary types of VoIP platform. Although not shown here, the same holds when looking at specific classes of vulnerabilities (e.g., DoS).

Conclusions

The large majority of disclosed threats focused on DoS attacks based on implementation issues. While fault-tolerance techniques (such as replication) can be applied in the case of servers, it is less clear how to provide similar levels of protection at acceptable cost and usability to end-user devices. Unfortunately, the ease with which mass DoS attacks can be launched over the network against client devices means that they represent an attractive venue for attackers to achieve the same impact.

Code injection attacks in their various forms remain a problem, despite considerable progress in creating defenses. We need to do a better job at deploying and using these defenses where possible and in devising new techniques suitable for the constrained environments that some vulnerable VoIP devices represent.

Weak default configurations also present a big problem, as they do across a large class of consumer and enterprise products and software. The situation is likely to be much worse in the real world, considering the complexity of

securely configuring a system with as many components as VoIP. Vendors must make an effort to provide secure-by-default configurations, and to educate users how best to protect their systems. Administrators are in need of tools to analyze their existing configurations for vulnerabilities. While some tools dynamically test network components (e.g., firewalls), we need tools that work higher in the protocol and application stack. Furthermore, we need ways of validating configurations across multiple components and protocols.

Finally, there is simply no excuse for protocol-level vulnerabilities. While there exist techniques for analyzing and verifying security protocols, they do not seem to cope well with complexity. Aside from using such tools and continuing their development, protocol designers and standardization committees must consider the impact of their decisions on system implementers, i.e., whether a feature or aspect of the protocol is likely to be misunderstood and/or mis-implemented. Unfortunately, while simpler protocols are desirable, they seem incompatible with the trends we have observed in standardization bodies.

Network administrators can and must be proactive. Concrete steps to protect VoIP systems include but are not limited to:

- Stay current with firmware updates and security news about the devices deployed in your network. It is easy to overlook the fact that a VoIP hardphone may require a software update, just as servers and desktops do.
- Change the default/administrator authentication credentials in all devices and services! Make sure you cover all services running in each device (e.g., the Web-based management interface).
- Use any of a number of free or commercial SIP fuzzing tools, especially before initial roll-out of VoIP services, and after each firmware/software update. Do this both against end devices (hardphones and softphones) and servers.
- Make it your business to know what services each VoIP device is running. Do not trust the vendor to have produced a locked-down system—several misconfiguration-induced vulnerabilities came from leftover services running on the device. A simple port-scan will typically reveal such problems. If a service is not absolutely necessary, stop it; if that is not possible, block it at the firewall and complain to the vendor.
- Take steps to harden your VoIP servers. This may involve using obscure OS security features, or a different OS altogether. If possible, consider using a redundant server configuration with different operating systems running the same application server. (Using different application servers would be ideal, but impractical over the long run due to incompatibilities and configuration drift.) If you use server redundancy, make sure to test it periodically! There's nothing worse than discovering your secondary server is misconfigured while your primary server is compromised. (In reality, there are many things worse than this. Nonetheless, it is a very unpleasant situation.)
- Harden/protect the infrastructure on which your VoIP services rely. Specific services that merit attention include DNS, DHCP, and TFTP. This involves many of the steps mentioned above, for each of these services.
- Limit arbitrary access to VoIP devices. While this seems at odds with the basic premise of VoIP, it is possible to channel communications through media gateways. While this risks introducing some scalability problems, it also offers the opportunity to monitor traffic for abnormal behavior and to block some types of attacks against end devices. Along the same lines, you may also want to consider putting all your VoIP traffic into a different

VLAN, especially if VLAN port configurations can be frozen (admittedly a difficult proposition in many environments).

- When possible, enable TLS authentication and encryption for SIP signaling and use SRTP for media encryption. While the use of SRTP in particular is not widespread, the benefits appear to outweigh the (performance-related) drawbacks.

While there is no guarantee that the above steps will prevent a compromise (or that they are complete), they would have helped against most of the disclosed vulnerabilities we examined. The bottom line is that, while the situation with respect to VoIP security is currently bleak, there are steps you can take to protect your infrastructure today.

ACKNOWLEDGMENTS

The US National Science Foundation and the French National Research Agency supported this work under Grant CNS-09-14312 and Contract ANR-08-VERS-017, respectively.

REFERENCES

[1] A.D. Keromytis, "Voice over IP: Risks, Threats and Vulnerabilities," Cyber Infrastructure Protection (CIP) Conference, June 2009: http://www.cs.columbia.edu/~angelos/Papers/2009/cip.pdf.

[2] B. Krebs, "Security Fix: Default Passwords Led to $55 Million in Bogus Phone Charges," June 2009": http://voices.washingtonpost.com/securityfix/2009/06/default_passwords_led_to_55_mi.html.

[3] A.D. Keromytis, "Reference to Vulnerabilities Surveyed," February 2010: http://www.usenix.org/publications/login/2010-02/pdfs/keromytistables.html.

[4] R. State, O. Festor, H. Abdelanur, V. Pascual, J. Kuthan, R. Coeffic, J. Janak, and J. Floroiu, "SIP Digest Authentication Relay Attack: draft-state-sip-relay-attack-00," March 2009: http://tools.ietf.org/html/draft-state-sip-relay-attack-00.

[5] John Oates, *The Register*, "Two Charged with VoIP Fraud," June 2006: http://www.theregister.co.uk/2006/06/08/voip_fraudsters_nabbed/.

[6] Dan Goodin, *The Register*, "Fugitive VOIP Hacker Cuffed in Mexico," February 2009: http://www.theregister.co.uk/2009/02/11/fugitive_voip_hacker_arrested/.

DAVID CHOFFNES AND
FABIÁN E. BUSTAMANTE

taming the Torrent

David R. Choffnes will receive his PhD this year from the Department of Electrical Engineering and Computer Science at Northwestern University. Not surprisingly, his research interests align with those of his advisor (below). In particular, he currently focuses on designing, building, and evaluating large-scale, practical distributed systems.

drchoffnes@eecs.northwestern.edu

Fabián E. Bustamante is an associate professor in the Department of Electrical Engineering and Computer Science at Northwestern University. His research interests include operating systems and distributed systems, and networking in both wired and wireless settings.

fabianb@eecs.northwestern.edu

OVER THE PAST DECADE, THE PEER-TO-peer (P2P) model for building distributed systems has enjoyed incredible success and popularity, forming the basis for a wide variety of important Internet applications such as file sharing, voice-over-IP (VoIP), and video streaming. This success has not been universally welcomed. Internet Service Providers (ISPs) and P2P systems, for example, have developed a complicated relationship that has been the focus of much media attention. While P2P bandwidth demands have yielded significant revenues for ISPs as users upgrade to broadband for improved P2P performance, P2P systems are one of their greatest and costliest traffic engineering challenges, because peers establish connections largely independent of the Internet routing. Ono [4] is an extension to a popular BitTorrent client that biases P2P connections to avoid much of these costs without sacrificing, and potentially improving, BitTorrent performance.

Most P2P systems rely on application-level routing through an overlay topology built on top of the Internet. Peers in such overlays are typically connected in a manner oblivious to the underlying network topology and routing. These random connections can result in nonsensical outcomes where a peer—let's say, in the authors' own campus network in the Chicago suburbs—downloads content from a host on the West Coast even if the content is available from a much closer one in the Chicago area. This can not only lead to suboptimal performance for P2P users, but can also incur significantly larger ISP costs resulting from the increased interdomain (cross-ISP) traffic.

The situation has driven ISPs to the unfavorable solution of interfering with users' P2P traffic—shaping, blocking, or otherwise limiting it—all with questionable effectiveness. For instance, when early P2P systems ran over a fixed range of ports (e.g., 6881–89 for BitTorrent), ISPs attempted to shape traffic directed toward those ports. In response, P2P systems have switched to nonstandard ports, often selected at random. More advanced ISP strategies, such as deep packet inspection to identify and shape P2P-specific flows, have resulted in P2P clients that encrypt their connections. Recently, some

ISPs have attempted to reduce P2P traffic by placing caches at ISP network edges or by using network appliances for spoofing TCP RST messages, which trick clients into closing connections to remote peers. The legality of these approaches is questionable. By caching content, ISPs may become participants in illegal distribution of copyrighted material, while interfering with P2P flows in a non-transparent way not only may break the law but also can lead to significant backlash. Given this context, it is clear that any general and sustainable solution requires P2P users to buy in.

One possible approach would be to enable some form of cooperation between P2P users and ISPs. ISPs could offer an oracle service [2] that P2P users rely on for selecting among candidate neighbor peers, thus allowing P2P systems to satisfy their own goals while providing ISPs with a mechanism to manage their traffic [7]. However, we have seen that P2P users and ISPs historically have little reason to trust each other. Beyond this, supporting such an oracle requires every participating ISP to deploy and maintain infrastructure that participates in P2P protocols.

To drive peer selection, Ono adopts a new approach based on recycled network views gathered at low cost from content distribution networks (CDNs) without additional path monitoring or probing. Biased peer selection addresses a key network management issue for ISPs, obviating controversial practices such as traffic shaping or blocking. By relying on third-party CDNs to guide peer selection, Ono ensures well-informed recommendations (thus facilitating and encouraging adoption) while bypassing the potential trust issues of direct cooperation between P2P users and ISPs.

We have shown that peers selected based on this information are along high-quality paths to each other, offering the necessary performance incentive for large-scale adoption. At the end of July 2009, Ono had been installed more than 630,000 times by users in 200 countries.

BitTorrent Basics

Before describing Ono, we discuss how BitTorrent peers select neighbors for transferring content. A more complete description of the BitTorrent protocol can be found in the article by Piatek et al. [5].

BitTorrent distributes a file by splitting it into fixed-size blocks, called pieces, that are exchanged among the set of peers participating in a swarm. After receiving any full piece, a peer can upload it to other directly connected peers in the same swarm.

To locate other peers sharing the same content, peers contact a tracker, implemented as a centralized or distributed service, that returns a random subset of available peers. By default, each peer initially establishes a number of random connections from the subset returned by the tracker. As the transfer progresses, downloading peers drop low-throughput connections and replace them with new random ones.

CDNs as Oracles

Ono biases peer connections to reduce cross-ISP traffic without negatively impacting, but indeed potentially improving, system performance. Ono's peer recommendations are driven by recycled network views gathered at low cost from CDNs.

CDNs such as Akamai or Limelight attempt to improve Web performance by delivering content to end users from multiple, geographically dispersed

servers located at the edge of the network. Content providers (e.g., CNN.com and ABC.com) contract with CDNs to host and distribute their content. When a client attempts to download content hosted by a CDN, the client performs a DNS request for the associated URL. By redirecting the DNS lookup into the CDN DNS infrastructure, the CDN can dynamically respond with IP addresses of replica servers that will provide good performance (see Figure 1).

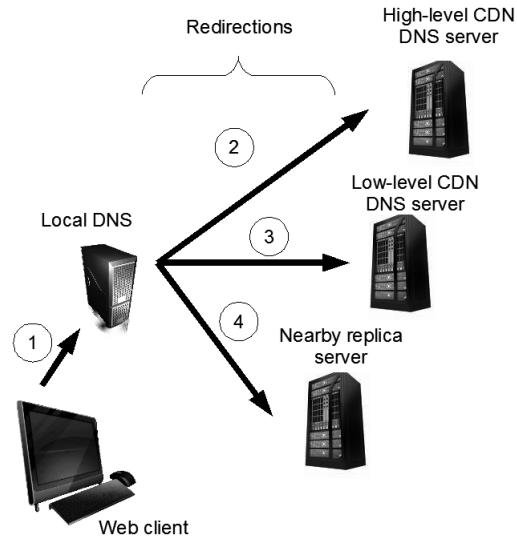


FIGURE 1: DIAGRAM OF CDN DNS REDIRECTION. THE WEB CLIENT PERFORMS A DNS LOOKUP (1), WHICH IS REDIRECTED TO A HIERARCHY OF CDN DNS SERVERS (2 AND 3) THAT EVENTUALLY RETURN THE IP ADDRESSES OF REPLICA SERVERS (4).

CDNs have all the attributes of an ideal oracle: they are pervasive, with an essentially global presence, have a comprehensive and dynamic view of network conditions [6], and, through DNS redirections, offer a scalable approach to access these views. Ono relies on CDNs as oracles, building on the hypothesis that when different hosts are sent to a similar set of replica servers, they are likely near the corresponding replica servers and by transition, also close to each other, possibly within the same ISP.

To use redirection information, we first need to encode it in a way that allows a client to compare the redirections that each peer witnesses. To this end, we represent peer-observed DNS redirection behavior using a map of ratios, where each ratio represents the frequency with which the peer has been directed toward the corresponding replica server during the past time window. Specifically, if peer P_a is redirected toward replica server r_1 75% of the time and toward replica server r_2 25% of the time, then the corresponding ratio map is:

$$\mu_a = \{r_1 \rightarrow 0.75, r_2 \rightarrow 0.25\}$$

More generally, the ratio map for a peer P_a is a set of (replica-server, ratio) tuples represented as

$$\mu_a = \{(r_k, f_k), (r_l, f_l), \ldots, (r_m, f_m)\}$$

Note that each peer's ratio map contains only as many entries as replica servers seen by that peer (in practice, the average number of entries is 1.6 and the maximum is 31), and that the sum of the f_i's in any given ratio map equals one.

For biased peer selection, if two peers have the same ratio map values, then the path between them should cross a small number of networks (possibly

zero). Similarly, if two peers have completely different redirection behavior, it is likely that the path between them crosses a relatively large number of networks. For cases in between, we use the cosine similarity metric to produce a continuum of similarity values between 0 (no similarity) and 1 (identical). This metric is analogous to taking the dot product of two vectors and normalizing the result.

In the next section, we show how Ono collects and distributes CDN redirection information in a scalable way and how it exploits this information to reduce cross-ISP traffic in BitTorrent.

Biasing P2P Connections with Ono

To perform biased peer selection, Ono must maintain a ratio map for each CDN-associated URL used for DNS lookups (e.g., a1921.g.akamai.net). As described in [6], using different CDNs and even different URLs for the same CDN can lead to different results in terms of redirection behavior. For the purpose of reducing cross-ISP traffic, these different URLs provide different granularity for proximity information.

The Ono plugin uses a built-in DNS client to perform periodic DNS lookups on popular URLs, which it uses to maintain ratio maps. Ono's overhead is extremely small: determining each peer's proximity requires network operations that scale *independently of the number of peers in the network*. To determine the cosine similarity value for a peer, Ono must be able to compare its ratio maps with those of other peers. The latter information can be obtained in a number of ways: (1) through direct exchange between peers, (2) from trackers, and (3) from some form of distributed storage. Ono currently supports all of these options. With direct exchange, when two peers running the Ono plugin perform their connection handshake, the peers swap ratio maps directly. Our implementation for the opentracker software allows Ono peers to report ratio map information to a tracker and receive a set of nearby peers in response. The last option uses the Vuze built-in distributed hash table (DHT) to distribute ratio maps, but due to its relatively large latencies this technique is currently disabled.

When Ono determines that another peer has similar redirection behavior, it attempts to bias traffic toward that peer by ensuring there is always a connection to it, which minimizes the time that the peer is choked (i.e., waiting for data transfer to start). To maintain the appealing robustness that comes from the diversity of peers provided by BitTorrent's random selection, Ono biases traffic to only a fraction of the total connections established for a particular torrent (currently, at most, half).

Ono is written in Java and designed as a plugin (i.e., extension) for compatibility with the Vuze BitTorrent client. We chose Vuze because it is one of the most popular BitTorrent clients, provides cross-platform compatibility, and features a powerful API for dynamically adding new functionality via plugins. Our plugin contains approximately 12,000 method lines of code, 3,500 of which are for the GUI and 3,000 for data collection and reporting (and thus not essential for Ono functionality). It is publicly available with source code at http://azureus.sourceforge.net/plugin_list.php, or it can be downloaded and installed from inside the Vuze client.

Experience in the Wild

To evaluate the effectiveness of Ono, we instrumented the plugin to measure and report performance information for participating users. As our results

show, Ono indeed reduces cross-ISP traffic without reducing transfer performance—in fact, Ono-selected peers on average *improve* performance compared to default peer selection. Thus, Ono not only reduces cross-ISP traffic but also provides an incentive for users to engage in ISP-friendly behavior. This incentive is the main reason behind Ono's large-scale adoption.

In our evaluation, we use traceroute measurements from Ono subscribers to connected peers to infer whether Ono-selected peers reduce cross-ISP traffic compared to ones selected by the standard (random selection) BitTorrent protocol. Traceroutes provide router-level views of paths between hosts. Since an ISP may contain many routers, we need to analyze the traceroute measurements using metrics that more closely correspond to ISP hops. Because the Internet is divided into separate administration domains in the form of autonomous systems (ASes), we expect that AS-level path information will provide better insight regarding cross-ISP links. Although there is no one-to-one relationship between ASes and ISPs, the number of AS-hops along a path gives us an upper-bound estimate on the number of cross-ISP hops. We generate AS-level path information from our traceroute data using the AS mappings provided by the Team Cymru group (http://www.cymru.com/).

For brevity, we present cumulative results when using the URL for LeMonde.com, the online version of a popular French newspaper. A comparison among all CDN-associated URLs can be found in [4].

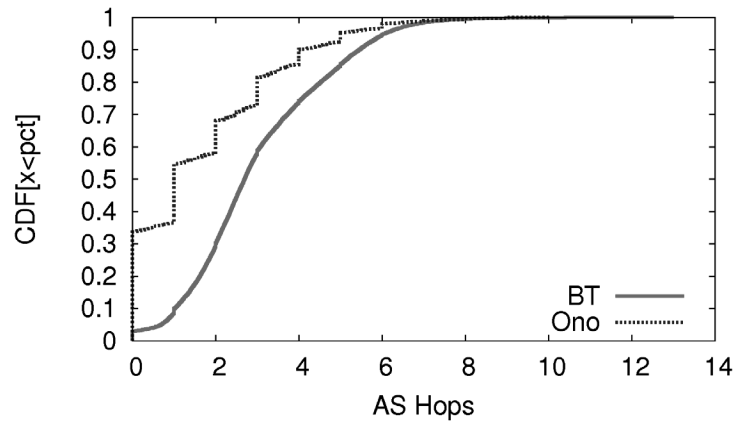


FIGURE 2: CDF OF AVERAGE NUMBER OF AS HOPS TO REACH ONO-RECOMMENDED PEERS AND THOSE FROM UNBIASED BITTORRENT. OVER 33% OF PATHS TO ONO-RECOMMENDED PEERS DO NOT LEAVE THE AS OF ORIGIN, AND THE MEDIAN NUMBER OF ASES CROSSED BY PATHS TO ONO-RECOMMENDED PEERS IS HALF OF THOSE PICKED AT RANDOM BY BITTORRENT.

Figure 2 presents a CDF of the number of AS hops taken along paths between Ono clients and their peers. Each value on the curve represents the average number of hops for all peers, either located by Ono or picked at random by BitTorrent, seen by a particular Ono client during a six-hour interval. The most striking property is that over 33% of the paths found by Ono do not leave the AS of origin. Further, the median number of AS hops along a path found by Ono is one, whereas this is the case for less than 10% of the paths found by BitTorrent at random. Thus, Ono significantly reduces the overall amount of cross-ISP traffic, thereby promoting "good Internet citizen" behavior that benefits not only the origin ISP but also nearby networks.

Not only does Ono reduce cross-ISP traffic, but it does so while locating high-quality paths to biased peers. For instance, the median latency to Ono-

recommended peers is 6 ms, whereas the same for peers picked at random is 530 ms—two orders of magnitude difference. We also saw improvements for traceroute-inferred loss: on average, paths to Ono-recommended peers exhibit nearly 31% lower loss rates and their median loss rate is 0, whereas the median loss rate for paths to unbiased peers is 2.1%.

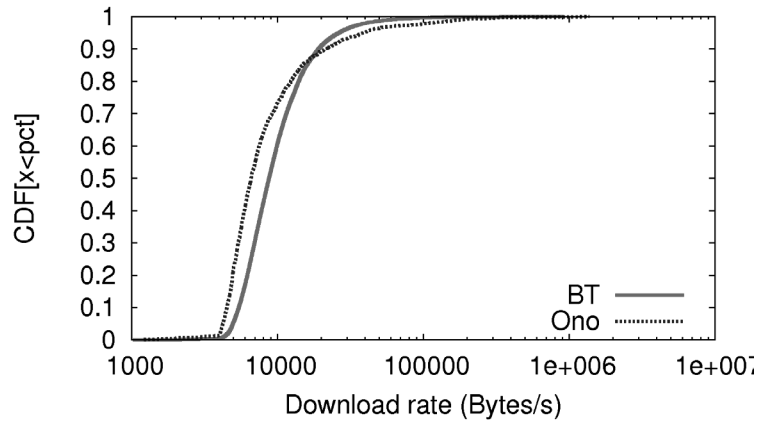


FIGURE 3: CDFS OF AVERAGE RATES FROM ONO-RECOMMENDED PEERS AND THOSE FROM UNBIASED BITTORRENT PEERS, ON A SEMILOG SCALE. THE AVERAGE DOWNLOAD RATE FOR ONO IS 31% BETTER THAN UNBIASED BITTORRENT AND THE DIFFERENCE IN MEDIAN DOWNLOAD RATES IS ONLY 2 KB/S.

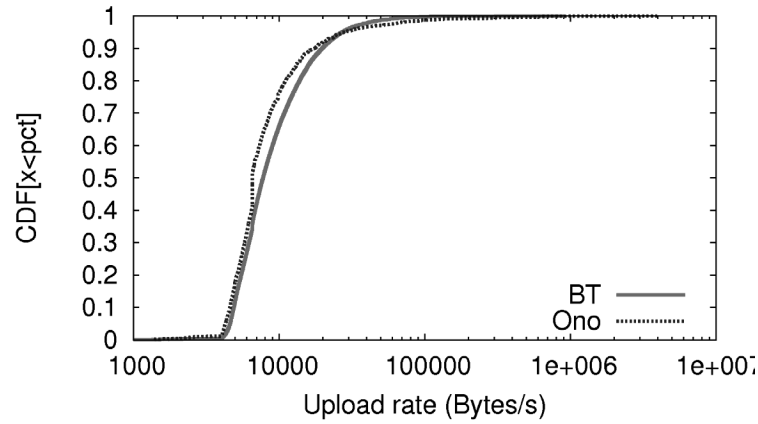


FIGURE 4: CDFS OF AVERAGE UPLOAD RATES FROM ONO-RECOMMENDED PEERS AND THOSE FROM UNBIASED BITTORRENT PEERS, ON A SEMILOG SCALE. THE AVERAGE UPLOAD RATE FOR ONO IS 42% BETTER THAN UNBIASED BITTORRENT AND THE DIFFERENCE IN MEDIAN RATES IS ONLY 1 KB/S.

In the end, however, it is safe to assume that most P2P users care how these paths impact *throughput* for their BitTorrent transfers. To evaluate this property, Figures 3 and 4 present CDFs of the average download and upload rates for biased and unbiased connections on a semilog scale. For this and the following figures, we use all transfer rate samples where the connection was able to sustain a 4 KB/s transfer rate at least once. Connections with lower rates tend to be dropped and do not contribute meaningfully to this analysis.

We begin by observing that peers recommended by Ono provide significantly higher peak download rates than those picked at random. In fact, this distribution features a heavy tail—although the median download rate from Ono-recommended peers is slightly lower than those picked at random

by BitTorrent, the *average* download rate for Ono is 31% higher than that of unbiased BitTorrent. This seems to indicate that the relatively high quality of paths recommended by Ono also result in higher peak throughput when there is sufficient available bandwidth.

Despite the fact that Ono reduces cross-ISP traffic by proactively reconnecting to nearby peers regardless of available bandwidth, the difference between median transfer rates for Ono and unbiased BitTorrent is only 2 KB/s. Note, however, that even when Ono-recommended peers do not provide higher median throughput than those picked at random, our approach does not noticeably affect completion time for downloads. This holds because Ono-recommended peers are only a fraction of the entire set of peers connected to each client and BitTorrent generally saturates a peer's available bandwidth with the remaining connections.

That said, we expected higher median performance for Ono-recommended peers, given the low latencies and packet loss along paths to them. Based on the relatively low average per-connection transfer rates in both curves (around 10 KB/s), we believe that performance gains for Ono-recommended peers are most likely limited because BitTorrent peers are generally overloaded. By splitting each peer's bandwidth over a large number of peers, the BitTorrent system achieves high *global* transfer rates while generally providing relatively low individual transfer rates to *each connection*. In this case, the bottleneck for BitTorrent clients is the access link to the ISP rather than the cross-ISP link [1]. While this situation occurs frequently, it is not universal and, as we now show, depends on ISPs' bandwidth-allocation policies.

Figure 5 plots a CDF of download rates from Ono clients located in the RDSNET ISP (http://www.rdslink.ro) in Romania. At the time of this study, the ISP offered 50 Mb/s unrestricted transfer rates over fiber for *in-network traffic* (i.e., traffic inside the ISP) and 4 Mb/s to connections outside the ISP, effectively pushing the bandwidth bottleneck to the edge of the network. The figure clearly shows that Ono thrives in this environment, significantly improving the download rates of Ono-recommended peers in comparison with that of randomly selected nodes. In particular, we see the average download rate for Ono-recommended peers improves by 207%, and their median download rate is higher by 883%.

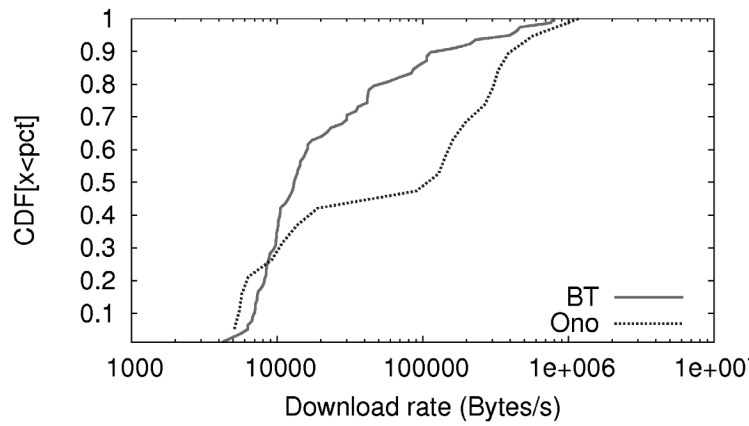


FIGURE 5: CDF OF AVERAGE DOWNLOAD RATE FOR AN ISP THAT PROVIDES HIGHER BANDWIDTH TO IN-NETWORK TRAFFIC. ONO THRIVES IN THIS ENVIRONMENT.

To compare against an ISP with uniform (and relatively low) bandwidth constraints, Figure 6 shows a CDF of download performance for Easynet (http://www.easynetconnect.co.uk), an ISP located in the UK. This ISP offers 4 or 8

Mb/s downstream with only 768 Kb/s upstream. As the figure clearly shows, any performance gains that could be attained by Ono in terms of transfer rates are negated by the suboptimal bandwidth allocation. Further, we believe that the higher median performance seen by default BitTorrent peer selection comes from the ability to find peers in other networks that are less constrained by upload bandwidth allocation and therefore provide higher throughput.

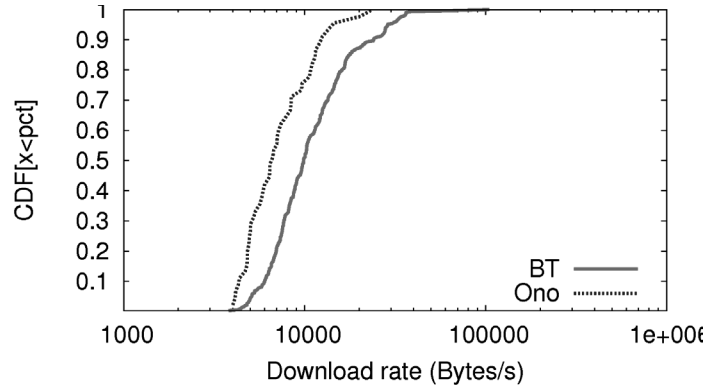


FIGURE 6: CDF OF AVERAGE DOWNLOAD RATE FOR AN ISP WITH AN ASYMMETRIC BANDWIDTH ALLOCATION POLICY, WHICH SIGNIFICANTLY CONSTRAINS ONO PERFORMANCE.

Finally, we demonstrate that the bandwidth allocation model in the RDSNET ISP, when coupled with Ono, provides a mutually beneficial environment in which BitTorrent users see higher transfer performance while reducing the cost for ISPs in terms of cross-ISP traffic. Figure 7 illustrates this using a bar graph, with the x-axis representing the number of AS hops along paths between peers and the y-axis representing the average of the download rates between these peers. It is clear that RDSNET, which offers higher transfer rates inside the ISP, allows users to obtain significant performance gains by reducing cross-ISP traffic. On the other hand, Easynet, which does not offer different transfer rates for in-network traffic, exhibits negligible performance differences for connections with different AS-path lengths. Consequently, performance from Ono-recommended peers will not be significantly different from those picked at random.

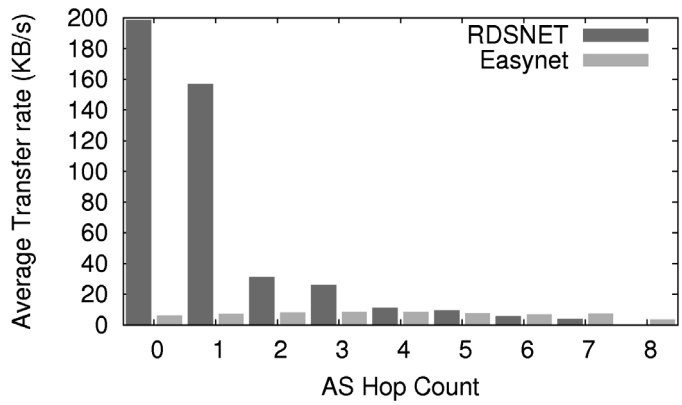


FIGURE 7: BAR PLOT RELATING AS HOP COUNT TO TRANSFER PERFORMANCE FOR ISPS WITH DIFFERENT BANDWIDTH ALLOCATION POLICIES. RDSNET GIVES BETTER TRANSFER RATES TO IN-NETWORK TRAFFIC AND EASYNET DOES NOT. IN THE FORMER CASE, ONO LEADS TO SIGNIFICANT PERFORMANCE GAINS.

These results make the case for a new ISP-based approach to the problem of taming BitTorrent that is compatible with biased peer selection as implemented in this work. Rather than blocking BitTorrent flows, ISPs should change their bandwidth allocations so that it is more favorable to connect to peers inside the ISP than to those outside. Assuming that the former traffic costs are much smaller than cross-ISP traffic costs, this approach should lead to substantial savings for ISPs, higher subscriber satisfaction, and fewer legal issues.

Conclusion

In this article, we briefly described how Ono reuses CDN redirection information to bias P2P connections, thus reducing ISPs' costs associated with P2P traffic without sacrificing system performance. While our current implementation is targeted specifically at the BitTorrent P2P protocol, the approach is being adopted for other services, including video streaming in the Goalbit project and P2P file transfer in Gnutella (in the Limewire client).

The Ono project is one piece in a broader research agenda that explores the potential for strategic reuse and recycling of network information made available by long-running services for building large-scale distributed systems. In that vein, we have reused CDN redirections to drive a high-performance detouring service [3] and reused passively gathered P2P performance information to automatically detect network problems. For more information, visit our Web site at http://aqualab.cs.northwestern.edu.

ACKNOWLEDGMENTS

We would like to thank Rik Farrow for his interest in our work. We are also grateful to Paul Gardner for his assistance in deploying Ono in Vuze. This work was supported by NSF CAREER Award Grant No. CNS-0644062.

REFERENCES

[1] A. Akella, S. Seshan, and A. Shaikh, "An Empirical Evaluation of Wide-Area Internet Bottlenecks," *Proceedings of the Internet Measurement Conference (IMC)* (2003).

[2] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang, "Improving Traffic Locality in BitTorrent via Biased Neighbor Selection," *Proceedings of the Int'l Conference on Distributed Computing Systems (ICDCS)* (2006).

[3] D. Choffnes and F. Bustamante, "On the Effectiveness of Measurement Reuse for Performance-Based Detouring," *Proceedings of IEEE INFOCOM* (April 2009).

[4] D.R. Choffnes and F.E. Bustamante, "Taming the Torrent: A Practical Approach to Reducing Cross-ISP Traffic in P2P Systems," *Proceedings of ACM SIGCOMM* (2008).

[5] M. Piatek, T. Isdal, T. Andrewson, A. Krishnamurthy, and A. Venkataramani, "Building BitTyrant, a (More) Strategic BitTorrent Client," *;login: 32*, 4 (August 2007).

[6] A.-J. Su, D.R. Choffnes, A. Kuzmanovic, and F.E. Bustamante, "Drafting behind Akamai: Travelocity-Based Detouring," *Proceedings of ACM SIGCOMM* (2006).

[7] H. Xie, R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: Provider Portal for (P2P) Applications," *Proceedings of ACM SIGCOMM* (2008).

DAVID PISCITELLO

SSR: securing the Internet's identifier systems

Dave Piscitello is a Senior Security Technologist for ICANN. A 30-year Internet veteran, Dave currently serves on ICANN's Security and Stability Advisory Committee and the Internet Policy Committee of the Anti-Phishing Working Group (APWG).

dave@corecom.com

NO SINGLE ENTITY IS RESPONSIBLE for the administration and oversight of all aspects of the Internet. One entity, Internet Corporation for Assigned Names and Numbers (ICANN) *is* tasked with the responsibility of coordinating the Internet's unique identifier systems and with participating in the global operation of the Domain Name System (DNS). Part of this responsibility involves ensuring the security and stability of these identifier systems. This responsibility is not only a key element of ICANN's mission statement but an obligation under the September 30, 2009 Affirmation of Commitments by the United States Department of Commerce and ICANN [1]. How ICANN intends to meet this obligation is described in its Security, Stability and Resiliency Plan (SSR).

In this article, I highlight the critical elements of that plan, scoping ICANN's remit and describing the activities ICANN will perform directly, as well as activities where ICANN will be one of potentially many collaborative players.

Who Runs—and Secures—the Internet?

The simple answer is "no one and everyone." Originally, the term Internet derived from the concept of an "an *inter*connection of *net*works." Today, the term Internet more broadly applies to the combination of technology infrastructure and the ecosystem of communities that create, make use of, and sustain this critical infrastructure. Internet communities—governments, public and private organizations, and individuals, also called *stakeholders*—cooperate to develop policies and business practices necessary to sustain a productive ecosystem, and they must do so at the same pace as Internet technology evolution.

These stakeholders must also respond to the persistent and growing sets of threats posed by criminal, combatant (terroristic), and miscreant actors. This set of "bad actors" exploits the transportation routes the Internet provides to perpetrate criminal and malicious activities. They also exploit the Internet's unique and critical sets of identifiers, using *domain names* and *Internet addresses* to facilitate unwanted activities ranging from identity theft, fraud,

and disruptions or denials of service, to illegal sales of controlled substances and pharmaceuticals and human trafficking.

ICANN's Remit

ICANN facilitates multi-stakeholder, consensus-based policy and program development to ensure that identifiers such as domain names and IP addresses are allocated fairly and equitably. These policies and processes influence how the DNS operates, how domain names may be composed and registered, and how addresses are allocated. These policies are typically reflected in contracts and agreements established between ICANN and operators around the world (governmental and private). Contrary to numerous popular perceptions, ICANN is not a regulatory, legislative, or law enforcement entity. ICANN is not involved in activities related to dealing with cyber-espionage, terrorism, or warfare and does not take a role in what constitutes illegal content on the Internet. Historically, and now formally as stated in a Security, Stability and Resiliency plan [2], ICANN has taken steps to ensure that the identifiers it coordinates are used in ways that do not threaten the Internet ecosystem.

ICANN's Direct Roles

ICANN has several direct roles in ensuring the Internet's security, stability, and resiliency. As the performer of the Internet Assigned Numbers Authority (IANA) function, ICANN administers and ensures the integrity of the very large set of assigned numbers that are critical to the correct operation of the Internet infrastructure. This set not only includes the familiar domain names and IP addresses, but dozens of other databases (registries) of system and protocol numbers as well. The IANA operation is also responsible for the coordination and publication of the authoritative root zone for the DNS. This activity involves the coordination and verification of zone information for 280 delegated top-level domains (TLDs) and 13 root name servers. IANA's performance objective for this activity is effectively "zero tolerance for misconfiguration error." ICANN will also shortly work in cooperation with its root zone management partners (US NTIA and VeriSign) to cryptographically sign the root zone, an action that is expected to accelerate the adoption of DNS Security (DNSSEC). DNSSEC will add origin authentication, zone data integrity, and authenticated denial of existence services that are designed to defeat DNS cache poisoning and hijacking attacks.

ICANN also operates one of 13 root name servers for the DNS. In this role, ICANN maintains a fully redundant, highly secured, anycast-enabled root name server system ("L") at geographically diverse locations. Each operation is kept resilient from attacks and operationally available following industry best practices in design, capacity, and security planning.

Another of ICANN's direct roles evolves naturally from the experience and expertise gained from operating large-scale DNS facilities. ICANN staff assist in the development and delivery of training and security awareness programs to TLD operators, when invited. These activities help improve the overall security of domain name resolution and registration services offered by TLD operators.

ICANN's Collaborative Roles

The ICANN acronym is often associated with the community of stakeholders and participants with which it collaborates. This blurred distinction does

cause confusion, but it also reflects the practical realities all stakeholders, including ICANN, face: (a) strengthening the security, stability, and resiliency of the Internet can only be accomplished through collaboration on a global level; and (b) cooperation of law enforcement, private sector, DNS, and security communities is needed to successfully intervene or respond to security incidents where the DNS is attacked or exploited on a global scale. ICANN's commitment to playing this role, as well as the commitments of TLD operators worldwide, are best exemplified by the ongoing effort to contain the Conficker worm [3]. ICANN staff worked with over one hundred TLD operators to preemptively block thousands of domain registrations identified daily by security companies who had cracked the Conficker executable and in doing so had discovered the algorithm the worm writers had used to generate domain names for would-be botnet rendezvous points. ICANN is working with the community to apply the lessons learned from this collaborative effort to improve and define response systems that will prove to be agile and adaptive to attacks and criminal activities. A recent example of this is the Expedited Registry Security Request, a collaborative effort between ICANN and gTLD registries to develop a process for quick action in cases where gTLD registries inform ICANN of "a present or imminent security incident to their TLD and/or the DNS" [4].

ICANN's SSR plan calls for participation in and engagement with organizations that work to contain or eliminate criminal activities such as phishing, fraud, identity theft, and abuse of intellectual property. These elements of the plan are reflected in several critically important areas of development, most notably programs relating to the addition of new TLDs and internationalized domain names. As part of its development of guidelines for new TLD applicants, ICANN solicited input from the Anti-Phishing Working Group (APWG), Registry Internet Safety Group (RISG), BITS Fraud Reduction Program, American Banking Association, Financial Services Information Sharing and Analysis Center (FS-ISAC) and Financial Services Technology Consortium (FSTC) to help define what the community believes constitutes *malicious conduct* [5]. ICANN also tasked a group of experts to study the risks associated with adding DNSSEC, new TLDs, and IPv6 to the root zone of the DNS [6]. Lastly, ICANN gathered input from experts from the security and financial communities, as well as with its own Security and Stability Advisory Committee (SSAC), to develop a high security voluntary verification program for new TLD registries [7]. Combined, the results of these joint activities will allow ICANN's contractual compliance program to better ensure that contractual obligations are taken into account by ICANN's contracted parties.

Way Forward for ICANN

This article calls attention to a fraction of the commitments ICANN has made in its SSR plan. The plan is ambitious even if one only assumes ICANN will take the lead on all the initiatives in the plan. The way forward for ICANN is to lead where its remit demands it lead, and collaborate when the opportunity to meet objectives in the SSR through collaboration appears. The SSR plan formalizes ICANN's commitment to contribute in the realms of security, stability, and resiliency as a central part of ICANN's role in coordinating global identifiers in a multi-stakeholder environment.

REFERENCES

[1] ICANN, "The Affirmation of Commitments: What It Means": http://www.icann.org/en/announcements/announcement-30sep09-en.htm#affirmation.

[2] ICANN, "Plan for Enhanced Internet Security, Stability and Resiliency": http://www.icann.org/en/announcements/announcement-2-21may09-en.htm.

[3] Conficker Working Group: http://www.confickerworkinggroup.org/wiki/.

[4] ICANN, "Expedited Registry Security Request Process": http://www.icann.org/en/announcements/announcement-01oct09-en.htm.

[5] ICANN, "New gTLD Program Explanatory Memorandum: Mitigating Malicious Conduct": http://www.icann.org/en/topics/new-gtlds/mitigating-malicious-conduct-04oct09-en.pdf.

[6] ICANN, "Root Scaling Study": http://www.icann.org/en/committees/dns-root/root-scaling-study-report-31aug09-en.pdf.

[7] ICANN, "New gTLD Program Explanatory Memorandum: Model for a High-Security Zone Verification Program," draft version 1.0: http://www.icann.org/en/topics/new-gtlds/high-security-zone-verification-04oct09-en.pdf.

RUDI VAN DRUNEN

peculiarities of radio devices

Rudi van Drunen is a senior UNIX systems consultant with Competa IT B.V. in The Netherlands. He also has his own consulting company, Xlexit Technology, doing low-level hardware-oriented jobs.

rudi-usenix@xlexit.com

IN THIS ARTICLE I WILL DESCRIBE THE peculiarities of using Radio Frequency (RF) devices in your systems. Most prominently, the WiFi subsystem in your laptop or access point will be covered, but I will also be touching upon Bluetooth and 3G (Cellular) devices.

Radio Systems

All wireless systems depend on radio waves to transmit and receive data. Radio wave frequency and the way data is encoded in radio waves differ from technology to technology.

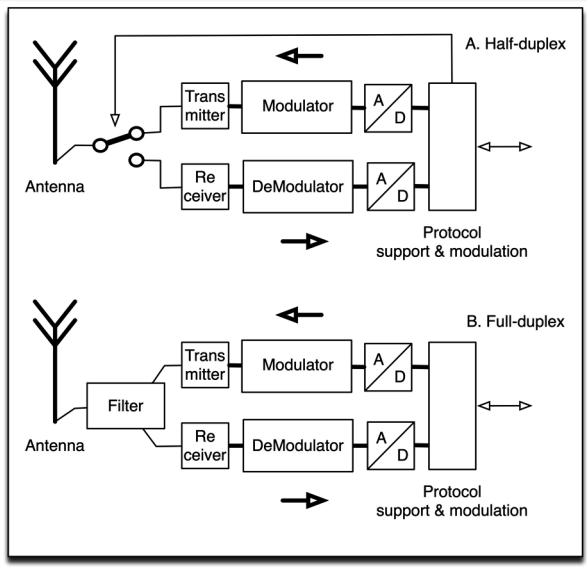


FIGURE 1: RADIO SYSTEMS: A: HALF DUPLEX; B: FULL DUPLEX

As illustrated in Figure 1, some radio systems switch between receiving and transmitting (half-duplex systems), and some radio systems are able to receive and transmit at the same time (full-duplex systems) using the same antenna. Full-duplex systems are often more complex, since designers need to prevent the high-power transmitter signal from getting into the sensitive receiver front end. The high-power signals, even if they are on another frequency, can make the receiver "deaf" or can even destroy the sensitive input amplifier electronics (we're talking about nano Volts sensitivity). Full-duplex systems usually get more throughput than half-duplex systems.

In the data encoding/decoding process we move from digital to analog technology; radio waves are all in the analog domain. Part of the encoding and

decoding is traditionally done in hardware using discrete electronics and integrated circuits by functional blocks known as modulators and demodulators, but as digital signal processors get more common and powerful, one can do this completely in software as well, resulting in software defined radio as shown in Figure 2. (More info on software defined radio can be found in [1].)

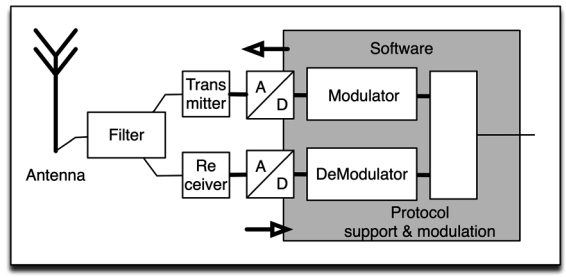


FIGURE 2: SOFTWARE DEFINED RADIO (SDR)

Almost all modern chipsets that provide WiFi, Bluetooth, or 3G capabilities nowadays are more or less software defined radios. By having the software doing the modulation/demodulation, it is much easier to adhere to newer protocols with the same hardware through simple firmware updates. Also calibration, which can be a tedious process in analog systems, can be skipped or replaced by simple software routines that calculate the different parameters to be used in the modulation and demodulation algorithms.

Wavelength

The radio waves, traveling at the speed of light (c = 300,000 Km/s through vacuum or air) have a specific wavelength (λ), which relates to the frequency the system operates at as f = c / λ. A typical WiFi system operating at 2.4 GHz has a wavelength of 12.5 cm (4.9 in). (See the first article in this series for more about wavelengths [2].)

Antennas

The last (or the first) part of a radio system is always the antenna. This device translates the electrical energy into electromagnetic (radio) waves, or the other way around. We can look at an antenna as a piece of wire ("the element") that can be brought into resonance, and often as a reflector to reflect and direct the radio waves. When the wire comes (partly) into resonance, at a particular frequency, it starts to efficiently transmit or receive radio waves. The frequency at which the wire element gets efficient (the resonance frequency) is dependent on its size (there is a relation to the wavelength), shape, and material. All different frequencies need different antennas.

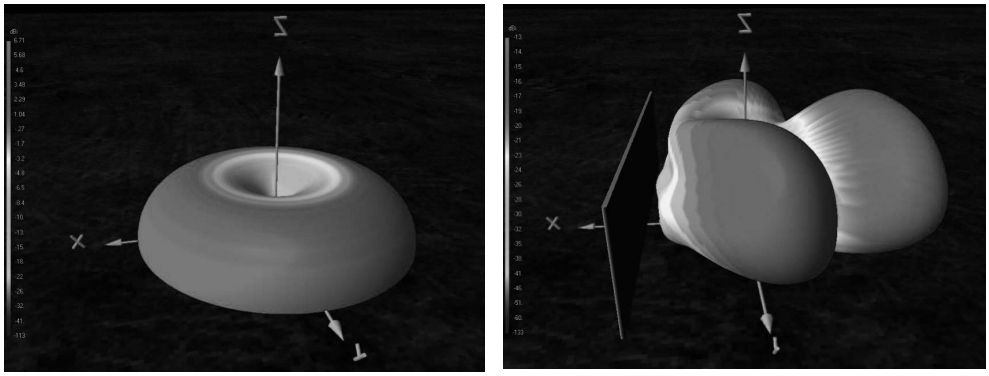
The frequency a system operates on differs from standard to standard. Table 1 shows the commonly used frequencies (some technologies can and are designed for operation in multiple frequency bands) in everyday data communication use.

Technology	1st frequency band	2nd frequency band
WiFi	2.4 GHz	5 GHz
Bluetooth	2.4 GHz	
WiMax	2.5 GHz	3.5 GHz
3G (UMTS)	1.8 GHz	1.9 GHz
ZigBee	900 MHz	2.4 GHz

TABLE 1: FREQUENCIES USED BY DIFFERENT TECHNOLOGIES

Next to the typical frequency an antenna is designed for there is the gain an antenna provides. The gain of an antenna works both in the transmitting mode and in the receiving mode and is highly dependent on the size and shape of the antenna.

It you have an omni-directional antenna, then the gain versus direction in the x-y (horizontal) plane is the same in all directions, whereas a directional antenna has a distinct direction in which the gain is at its largest. Figure 3 shows a simulation of two different antennas and their gain characteristics.



A: OMNI-DIRECTIONAL **B: DIRECTIONAL**

FIGURE 3: SIMULATED RADIATION PATTERNS

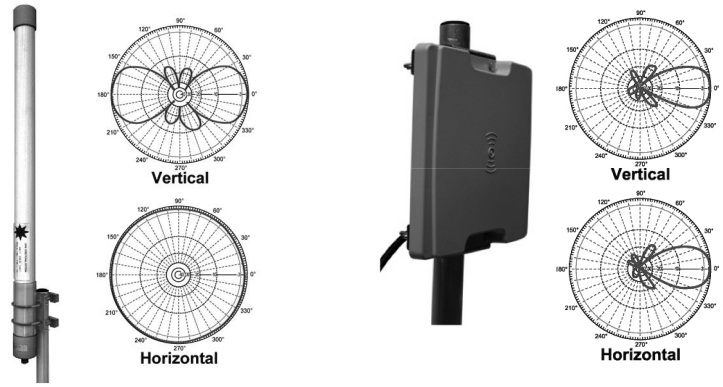
Omni-directional antennas are often used by access points where you want to cover as large an area as possible. You want the clients to be able to connect from all directions.

If for some reason you need a segment of the donut shape as coverage area, there are special sector antennas for this purpose. Mobile operators providing 3G service often use sector antennas with different opening angles (60, 90, or 120 degrees). By using different sectors they can shape the coverage area to their needs or use more transmitters in their base station, each serving a particular direction. The sector antenna is just a special case of the omni-directional antenna used to cover an area for clients.

Directional antennas, on the other hand, have a very small coverage angle: they often have opening angles of less than 10 degrees. Typical use of a directional antenna is in making point-to-point connections where the location of the other antenna is static and exactly known.

A special, mixed form of omni and directional antennas can be found in an active antenna that is able to direct a particular signal to a particular client within the 360 degrees of the omni antenna. This antenna requires pretty complex RF electronics to do phase shifts of the signals and uses multiple elements to transmit. This so-called phased-array antenna is often used by WiMax providers in their base-stations.

Figure 4 shows the actual real-life pictures of two antennas with their characteristics.



A: 8DBI OMNI-DIRECTIONAL B: 12 DBI DIRECTIONAL PANEL

FIGURE 4: PICTURES AND DIAGRAMS OF AN OMNI (A) ANTENNA AND A DIRECTIONAL (B) ANTENNA (SOURCE: WWW.GANDALF.NL)

The dB

In Figure 5 we introduce a new unit of measurement, the dB (decibel). The dB is not an absolute unit but denotes the relationship between two signals. When talking about power levels in W(atts), the ratio of two signals can be expressed in dB by calculating the 10-base Logarithm ($\log_{10}$) of the ratio between the actual level to a reference level, multiplied by 10 (since 10 dB (decibel) equals 1 B (Bell).

If you want to express a twofold increment in power, moving from 100 mW to 200 mW, for example, the dB increase equals

$$10\log_{10}\left(\frac{200mW}{100mW}\right) = 3dB$$

So if an antenna amplifies a signal 100 times in a particular direction, it has a 20 dB attenuation in that direction.

Often when talking about antennas we use the word "gain," which is synonymous with "attenuation." For antennas, we compare the gain of an antenna with a hypothetical isotropic radiator which radiates uniformly in all directions, and express the gain in dBi, whereas if we talk about (absolute) power levels we denote it by dBm. With dBm, we compare to 1 mW of power, so 100mW corresponds to

$$10\log_{10}\left(\frac{100mW}{1mW}\right) = 20dBm$$

Now let's put all this theory into action. We all know about WiFi systems and their peculiarities. To be able to make a point-to-point connection (building to building) using WiFi, we need to select equipment such as antennas, cabling, and access points/client devices in such a way that the connection is reliable but also within the limits defined by the FCC or its European Counterpart, ETSI, in maximum RF power output. Here we will show how to lay out such a system that will perform while operating within limits.

An Example

Having set up many point-to-point links in a wireless community network [3], I will show the theory in action on designing a 3 Km (a little less than 2 miles) link connecting two hops in the network. This example is shown in Figure 5. Here all components that add or reduce gain in the RF path are shown. The system we are designing uses 802.11b technology on an 11 Mb link rate. The connection is using the 2.4 GHz band.

802.11b technology has proven to be far better performing in outside environments than 802.11g due to the more robust RF modulation.

Making the calculations, we start off with calculating the loss that is imposed on the radio signal while traveling through free space:

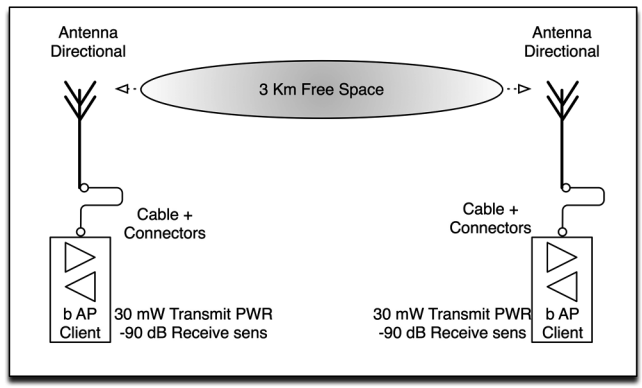


FIGURE 5. A BUILDING-TO-BUILDING OUTDOOR WIFI SETUP

The signal loss through air (free space loss) is defined as:

$$\text{Loss[dB]} = 92.5 + 20 \log f\,[\text{GHz}] + 20 \log d\,[\text{Km}]$$

with f the operating frequency in GHz and d the distance in kilometers.

In this case we have a loss of 92.5 + 7.6 + 9.5 = 109.6 dB.

We have access point and client devices that output 30 mW = 10 log10 (30 mW/1mW) = 14.8 dBm of RF energy and need a signal strength of –90 dBm for maintaining a good connection. All cabling and connectors in the system on each side have a loss of 1.2 dBi, recalling that we have cabling and connectors on both ends.

Now we can calculate the needed antenna gain. We need –90 dBm on the input of the device for a good connection. Allowing some margin, we define the minimum input signal as –85 dBm. This signal must come from the 14.8 dBm transmitter, going through a set of cabling twice, which gives a total of –2.4 dBi gain, and the free space, which gives –109.6 dB gain.

So the antennas must add another 14.8 – 109.6 – 2.4 – (-85) = 12 dBi, which can be accomplished using two antennas that have a gain in the desired direction of 6 dBi.

It is important to use the antennas in a symmetrical setup—that is, both sides having the same gain, as both devices send and receive; otherwise the power in front of the largest antenna will exceed maximum.

So a 6 dBi antenna for each side would do the job. To get some headroom, it is wise to select 7 dBi antennas, which gives you another 2 dBi extra margin. In a rainy climate, you might need some extra gain to cope with the extra loss you get from wet air. To check if we are within limits defined by the regulatory body, we have to calculate the output power of the antenna (the

field strength in front of the radiator), which in this case is 14.8 dBm (AP) + 7 dBi (antenna) – 2.5 dBi (cabling set) = 19.3 dBm = 85 mW of RF output power, which is allowed both in Europe and in the USA for the 802.11b band.

Another thing to take into account when designing a point-to-point WiFi link is that for the above free space loss calculation the RF beam needs to be unobstructed, so a line of sight is needed and the Fresnel zone must be clear. The Fresnel zone is an ellipsoid between the transmitter and receiver in which the radio waves live (see Figure 6).

The maximum radius of this Fresnel zone can be calculated by the following formula:

$$r[m] = \sqrt{\frac{d[Km]}{4 * f[GHz]}}$$

For the 3 Km link, the maximum radius of the ellipsoid would be about

$$17.32 * \sqrt{3/(4*2.4)} = 9.6 meters = 31\ ft.$$

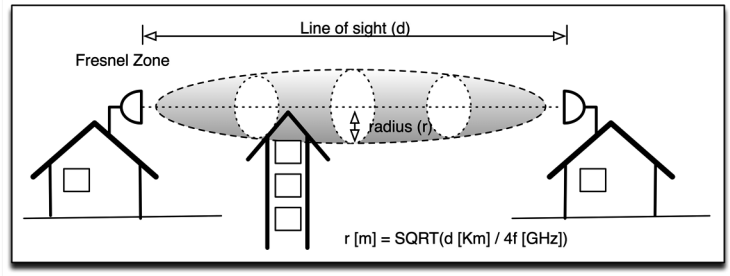


FIGURE 6: THE FRESNEL ZONE

Large objects covering (part of) the Fresnel zone will increase the free space loss, and therefore larger antenna gain might be needed. It is wise to have the Fresnel zone less than 10% occupied with trees or buildings. A larger percentage of occupation will seriously increase the free space loss.

If you make very long distance point-to-point connections, it is important not only to look at the Fresnel zone but also to take the curvature of the Earth into account. The Earth's curvature may result in partial coverage of the Fresnel zone when the antennas are not mounted high enough.

Range-Limiting Factors

There are a number of factors that limit the range of a wireless datacom system.

MULTIPATH

As with any waves, radio waves can be reflected by surfaces such as walls. These reflected waves, which are weaker than the direct wave, are received slightly after the primary signal (see Figure 7).

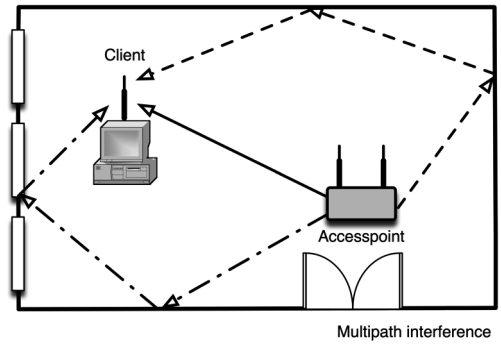


FIGURE 7: MULTIPATH INTERFERENCE

This causes interference and distortion of the resulting signal. This effect, commonly known as multipath interference, can distort the signal and make decoding/demodulating the RF signal particularly difficult for the hardware. Reflections can be of a different nature, as shown in Figure 8.

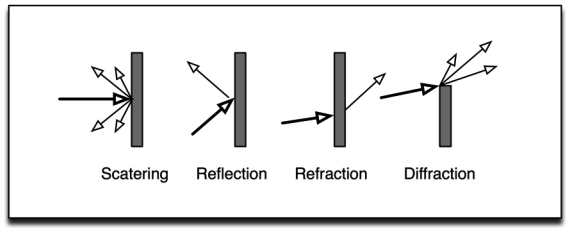


FIGURE 8: DIFFERENT WAYS REFLECTION WORKS

In some cases, a strong enough signal that is received out of phase with the direct signal can essentially create a blank, a spot where no signal is available, while only a few feet away you may have a strong signal. This is called a multipath null.

Using omni-directional antennas you will create large amounts of multipath interference, as the RF radiation is transmitted in all directions. Using directional antennas makes this less of a problem, as most RF energy is transmitted in one distinct direction.

ATTENUATION

An RF signal is reduced in strength once it passes through different materials. This effect is known as attenuation. The degree of reduction as related to material is shown in Table 2.

Material	Attentuation @2.4 GHz	Attentuation @5 GHz
Solid wood door	–6 dBi	–10 dBi
Steel fire exit/door	–13 dBi	–24 dBi
Concrete wall 18”	–19 dBi	–30 dBi
Interior wall 6”	–9 dBi	–4 dBi
Single pane window	–7 dBi	–6 dBi
Cubicle wall	–6 dBi	–2 dBi

TABLE 2: ATTENTUATION OF VARIOUS MATERIALS FOR WIFI SIGNALS

Table 2 shows that materials such as concrete walls or floors have much more attenuation than cubicle walls. In designing a radio network it is important to determine the number and material of the obstacles the signal must travel through. This has to be included in calculations of free space loss.

SIGNAL-TO-NOISE RATIO

The relation of the target signal of the radio system to other signals in the same frequency band is called the signal-to-noise ratio (SNR) and is expressed in dB as: SNR [dB] = Signal strength [dBm] – Noise floor [dBm] (see Figure 9). The more noise there is, the more difficult it becomes for the front-end electronics (or software) to filter the desired signal from the noise. Often the data rate the system is working on will drop if the SNR decreases. The more complex the modulation (encoding) of the RF signal is, the higher the bandwidth the system is capable of, and the higher the minimum SNR needs to be to get reliable communications.

Noise can be generated by different pieces of equipment. For 2.4 GHz, noise coming from stray microwave radiation, video transmitters, or ham radio operators is notoriously bad. Other sources of noise are X10 home automation equipment and cordless telephones.

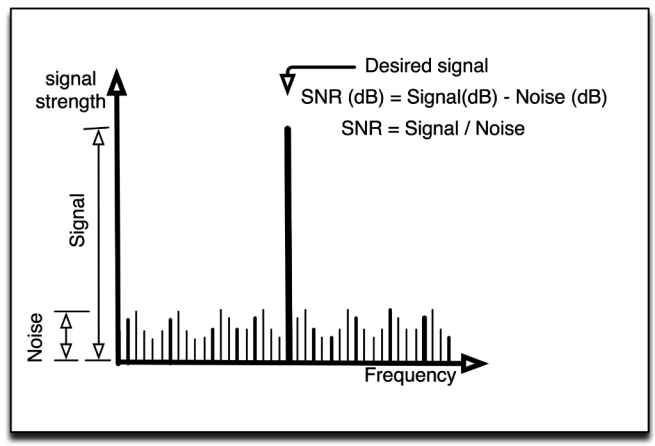


FIGURE 9: SIGNAL-TO-NOISE RATIO

An example here is the operation of Bluetooth and WiFi at the same time. The SNR of one system will decrease when the other system is turned on and the antennas are located close together. The Bluetooth RF signal is noise to the WiFi subsystem and vice versa. Other WiFi signals from adjacent channels are also noise to a client. Since the spectrum width of WiFi signals overlap, it is important for performance to use channels that are at least three apart.

GENERAL

These effects are applicable to all RF systems. WiFi (IEEE 802.11), Bluetooth (IEEE 802.15), ZigBee (IEEE 802.15.4), and 3G communication systems all use RF in roughly the same frequency space (2.5 GHz), and all have the same characteristics in the lowest layer. However, the higher layers of the different protocols differ quite extensively, and effects on the RF layer may be less profound for the user, since the other layers have the ability to correct errors introduced on the RF layer. An example here is that WiFi 802.11 a/b/g systems often use diverse techniques to overcome problems due to

multipath interference or multipath nulls. Here two antennas are used and the strongest signal is fed into the demodulator.

Conclusion

Building a reliable RF link is not straightforward. Radio frequency signals have their own special ways of traveling from the transmitter to the receiver and have to be handled carefully. Taking everything into account, RF signals are an easy and convenient way to transmit data. Because the medium is shared, however, you need to be aware of eavesdropping and use protocols or settings that match your security policy.

ACKNOWLEDGMENTS

I want to thank Rik Farrow for his comments on this article. It is always nice to have comments from people outside the field, since engineers get dragged into the heavy stuff too easily. My employer, Competa IT, also deserves to be mentioned here for the freedom I get to write these articles.

[1] http://www.sdrforum.org/pages/aboutSdrTech/whatIsSdr.asp.

[2] http://usenix.org/publications/login/2009-04/pdfs/vandrunen.pdf.

[3] http://www.usenix.org/event/usenix03/tech/freenix03/full_papers/vandrunen/vandrunen_html/index.html.

DAVID N. BLANK-EDELMAN

practical Perl tools: you never forget your first date (object)

David N. Blank-Edelman is the director of technology at the Northeastern University College of Computer and Information Science and the author of the O'Reilly book *Automating System Administration with Perl* (the second edition of the Otter book), available at purveyors of fine dead trees everywhere. He has spent the past 24+ years as a system/network administrator in large multi-platform environments, including Brandeis University, Cambridge Technology Group, and the MIT Media Laboratory. He was the program chair of the LISA '05 conference and one of the LISA '06 Invited Talks co-chairs.

dnb@ccs.neu.edu

AS THE MIGHTY BOOSH WOULD SAY, come with me now on a journey through time and space. Actually, that's a bit of an exaggeration. We're only going to go on a journey through time. Okay, too ambitious. New idea: we're going to look at Perl modules that deal with time. More precisely we're going to embark on a small survey of the most popular packages people use to handle date-like things in Perl today.

Before we dive in, I should mention that my predecessor in this column's pilot seat, Adam Turoff, wrote a date-related article back in 2005 in which he discussed a couple of the choices I'll mention below. We'll take a slightly different tack in this column (plus a decent amount has changed in the last five years), but I wanted to dispel any worry that the déjà vu feeling you might have is actually a glitch in the Matrix when they change something. Or maybe it is, I dunno.

Not the Built-In Stuff

For most of this column, I'm going to intentionally ignore both the built-in time functions that are part of the language and the time-related modules that ship with the core. Let's do a very quick review so we can get on to the more sophisticated stuff.

By far, the most used built-in function is time(), which will return a number like this:

```
1259340967
```

This is the "number of non-leap seconds since whatever time the system considers to be the epoch. . . . On most systems the epoch is 00:00:00 UTC, January 1, 1970." It is pretty common to use this function to grab the current time and then later on call it again to determine how much time has elapsed.

The very next thing most people do with the return value from time() is feed it to the localtime() (and less commonly) gmtime() functions. Both take that number of seconds and translate them into a list of the time buckets humans are likely to use, namely:

```
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday, $isdst) =
                                        localtime (time);
```

The difference between localtime() and gmtime() is that localtime() attempts to return values based on the current time zone, and gmtime() returns it relative to GMT, also known as UTC.

The return values above are mostly self-explanatory except for the last few. $wday is the day of the week (starting at 0 for Sunday), $yday is the day of the year (also starting at 0), and $isdt is true if Daylight Saving Time is in effect. If remembering which item is in which position in a list like that is too annoying for you, Perl ships with Time::localtime and Time::gmtime modules that let you work with those values as an object instead of a list. This lets you write code that looks like this, to quote an example from the documentation:

```
use Time::localtime;
printf "Year is %d\n", localtime->year() + 1900;
```

The idea of representing dates as objects is something that will figure quite prominently in the rest of this column.

I should mention that this is not how I use localtime() most of the time. I mostly call localtime() in a scalar context, explicitly like this:

```
print scalar localtime; # output: Sat Nov 28 22:31:27 2009
```

In a scalar context, it returns a ctime()-like string (what you might expect to see if you typed "date"). I use this all the time for putting timestamps into logs and files. If you want to generate a string with a different format, that segues nicely to the last two functions found in the core Perl distribution I want to mention before we move on: strftime and mktime. The POSIX::strftime() function takes a format string followed by a list of values like those returned by localtime() and gmtime() and returns the appropriately formatted string. The example the documentation provides is:

```
use POSIX;
$str = POSIX::strftime( "%A, %B %d, %Y", 0, 0, 0, 12, 11, 95, 2 );
print "$str\n"; # prints "Tuesday, December 12, 1995"
```

If you wanted to return the number of seconds since the epoch, you would use POSIX::mktime() instead (again from the POSIX module docs):

```
use POSIX;
$time_t = POSIX::mktime( 33, 26, 12, 27, 10, 109 );
```

Keep It Small, Fast, and Simple

That's the motto of the module Date::Calc. So far we've just seen ways to bring time information into our programs but no real information on how to manipulate it. Simple calculations such as "Find the time exactly an hour from now" are easy, but how about "What's the first Monday in December of this year?" With Date::Calc, the answer is:

```
use Date::Calc qw(Nth_Weekday_of_Month_Year);
# year, month, day of week, Nth occurrence
print Nth_Weekday_of_Month_Year(2009,12,1,1); # prints "2009127"
```

Date::Calc has a ton of functions like:

- - leap_year($year) to determine if the year is a leap year
- - Monday_of_Week($week,$year) to determine the first day of that week
- - Add_Delta_Days($year,$month,$day, $Dd) to add $Dd days to the date

That's just a few, so be sure to see the documentation for the full scoop. I'd especially recommend you check out the Recipe section of the documentation, which offers you ways to answer questions like "How can I calculate the last business day (payday!) of a month?" and "How can I send a re-

minder to members of a group on the day before a meeting which occurs every first Friday of a month?"

There's a considerable amount of power to be found in this module, but it does have its drawbacks. The first is a dependence on a C compiler (ideally the C compiler the Perl binary was built with) to build the module. Key sections are written in C, which is how it manages to provide at least the "fast" in its motto. There are a number of situations where this requirement would rule out the use of the module, so the Date::Calc author has also re-implemented those sections in pure Perl. Date::Pcalc is the result, trading speed for portability and deployability. Date::Calc is also fairly low-level in its abstraction. Although it has an optional OOP wrapper module, on the whole the module doesn't offer a particularly object-oriented shiny glint by default. At the very least, you still have to think in terms of putting together more complex calculations using chains of smaller operations (which, in general, I think is a good approach). For more DoWhatIMean sauce, we're going to have to look elsewhere.

Simple Representations

One way to provide more DoWhatIMean-itude (okay, I promise that's the last time I use that phrase, at least in this column) is to adopt a better representation of the very things we've been talking about, namely dates and times. If we can somehow take more natural descriptions of them (e.g., "2009-11-29") and work with those descriptions in a relatively intuitive manner, it will make things much more pleasant. Certainly more pleasant than trying to work with the number of seconds since the birthday of the actor who played "Mitch" in the movie *Real Genius* (or some such other arbitrary point in time).

There are a number of modules that provide this sort of representation. For working strictly with dates (without times), one of the simplest is, you guessed it, Date::Simple. Date::Simple lets you write code like:

```
use Date::Simple (':all');
my $date = Date::Simple->new('2010-01-29');
print $date->year;          # prints 2010
print today();              # prints '2009-11-29' when this was written
print date('2009-12-31') - date(today());      # prints 32 on that day
```

If you are more interested in working with times and dates, we can come back to a concept we saw earlier with Time::localtime. The Time::Piece module "replaces the standard localtime and gmtime functions with implementations that return objects." This means if you start with:

```
use Time::Piece;
my $tobj = localtime;
```

you have an object that provides a substantial number of methods, such as:

```
print $tobj->min;
print $tobj->year;
print $tobj->monname;           # prints Nov
print $tobj->fullmonth;         # prints November
print $tobj->time;              # prints 23:32:00
print $tobj->date;              # prints 2009-11-29
print $tobj->day_of_year;
print $tobj->month_last_day;    # prints the last day of the month
```

Hopefully, you read along that list and noted that it got more interesting and powerful as the list went on. Besides the very legible OOP-ness of the above, we also get the chance to work with the objects in a reasonable fashion to do arithmetic. For example, let's assume we have two Time::Piece objects:

```
$diff = $tobj1 - $tobj2;
```

If we just printed $diff, we'd get the number of seconds between the time/dates those objects represented. But even cooler than that, $diff is actually an object itself. It's a Time::Seconds object (as defined by the Time::Seconds module in the Time::Piece distribution). Why is that cooler? Well, it means we get to call methods on the resulting object like $diff->seconds, $diff->minutes, $diff->days, $diff->weeks, $diff->months, and even $diff->years. It's kind of nice to be able to work with these representations without needing to do the conversion arithmetic by hand. Time::Piece can do other neat stuff (e.g., date comparisons); see the documentation for more details.

If both the Date::Simple and the Time::Piece abstractions appeal to you and you feel torn deciding between the two, you may find that Date::Piece can help with that inner struggle. According to the documentation, it "extends Date::Simple and connects it to Time::Piece." What this means in practice is you can work with just dates and then pin your result down to a specific time on a date. The example the documentation gives is:

```
use Date::Piece qw(date);

my $date = date('2007-11-22');
my $time = $date->at('16:42:35');
print $time, "\n"; # is a Time::Piece
```

More Complex Frameworks

We have two more stops on this tour. There are two larger date/time frameworks that have found favor in the Perl community. Both are all-encompassing and will do everything you could possibly want, stopping short only at making you breakfast. They have their own way of looking at the world of dates and times, which is why I think it is good to look at both to find one that meshes well with the way you want to work.

The first, Date::Manip, really made its reputation in the community on the strength of its date-parsing skills. It's all well and good to talk about working with more natural representations of dates and times, but all of the examples we've given so far assume the programmer gets to pull those representations out of thin air. Equally often, we get handed a file, be it a logfile or some other date/time-laden glob of data, and our first task is to somehow translate its representation of dates and times into a form we can manipulate. For simple date parsing, a module like Date::Parse in the TimeDate distribution will work well. For the more complex stuff, there's very little that can touch Date::Manip. In addition to handling computer-y dates like those we've seen (e.g., "Mon Nov 30 22:30:46 2009"), it will happily parse strings like:

```
January 30
2001-01-01
Mar052009
Dec 1st 1970
next year
last Wednesday in December
3rd Tuesday in September
last Tuesday in 1973
```

```
14th
today
yesterday
Jan 2 2009 at noon
in 3 days at midnight
2 weeks ago on Friday at 10:00
```

Date::Manip is much more than just a fancy date parser. The documentation says:

> Among other things, Date::Manip allows you to:
>
> - Enter a date in practically any format you choose.
> - Compare two dates, entered in widely different formats to determine which is earlier.
> - Extract any information you want from ANY date using a format string similar to the UNIX date command.
> - Determine the amount of time between two dates, or add an amount of time to a date to get a second date.
> - Work with dates using international formats (foreign month names, e.g., 12/10/95 referring to October rather than December).
> - Find a list of dates where a recurring event happens

To do all of this, Date::Manip concerns itself with dates, deltas (amount of time between dates), and recurrences. You can construct objects that represent each and rub them together in ways that make sense (e.g., apply deltas to dates). Both this framework and the next one we're going to see are swimming in documentation, so I won't go very far into how you actually work with the module. Here's a very small example of Date::Manip code:

```
use Date::Manip::Date;

my $date = new Date::Manip::Date;
my $err = $date->parse('in 3 days at midnight');
print $date->printf('%C'); # prints 'Fri Dec  4 00:00:00 EST 2009'
```

Looks pretty simple on the surface, and that's probably a good thing.

So what's not to like? To answer this question, I have to repeat some of Date::Manip's history before I get to pile on the caveats. Once upon a time, Date::Manip was a very large, monolithic module that was known not only for its parsing power but also less positively for its memory requirements and comparatively slow speeds (at least when compared to some of the other modules that we've looked at). Previous versions provided a strictly non-OOP interface, and that made some people unhappy too.

The author took all of these things to heart and embarked on a total rewrite, going from version 5.x to 6.0x to mark the change. Date::Manip 6.0x is still written entirely in Perl, which means it isn't going to make Speedy Gonzales look like Regular Gonzales (as they say on Futurama), but the other criticisms have largely been, or are being, addressed quite well. It has a new OOP interface that can do everything the old functional one can and more. It's now a set of modules so you can load what you need, and so on. One parting caveat that is important to mention before moving on: the 6.0x branch requires Perl 5.10.x. If you still haven't upgraded to the latest stable version of Perl, you will need to use an older 5.x version of Date::Manip.

If Date::Manip doesn't excite you, perhaps our last module framework, DateTime, will. In a previous column, I mentioned the Perl Email Project. That was an attempt to replace the scattered functionality found in existing email modules with a new set of simple, well-architected, and unified packages that could become the definitive way of handling mail in Perl. A similar

effort was undertaken to do the same with time and date handling. The result was the DateTime project (http://datetime.perl.org) which has yielded a whole framework of DateTime::* modules for date/time representation, manipulation, parsing, and so on.

The main module, DateTime, lets you write code like this:

```
use DateTime;
my $dt = DateTime->new(  year       => 1973,
                         month      => 9,
                         day        => 15,
                         hour       => 23,
                         minute     => 10,
                         second     => 0,
                         nanosecond => 0,
                         time_zone  => 'America/New_York',
                         );
print $dt->year;
print $dt->day;
```

Does that code look a little familiar? What if I add this code to the previous example:

```
my $dt1 = DateTime->now();
my $diff = $dt1 - $dt;
my ($years,$months) = $diff->in_units('years', 'months');
  # prints '36 years and 2 months old'
print "$years years and $months months old\n";
```

Yes, we've seen this sort of date/duration object representation and date calculation in previous sections of this column. The syntax is a little different, but besides one small twist (the '-' is overloaded so we can actually subtract one object from another), it's the same song.

The core DateTime module doesn't have all of the functionality we've seen in other modules, but it doesn't have to. It is meant to glue together with other modules in this framework. For example, DateTime makes it possible to load what it calls formatters that provide new format parsers and output routines. This makes it possible, for example, to extend DateTime to handle some of Date::Manip's impressive formats (courtesy of DateTime::Format::Natural) and some more, umm, esoteric ones such as "The big hand is on the twelve and the little hand is on the six" and "La grande aiguille est sur le douze et la petite aiguille est sur le six" (courtesy of "DateTime::Format::Baby"). See the datetime.perl.org Web site for a listing of other extension modules and further documentation on the module.

I'm not going to run out of time to talk about date and time handling in Perl, but I'm certainly going to run out of space, so let's end things here.

Take care, and I'll see you next time.

PETER BAER GALVIN

Pete's all things Sun: Crossbow

Peter Baer Galvin is the chief technologist for Corporate Technologies, a premier systems integrator and VAR (www.cptech.com). Before that, Peter was the systems manager for Brown University's Computer Science Department. He has written articles and columns for many publications and is co-author of the *Operating Systems Concepts* and *Applied Operating Systems Concepts* textbooks. As a consultant and trainer, Peter teaches tutorials and gives talks on security and system administration worldwide. Peter blogs at http://www.galvin.info and twitters as "PeterGalvin."

pbg@cptech.com

AS SOLARIS 10 GETS READY TO ENTER its fifth year, it's already looking rather aged. The distance between it and its offspring, OpenSolaris, is growing, and it's changing slower than it used to. In fact, there are a large set of features available in OpenSolaris that are not, and likely will not be, integrated back into Solaris 10. Unfortunately, the mainstream production operating system from Sun is Solaris, and the move to Solaris "next," which will be based on OpenSolaris, is still in the future.

The good news is that the changes in OpenSolaris are so extensive that it's difficult for Sun to reintegrate those changes into Solaris 10. The bad news is that to use these new features, we have to use the less-supported (at least by ISVs, if not by Sun) OpenSolaris. Those features are becoming so compelling that, for some environments, it might be worth the added OpenSolaris production challenges to gain access to those new features. One such feature is ZFS deduplication, which removes blocks that are already stored within ZFS when a write request is received, replacing them with pointers to the existing blocks, rather than storing yet another copy of that same block. That functionality is the major feature of Data Domain, which was recently sold to EMC for a very large sum of money. That is not the topic this month (but will certainly be included in a future column). Rather, another exciting, innovative, and important OpenSolaris feature is beckoning our attention: Crossbow.

Crossbow is a major new feature of OpenSolaris, hiding within its simple interface powerful virtualization and resource management abilities. Crossbow makes it possible to implement an entire network within a single Solaris instance and to put fine-grain controls on how much data is flowing from and to each network component. Crossbow was integrated into OpenSolaris this year, in build 106, with its first official release coming in OpenSolaris 2009.06. The paper describing Crossbow was published at the USENIX LISA '09 conference and won the conference's Best Paper award [1]. In this column I explain what Crossbow is and how to implement and manage its features.

Overview

Crossbow builds on the previous Solaris networking improvement projects, including Firetruck (which brought speed and multi-threading improvements to Solaris 10). Crossbow is a new virtualization layer within OpenSolaris, part of the core network feature set. Most of the features are off by default but are easily enabled and are efficient, powerful, and easy to use.

The features of Crossbow include:

- The ability to implement an entire virtual-wire network (vWire), virtually, within OpenSolaris, including firewalls, servers, routers, and switches
- Fine-grained network resource management Quality of Service (QoS), including per-protocol bandwidth limits, traffic priorities, CPU assignments, and VLAN tags
- vWires that can be reduced to a set of objects and rules which can be modified or replicated, allowing network modeling, debugging, and performance tuning within a virtualized network
- The ability to take advantage of native hardware features for security and performance
- Full zone/container and Xen domain awareness

Essentially, the feature set of Crossbow decouples an application from the physical network, allowing flexible network design based on application needs rather than underlying physical components. Such an application can then be moved or duplicated more easily, since the same virtual network can be deployed on varying physical components.

Crossbow provides all of these features without significant performance impact on network traffic (according to testing by Sun [1]). Solaris shops will very likely want to take advantage of these new features.

The remainder of this column explores these features and demonstrates how to use them, including the very impressive (but unsupported) vWire Builder. To get started you might want to read the "How To" guide published by Sun [2], and the Crossbow documentation [3].

Exploration

There is a bit of new terminology that comes with Crossbow, which is necessary to understand Crossbow's features and functions.

- VNIC: A virtual network interface controller. Exactly the same as a physical network NIC, but virtual. A VNIC can be plumbed, ifconfiged, snooped, and dladmed.
- Etherstub: A virtual switch (named a "stub" because it is similar to the programming concept of stubbing a subroutine as a placeholder before writing the full routine). It connects VNICs.
- Flow: A resource management component. Zero or more can be assigned to a link (physical or virtual) to implement QoS. Flows can control services (protocols and local and remote ports), transports, and local and remote IP addresses and subnets.

A physical port does not need to be plumbed or configured: a VNIC can provide all of that functionality. A VNIC within an Etherstub cannot send traffic directly outside that virtual switch, but the traffic can be routed outside the Etherstub.

Crossbow takes advantage of hardware features provided by the NICs. For example, most modern NICs have hardware classification capabilities that Crossbow uses to create "hardware lanes"—collections of hardware re-

sources such as ring buffers and DMA channels that accelerate and manage performance.

"Traffic flows" can be created to manage the resources used by these other components. Traffic flows span the whole network stack from NIC through sockets, allocating and limiting resources. For instance, we could limit all UDP traffic traveling through a NIC to a certain maximum amount of bandwidth. Traffic flows can be used to limit traffic to containers as well. Traffic flows are managed via the flowadm command.

Because VNICs are essentially a full hardware NIC, virtualized, features such as snoop work with them. Before Crossbow, the only way for a container to manage a NIC and perform functions such as snoop was to dedicate a NIC port to the zone via the "exclusive-ip" container configuration feature. With Crossbow, VNICS can be given to containers and managed (and snooped by the container). One limit on VNICs is that they cannot be created on top of other VNICs. Like containers, VNICs are one layer deep. VNICs are managed by the dladm and ifconfig commands. Multiple VNICs configured on the same NIC cause the automatic creation of a virtual switch to connect all of the NIC's VNICs. Each NIC with VNICs gets its own independent virtual switch.

The Etherstub feature is similar to the auto-created virtual switches, except that it is independent of the NIC hardware. It can be used to create a virtual network that allows communication between VNICs within the kernel, without any hardware interaction. VNICs that are not part of the same virtual switch cannot talk to each other, while those that are part can communicate. Figure 1 shows the kinds of network configurations that can be created by Crossbow.

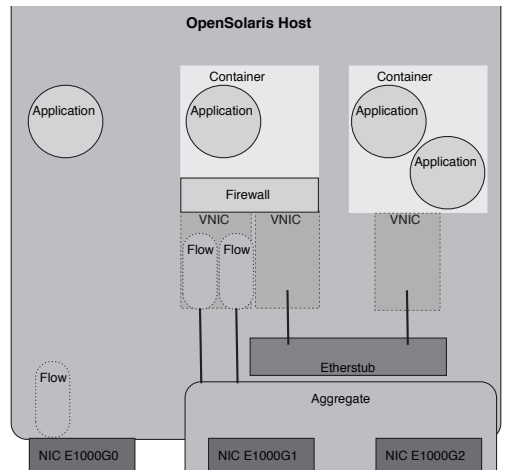


FIGURE 1: A SOLARIS SYSTEM WITH CONTAINERS AND CROSSBOW FEATURES

How do these commands work?

To create a new virtual NIC named "vnic1" on physical network device "e1000g0", execute:

```
$ pfexec dladm create-vnic -l e1000g0 vnic1
```

Because they are cheap, let's create another one called "vnic2":

```
$ pfexec dladm create-vnic -l e1000g0 vnic2
```

To check the status of all VNICs, execute:

```
$ pfexec dladm show-vnic
```

LINK	OVER	SPEED	MACADDRESS	MACADDRTYPE	VID
vnic1	e1000g0	1000	2:8:20:84:99:e7	random	0
vnic2	e1000g0	1000	2:8:20:82:e:fb	random	0

Notice that the MAC address is assigned at random, by default. Controls are available within the dladm command to allow manual MAC address management.

If the VNIC is going to have properties set, you can set them at creation time or afterward. For example, to create VNIC "vnic3" and limit the bandwidth of all traffic though that NIC to 40MB/sec:

```
$ pfexec dladm create-vnic -l e1000g0 -p maxbw=40 vnic3
```

To modify an existing VNIC, "vnic1," to have all of its activity execute only on CPU 1, and to give it a lower priority than any "medium" and "high" priority VNICs on the same NIC, you could say:

```
$ pfexec dladm set-linkprop -p cpus=1,priority=low vnic1
```

To get a detailed list of all the properties of VNIC vnic3, execute:

```
$ pfexec dladm show-linkprop vnic3
```

LINK	PROPERTY	PERM	VALUE	DEFAULT	POSSIBLE
vnic3	autopush	-w	--	--	--
vnic3	zone	rw	--	--	--
vnic3	state	r-	unknown	up	up,down
vnic3	mtu	r-	1500	1500	--
vnic3	maxbw	rw	40	--	--
vnic3	cpus	rw	--	--	--
vnic3	priority	rw	high	high	low,medium,high
vnic3	tagmode	rw	vlanonly	vlanonly	normal,vlanonly

To show the status of all VNICs on the system, execute:

```
$ pfexec dladm show-vnic
```

LINK	OVER	SPEED	MACADDRESS	MACADDRTYPE	VID
vnic1	e1000g0	1000	2:8:20:84:99:e7	random	0
vnic2	e1000g0	1000	2:8:20:82:e:fb	random	0
vnic3	e1000g0	40	2:8:20:a7:28:cc	random	0

Now let's create a virtual network switch and place two virtual NICs on that switch. This would be useful, for example, to allow two zones to talk to each other, but not on any physical network ports.

```
$ pfexec dladm create-etherstub inet1
$ pfexec dladm create-vnic -l inet1 vnzone1
$ pfexec dladm create-vnic -l inet1 vnzone2
```

To show the status of all links within the system, execute:

```
$ dladm show-link
```

LINK	CLASS	MTU	STATE	OVER
e1000g0	phys	1500	up	--
vnic1	vnic	1500	up	e1000g0
vnic2	vnic	1500	up	e1000g0
vnic3	vnic	1500	up	e1000g0
inet1	etherstub	9000	unknown	--
vnzone1	vnic	9000	up	inet1
vnzone2	vnic	9000	up	inet1

To continue that example, when building the zones, the two zones would be configured to use VNICs "vnzone1" and "vnzone2." Note that instant and

periodic information about various aspects of network traffic is available via the usual "-i" option to the commands. For example:

```
$ dladm show-link -s -i 5
LINK        IPACKETS  RBYTES    IERRORS  OPACKETS  OBYTES    OERRORS
e1000g0     49732     54930394  0        7829      758620    0
e1000g0     2         134       0        1         182       0
```

More fine-grained traffic management is performed via the flowadm command. For example, to manage port 80 traffic on "vnic1," first a flow is created that describes that traffic. We give it the name "httpflow":

```
$ pfexec flowadm add-flow -l vnic1 -a transport=tcp,local_port=80 httpflow
```

Next, we shape that traffic. For example, here we limit port 80 traffic on "vnic1" to 100Mb/sec (full duplex):

```
$ pfexec flowadm set-flowprop -p maxbw=100M httpflow
```

We could also set a limit on how much traffic we send to other systems' port 80s by using the property "remote_port." To list all flows on the system and examine the state of the httpflow flow:

```
$ pfexec flowadm show-flow
FLOW        LINK        IPADDR          PROTO    PORT    DSFLD
httpflow    vnic1       --              tcp      80      --
$ pfexec flowadm show-flowprop httpflow
FLOW        PROPERTY        VALUE     DEFAULT         POSSIBLE
httpflow    maxbw           100       --              100M
httpflow    priority        --        --
```

Flow statistics are available via the expected:

```
$ pfexec flowadm show-flow -s
```

Crossbow also uses the accounting subsystem to track network traffic. Here we enable that network accounting and examine its status:

```
$ pfexec acctadm -e basic -f /var/log/net.log net
$ pfexec acctadm net
   Net accounting: active
   Net accounting file: /var/log/net.log
   Tracked net resources: basic
   Untracked net resources: src_ip,dst_ip,src_port,dst_port,protocol,dsfield
$ pfexec dladm show-usage -f /var/log/net.log
```

A fairly astounding example of what can be done with Crossbow comes via the unsupported but very cool vWireBuilder tool [5]. Figure 2 shows a screen shot of the tool in use. In this example I've dragged a few network components onto the canvas, right-clicked on them to set properties such as IP addresses, and then compiled and executed the configuration. vWireBuilder then created this network configuration within my system by creating containers for each system function (firewall, Web server) and VNICs and Etherstubs for the network connections. It started a Web server within the Web server container. IP QoS was enabled by right-clicking on the various VNICs and setting bandwidth limits. I then added a network traffic load-generator, and vWireBuilder monitored the traffic via a drop-down menu item. Note that it did not configure the firewall within the firewall container; that would still need to be done by hand. The entire creation, from starting the tool to having the network configuration instantiated within my system, required only a few minutes. There is a demo video included with the tool that walks through some of its uses [5].

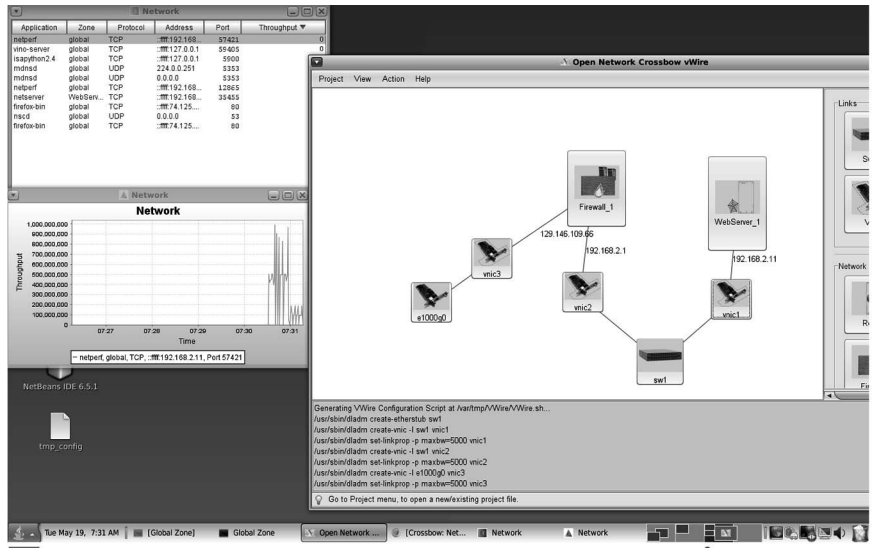


FIGURE 2: VWIREBUILDER CREATING A VWIRE

The Future

Crossbow seems to be a great framework for the next generation of Solaris networking features. Challenging and previously impossible network configurations, including fine-grained performance management, are made simple by its elegant feature set. For a just recently implemented facility, it works well and is feature-rich. Certainly more features will appear over time to refine and expand its functionality. In the future, the vWireBuilder GUI could capture a network configuration and allow its modification and then its instantiation with an OpenSolaris host. Such a feature would be a powerful addition to the system administrator's toolkit.

Conclusions

Project Crossbow extends the functionality and performance of the already impressive OpenSolaris network stack. Solaris 10 and OpenSolaris provide basic support for 10Mb through 10Gb HBAs, link aggregation at layer 2, IP multipathing (IPMP) at layer 3, VLAN tagging, routing, firewalling, and traffic snooping. Crossbow adds to this virtual NICs, virtual switches, and fine-grained IP QoS management of all networking resources. This new addition provides all of the components needed to implement, analyze, and performance-manage fully virtual networks. These features should allow Solaris administrators to design networks independent of the host hardware, and in the future the use of tools like vWireBuilder should allow for creation, modification, capture, and duplication of full virtual network infrastructures. The current state of Crossbow is expectedly in flux. There are demonstrations and blogs that refer to commands that no longer exist and to commands that may exist in the future. This results in a bit of confusion when trying out the features of Crossbow, but that exploration is rewarded with hints of a terrific new network feature set in OpenSolaris. See the Project Crossbow Wiki for more documents, discussion, details, and information on future features [6].

Tidbits

The USENIX LISA conference ran in November in Baltimore, and it had great content. I enjoyed teaching my Solaris tutorials and sitting in on some excellent presentations by other tutorial instructors and speakers. It was nice to see such a strong showing by Sun there, in the form of talks by Sunay Tripathi [7] about Crossbow and Bryan Cantrill [8] about the analytics in the Sun 7000 (fishworks) products. There were also Birds of a Feather sessions by Sun to present various activities that Sun is undertaking and solicit feedback. You can watch videos of these presentation, or listen to MP3s, via the LISA '09 Web site.

In OpenSolaris, as of build 129, ZFS has in-line deduplication. This is an important milestone for OpenSolaris, as companies that sell applications that perform in-line deduplication (Data Domain) are selling like $2B hotcakes. To try deduplication, first you must get your hands on the OpenSolaris 2009.06 distribution. After installing that, you need to update it to the latest developer build of OpenSolaris via these commands:

```
# pkg set-publisher -O http://pkg.opensolaris.org/dev opensolaris.org
. . .
# pkg image-update
. . .
```

Follow any extra instructions that image-update command requires. The net result should be a new OpenSolaris boot environment that incorporates the latest bits. A reboot into the new boot environment should bring up an OpenSolaris containing ZFS deduplication. Enabling deduplication is as trivial as setting a new property on an existing ZFS pool (to deduplicate future blocks written anywhere in that pool) or to an existing file system to just deduplicate inbound blocks being written there.

```
# zfs set dedup=on zpool
# zfs set dedup=on zpool/home
```

Rather astounding! There are more details, and discussions of the algorithms used, in Jeff Bonwick's blog: http://blogs.sun.com/bonwick/entry/zfs_dedup.

REFERENCES

[1] https://db.usenix.org/events/lisa09/tech/full_papers/tripathi.pdf.

[2] http://www.opensolaris.com/use/crossbow_network_containers.pdf.

[3] http://hub.opensolaris.org/bin/view/Project+crossbow/Docs.

[4] http://wikis.sun.com/display/OpenSolaris/vWireExample.

[5] http://hub.opensolaris.org/bin/view/Project+crossbow/demo.

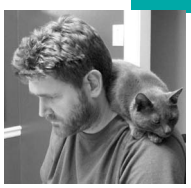
[6] http://hub.opensolaris.org/bin/view/Project+crossbow/Features.

[7] http://blogs.sun.com/sunay/.

[8] http://blogs.sun.com/bmc/.

DAVE JOSEPHSEN

iVoyeur: breaking up is hard to do



Dave Josephsen is the author of *Building a Monitoring Infrastructure with Nagios* (Prentice Hall PTR, 2007) and is senior systems engineer at DBG, Inc., where he maintains a gaggle of geographically dispersed server farms. He won LISA '04's Best Paper award for his co-authored work on spam mitigation, and he donates his spare time to the SourceMage GNU Linux Project.

dave-usenix@skeptech.org

ON MAY 6, 2009, A MESSAGE WITH THE subject line "Nagios is dead, long live Icinga" was posted to the nagios-devel list [1]. It briefly described a fork of Nagios core called Icinga and outlined the reasons why the developers thought a fork necessary. That alone was pretty earth-shattering news to the Nagios community at large, but as the thread developed, it became even more controversial; secret meetings, corporate conspiracies, deceit, betrayal, public hangings . . . well, all of that stuff except the public hangings. Anyway, I thought this month it would be fun to put on my journalist hat (otherwise known as my "prejudiced troll" hat) and relate the story thus far.

Nagios is a fantastic project, and a fork could affect a lot of folks, including a large percentage of the readership of this magazine. My first reaction to the idea of a fork was negative; it didn't seem as though good things could possibly come from it. When I saw who was behind the fork, several German developers from the Nagios Advisory Board as well as Netways.de (the guys who run nagiosexchange.com and organize arguably the largest annual Nagios conference), it became obvious that there were some legitimate gripes behind it. Although there are a fair number of personalities and motivations involved, I think those gripes are probably the best place to start. Let's take a look at Icinga's stated reasons for forking, the reaction of Ethan Galstad (the creator of Nagios) to them, and also some analysis of my own (because what would you do without my razor-sharp insight?).

Icinga's reasons, taken from their Web site and the thread mentioned in [1], are:

- Nagios development has stalled, because there is only one person with CVS commit access (namely, Ethan), and he's been inactive (meaning he's been ignoring bug fixes and new feature patches from the community, and further that he's not responding to email or list traffic).
- Nagios Enterprises, the commercial face of Ethan, has lately begun "harassing" developers about trademark infringement (meaning Ethan's lawyers are going around demanding devs and community members turn over domain names they've registered that have the name "Nagios" in them).
- NDOUtils (a database connectivity toolkit) is "still buggy."

- The Web interface is old and/or ugly (arguably a question of esthetics, but the general feeling for a number of years has been that the UI should be rewritten in PHP; it is currently a CGI written in C).

Although these seem like four unrelated issues at first glance, they actually break down into two groups of very closely related issues. The first group being the first two bullet points—that development has stalled because Ethan is a bottleneck, and that community members are being hassled by Nagios Enterprises lawyers. As evidenced by [2] and [3], the fork members felt that Ethan went AWOL after the 3.0 release. Simple bug fix patches were ignored (although a few critical bug fixes were, in fact, merged into core), Ethan was not providing a detailed road map forward, and new features they desperately wanted were nowhere near being merged into core.

Understanding that Ethan is a busy guy, several members of the Advisory Board got together during and after the 2008 Netways conference and created a patch queue system to streamline the process [4]. Ethan seemed to ignore the system, as well as attempts on the part of the community to open up the development process. When the trademark infringement notices started coming in, it was particularly frustrating to many members of the community. Their concerns went unheard while the gift of their labor and their years of enthusiastic evangelizing were being met with cease-and-desist letters.

For his part, Ethan responded that he had been forced away from development work by some legal trouble [5]. Without providing any real detail, he did mention Netways.de as the source of the problem. Although Ethan doesn't tell us this, what apparently happened was Netways.de (a German corporation) registered the name "Nagios" as a trademark in Germany [6], and Ethan was forced to sue them to get his name back. His attentions thusly diverted, Nagios core began to rot. Given this context, we gain some insight into the lack of attention to project work on his part, as well as the trademark authoritarianism. Given the legal entanglement, when Ethan was blindsided by a fork, his off-the-cuff reaction was that it was an attempt by Netways.de to undermine his credibility with the community.

From Ethan's perspective, Nagios had had a falling-out with Netways.de, so Netways.de was attempting to fork Nagios and pirate the community away to a project they controlled. If they couldn't steal Nagios in the courtroom, then they would attempt a PR coup and rename the project Icinga. Netways.de has explicitly denied this accusation [7], but the fact that the Icinga project lead is also the CTO of Netways.de lends some strength to this perspective. Netways.de's case is also not helped by the fact that the fork was planned and orchestrated in secret and without any attempt to notify or warn Ethan.

Tinfoil hats off and drama aside, I think I share the opinion of most of the community and tech press out there when I say that Nagios did not have a scalable development model, and that's a bad thing. If a project the size of Nagios can be halted by diverting one person's attention, that's not just a gaping procedural problem but also a security bug. Netways.de didn't just cause Ethan legal headaches, they DoS'd his development model (whether they meant to or not is another interesting question). Clearly Nagios needs a few more lead devs with commit access.

If Icinga has done anything, it's opened Ethan's eyes to this fact, and he claims to be taking steps to rectify that situation [8]. Other devs in the community have confirmed that he's approaching them about lead dev roles and commit access [6]. He's also vowing to correct his admitted communication deficiencies [8], and he's written numerous blog posts to clarify his position on the various topics surrounding the fork.

I've never personally had any contact with Ethan (having written the Prentice Hall book on Nagios, I can personally attest to the fact that he's a hard guy to get ahold of—even my publisher couldn't do it), but one does get a sense of a person from reading their code, and the sense I get from the work I've done in the Nagios Core source is that Ethan is a bright and meticulous craftsman. I don't think it's going to be easy for him to find people he feels have the right combination of skill and common sense to give them commit access. Hopefully, by the time you're reading this he'll have found at least two or three.

The second set of bullet points, that NDOUtils is buggy and that the UI is old, ugly and/or poorly designed, are legitimate in that they are undeniably factually correct. NDOUtils is in fact buggy, and although I don't personally have a problem with CGIs written in C, I assume most people under 40 would concede that, for better or worse, that's just not how things are done anymore.

Ethan, for his part, has not commented on these points, but other developers in the community have [9]. The thing is, there are quite a few deficiencies in Nagios Core that might cause one to consider forking, and although the UI might be up there for some people, NDOUtils probably wouldn't be, so I think it's a little disorienting for most Nagios devs to see these two issues so high on Icinga's list. From the Icinga devs perspective, however, these two issues are so closely related they're actually interdependent. The relationship between NDOUtils and the UI is subtle for the rest of us, because it revolves around a technical assumption being made by Icinga's UI design team. It's an assumption that I don't think is necessary, but I do see that it's being made and why, so I'll attempt an explanation.

The Icinga UI devs have a problem typical for anyone writing Nagios tools that deal with state data: namely, there is no trivial way to ask Nagios for the current state of a given service or host (now *there's* something one would expect to see in a list of "reasons to fork Nagios core"). So the Icinga devs want to export state data to an interface other than the built-in Nagios UI (because in this case they're re-implementing that UI). As it turns out, this is a problem these particular devs have run into before, because some of them wrote NagViz (a visualization add-on for Nagios) during which they solved this problem by exporting state data to MySQL using NDOUtils. So the Icinga UI devs are using a bit of constructive laziness and making the underlying assumption that NDOUtils is going to be a core component of the new PHP-based UI they're creating.

There's a lot I'm tempted to say about this line of thinking, but the most topical point is that the design they've chosen to pursue could be implemented in Nagios without any changes in core whatsoever. Indeed, the current version of Icinga appears to be Nagios with a patched NDOUtils add-on and a stand-alone PHP application reading state data from a database. The Icinga guys don't need a fork to make this UI thing happen, and trying to fight for mind-share for a new UI would probably be a great deal easier than fighting for mind-share for an entire fork. So why fork to glean functionality that you already have? Why not just fork the NDOUtils add-on?

Another point of interest is that NDOUtils is not what I think most Nagios admins would consider the "right answer" for this job. It's fine for people who want to write add-ons that need access to the daemon's state data, but it's not what I'd call the optimal interface to build a replacement UI around. The Icinga guys are beyond writing add-ons here. If I were them, I'd be looking to write my own NEB module, one that was lightweight and specialized to my purpose, and I'd probably avoid using an external database tier at all. Why sync state data to an external DB, and then sync the UI to the DB?

Instead, why not just sync the UI directly to the daemon using a specialized message-passing NEB module, or even something like op5's Merlin [10]? Having written an NEB module or two [11], I can assure you this sort of thing is completely within the current operational capabilities of Nagios Core, no forking required.

But since they are forking core, they could take it one step further and fix the actual problem inside the daemon. The more I think about it, the more puzzling it is to me that the Icinga devs have decided on the direction they have. Given that they're forking Nagios core, and given that they are a smart bunch of guys with plenty of experience hacking Nagios, this dependency chain they're creating between a database add-on and the core UI just doesn't make sense. I would hope that if one had the opportunity to rewrite Nagios one would consider functional improvements first and eye candy later. State data is hard to export, so write an API that more directly exports it, or, if databases are what you want, then internally replace the state data file with a database. Writing the sexy UI first and then patching an already kludgy add-on to provide it data smacks of a dangerous and extravagant naiveté. It screams "kewlist GUI wins!" and it's that kind of thinking that would make me think twice about handing out CVS commit access too.

As you might have guessed by now, I'll be sticking with Nagios, but I'll also be keeping an eye on Icinga. Hopefully, something interesting will develop from this fork, but my real hope is that Ethan fixes the bottlenecks and communication problems, and Nagios and Icinga can get past their differences and merge back into a single project again. The legal complications of their parent companies makes that unlikely in the foreseeable future, but I think it's a net loss for the community to have to split its effort between these two projects for any length of time. It also makes tenuous the positions of the myriad companies out there who are providing commercial products and services based around Nagios. These companies, such as Groundwork [12] and op5 [13], are currently where much of the ground-breaking work on Nagios is being done. The community loses if those companies have to switch gears and worry about supporting a fork. Time will tell.

Take it easy.

REFERENCES

[1] http://thread.gmane.org/gmane.network.nagios.devel/6246.

[2] http://article.gmane.org/gmane.network.nagios.devel/6267.

[3] http://article.gmane.org/gmane.network.nagios.devel/6270.

[4] http://www.mail-archive.com/nagios-users@lists.sourceforge.net/msg21839.html.

[5] http://article.gmane.org/gmane.network.nagios.devel/6253.

[6] http://blogs.op5.org/blog4.php/2009/05/07/the-future-of-nagios.

[7] http://markmail.org/message/yba2p2p3w7aq7p4v.

[8] http://community.nagios.org/2009/05/11/the-future-of-nagios-where-do-we-go-from-here/.

[9] http://article.gmane.org/gmane.network.nagios.devel/6273.

[10] https://wiki.op5.org/merlin:start.

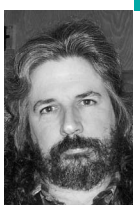
[11] http://www.usenix.org/publications/login/2008-12/pdfs/josephsen.pdf.

[12] http://www.op5.com.

[13] http://www.groundworkopensource.com.

ROBERT G. FERRELL

/dev/random: project management—a primer



Robert G. Ferrell is an information security geek biding his time until that genius grant finally comes through.

rgferrell@gmail.com

PROJECTS ARE THE CHIEF MECHANISMS by which modern companies dispose of their liquid assets. A great many guides to IT project management exist out there, but all suffer from one major shortcoming: none of them were written by me.

If you allow them, the "experts" will drone on and on about the insane intricacies of the project management process, because we as a society have a deep pathological need to reduce every facet of daily existence to incomprehensible PowerPoint bullets. Yet we are all project managers to a certain extent. Oh, I could sit in my (mammoth) ivory tower and toss off terms like *work authorization systems*, *duration compressing*, and *reverse resource allocation scheduling* to impress you, but I'm not going to, because that sort of yammer gives me a headache, not to mention the fact that I really don't know what any of it means. What I will do is give you, based on considerable life experience, a condensed overview of the typical project lifecycle that you can actually comprehend without an interpreter or prescription painkillers. I won't guarantee a positive ROI, however.

Phase I: Initiation

In this phase someone, usually someone with a key to the executive washroom and a tenuous grasp on reality, comes up with an idea that he or she wants to see implemented in the organization. The germ of this concept probably came from a trendy magazine next to the toilet, late-night perusal of an online forum, or a chimichanga with habañero salsa-induced nightmare. In the most toxic cases, all three.

The project manager 's job is to translate this executive hallucination into a plan that ordinary human beings actually have some minimal chance of eventually accomplishing, if not in fact, at least on paper. This brings us to Ferrell's Law of Projects: *Metrics Are All That Matters*, and its corollary, *Don't Sweat the Stuff You Can't Put in a Spreadsheet*.

Phase II: Planning

Now the game is truly afoot. The project manager will typically at this point plot out a rough course of action, establish a preliminary timeline, take stock of the available resources to carry out the various stages of the project, assess the likelihood of successful completion, and update her résumé. The veteran PM will begin vigorously to blur the

boundaries between reality and fantasy, even this early in the process, in order to obfuscate the stark nature of her preordained failure later on. Everyone who resides outside the executive suite knows the project as originally conceived has no realistic chance of succeeding, but all pretend to be merrily oblivious in order to keep their jobs. Project management puts the "funk" in dysfunctional.

A steady stream of pie charts, flow diagrams, slides, and even, in some extreme cases, three-color brochures on glossy card stock, will now spew from the PM's white-hot caldera. This torrent will not abate until implementation is well underway.

Phase III: Executing

As this phase gathers steam, all of the bric-a-brac has been ordered, has been delivered, and waits patiently in one or more piles nestled amongst the servers and network racks. If any new hires are involved, they are sitting in an HR conference room having several hundred pages of more-or-less incomprehensible employment-related gibberish folded into their brains like compost into a potato patch. The metrics geyser shifts from projections and cost-benefit analyses to adjusted productivity curves and time-to-maturity models. The PM begins receiving the occasional headhunter call on her cell phone. She answers any number she doesn't recognize with, "Mother, what's wrong?" and scurries into the ladies' room, janitorial closet, or employee cafeteria—whichever is closest—to take the call.

Meanwhile, incubation complete, the shipping containers have hatched and the embryonic progeny of the earlier phases takes a neonatal breath in the server room. Brightly colored cables slink in all directions. LEDs blink furiously. Configurations are configured, databases are populated, communications are debugged, users are authorized, and the information systems juggernaut lurches into ponderous motion. Another checkbox in the project timeline welcomes an inky resident.

Phase IV: Monitoring and Controlling

This is the period during which the newness rubs off and obsolescence sets in. At first it will be the occasional relatively inconsequential part that gives up the ghost, but soon the entire system will start to wobble and flap in the breeze like a decrepit scarecrow. Gradually, ineluctably, the stain of corruption inches forward while those few whose career aspirations have not yet been burned utterly from their souls begin to distance themselves from the doomed project. "I told you so" memos fly willy-nilly and damage control meetings are taken. Scapegoats are led from their pens to be groomed for ritual public display and bloodletting. The PM is taking "sick leave" days for job interviews.

Phase V: Closing

The name of this phase is a bit misleading. In point of fact, what goes on here is really more the preparation for the Initiation phase of the next project than the closeout of the current one, since by this time the systems constituting the old project have totally ceased to operate, with the hardware having been mined for any usable spare parts. The IT department has quietly reverted to the perfectly functional system that was in place prior to the ill-fated project. The users haven't really noticed, as the graphic interfaces for the two systems are virtually identical. The project manager has taken a

much more lucrative position elsewhere, and a new PM has been brought in to assess the unfortunate situation and assign all blame to his predecessor.

Over the next six weeks or so he painstakingly gathers the same set of statistics, user requirements, and out-of-context statements of dissatisfaction with the system that the previous PM had compiled and offers his conclusions in a presentation that matches the previous PM's almost slide-for-slide (he may even use the *same* slides). Having thus been briefed, executive management experiences another excursion into questionable cuisine, from which germinates the next in a series of brilliantly ill-conceived projects. Failure by design has become a critical element of the corporate mission statement. They're ready to move into government contracting at this juncture.

There's a process in cellular respiration that any undergraduate biology major could tell you about called the *electron transport chain* in which a lot of complicated chemical reactions take place in a specific order just to move electrons from one atom to another. That's evidently the model for the project management cycle: a whole series of interconnected complex activities the final result of which is the transfer of a PM from one company/department to another. The *manager transport chain*, I guess we could call it.

All hail Rube Goldberg, the father of modern business management practices.

book reviews

ELIZABETH ZWICKY, WITH RIK FARROW, SAM F. STOVER, AND KIM GRILLO

WICKED COOL PHP: REAL-WORLD SCRIPTS THAT SOLVE DIFFICULT PROBLEMS

William Steinmetz with Brian Ward

No Starch Press, 2008. 181 pp.
ISBN 978-1-59327-173-2

Wicked Cool PHP is aimed at an audience that can write PHP—possibly because they learned how to code "Hello World!" somewhere else, possibly just because it's not that hard if you already know other programming languages—but want to know how to do interesting things. In another context, it might be called a PHP pattern book. It gives frameworks for accomplishing common tasks, along with information about common mistakes people make when trying to do these things.

It comes from a sensible but relatively naive security perspective. For instance, it tells you to use MD5 hashes instead of shuttling passwords around, but does not have any information about salts. It gives information about letting Web site users send mail, but doesn't suggest that you need controls to keep from becoming a spam engine. It doesn't give you any information about avoiding cross-site request forgery (where a user who has a valid cookie is tricked into sending a request). Nonetheless, it does give appropriate warnings about PHP configuration, sanitizing user input, and avoiding SQL injection and cross-site scripting. It's not suitable as your only resource for building a complex site, but it's a good starting point.

ESSENTIAL PHP SECURITY: A GUIDE TO BUILDING SECURE WEB APPLICATIONS

Chris Shiflett

O'Reilly, 2006. 103 pp.
ISBN 0-596-00656-X

As you might expect, this does a much better job of covering security than *Wicked Cool PHP*. It goes much deeper into threats, covers general security techniques and attitudes (never trust the user! defense in depth!), and provides convincing examples of attacks. (A surprising number of people have difficulty getting their heads around the idea that people might not access your Web server from your forms.)

Unfortunately, it's not an easy read. For a motivated reader, a PHP programmer who wants the essentials from a PHP perspective, it's probably still a good choice. It's probably your only choice for some specialist topics; it covers securing your part of a shared Web server, when other books are likely to leave you frustrated and despairing.

YES: 50 SCIENTIFICALLY PROVEN WAYS TO BE PERSUASIVE

Noah J. Goldstein, Steve J. Martin, and Robert B. Cialdini

Free Press, 2008. 272 pp.
ISBN 978-1-4165-7096-7

You may be sure that you are too smart to fall for silly tricks like having a sales pitch come from somebody "like you." Don't be so sure. In one study, nearly twice as many people were willing to fill out a survey form when it came from somebody with a name like theirs. And not one of them believed the name had anything to do with it.

This is the latest summary of research on influence, which Robert Cialdini is the grand master of. It's easy to read, convincing, and reassuringly convinced of the importance of actually being a reasonable person. It's also a little scary as you see just how effective some of these techniques can be.

Aside from being a fascinating read, this book is a useful tool for anybody who's trying to get people to do something (upgrade their software, clean out their over-full mailboxes, use approved channels to report problems, report the problems in the first place).

DON'T MAKE ME THINK: A COMMON SENSE APPROACH TO WEB USABILITY, SECOND EDITION

Steve Krug

New Riders, 2006. 197 pp.
ISBN 0-321-34475-8

To finish up, we neatly combine the themes of Web development and persuasion with the second edition of a classic book about Web design. Well, actually, about Web usability, which is to say, it's not about whether or not you should paint it blue and make the corners rounded, it's about building something that people will actually enjoy using. Not because you should panic now—on the Web people have no attention span and will flee to a competitor—but because, get this, people don't like it when things are hard to use, and torturing your customers is never good.

OK, so that doesn't sound like a deep insight, and without some advice on how to make things easy to use, it wouldn't be all that useful. But it's more of an insight than you might think, and it's a pleasant antidote to the people who wish to assure us that it is all different on the Web. This book will not only convince you that it's more important to have a usable Web site than to have a beautiful one, it will also provide you with a handy form letter to pass on to the boss who thinks that a Flash intro page is the best idea ever.

Along the way, you get a brief introduction to usability testing and how to do it cheaply and effectively, some amusing cartoons, and a number of straightforward instructions. It truly is both fun and educational.

CONFESSIONS OF A PUBLIC SPEAKER

Scott Berkun

O'Reilly, 2009. 220 pp.
ISBN 978-0-596-80199-1

REVIEWED BY RIK FARROW

This is a book of advice for potential and current public speakers. Berkun points out that everyone is a public speaker, even if this speaking only occurs over beers. But he also writes that most people who read his book won't be good public speakers (p. 140). He then repeats the single bit of advice he's been harping on: practice. Berkun goes beyond suggesting the practice is important, or even a virtue—it is the essential ingredient for good presentations.

If that was all he had to say, this book would be quite short. But Berkun has a conversational writing style that's easy to read, and he peppers his writing with storytelling, making this truly a book about confessions. Some of the confessions, such as that all speakers have some level of fear before a presentation, are very comforting. Berkun's story about being part of the B-roll footage for a CNBC five-hour primetime TV series was one of being panicked, and later, humbled: "First, I have no idea what is going on, yet I am the center of attention, much like how it would feel to be invited over for dinner by a family of cannibals" (p. 97).

Berkun offers lots of advice, much more than you can take in with one reading. Akthough I have presented routinely since 1986, I found lots of ideas I hadn't thought of myself, or things that I had done sometimes without making it part of my bag of tricks. For example, Berkun covers audience issues—like the giant, mostly empty room, or the frequent questioner who wants to take over your talk—by providing useful advice for dealing with these very real issues. Of all the suggestions offered, I want to repeat one that Berkun also repeats: the audience is there for you. Even if they want to attack you later, they want you to present your points first. And most of the time the audience is there because you are going to give them something useful, and so it has a vested interest in your success.

Even though most people will never become professional public speakers, we are, as Berkun says, all public speakers at some level. I found his book easy to read, enjoyable as well as useful, and a book I can recommend.

HACKING: THE NEXT GENERATION

Nitesh Dhanjani, Billy Rios, and Brett Hardin

O'Reilly, 2009. 296 pp.
ISBN 978-0596154578

REVIEWED BY SAM STOVER

This book is fantastic! Everyone on the planet should read it. Here are the goods:

Chapter 1 introduces the more obvious intel-gathering techniques such as dumpster diving and social engineering, but also logical techniques such as automated Google hacking, file metadata gathering, social network analysis, and email harvesting.

Chapter 2 dives into a fair bit of detail concerning JavaScript methods for turning browsers into access points into an organization. Lots of good reading on XSS, CSRF, attack automation, content ownership,

and even using the browser to steal files off of local file systems.

Chapter 3 might seem disappointing to the technically savvy, as it deals with protocols like FTP, Telnet, ARP, and SMTP. The authors justify the chapter, saying that these are long-standing issues with older protocols, and I have to agree. This might be old hat to most of us, but it's still a problem in the real world.

Chapter 4 explains blended threats, starting with application protocol handlers, which I found interesting. Not only are protocol handlers explained and used in blended threats, but methods are given to enumerate protocol handlers in Windows, OS X, and Linux. Good stuff.

Chapter 5 addresses cloud computing, with a focus on Amazon (EC2) and Google (AppEngine). Some basic attacks are presented, like poisoned virtual machines, and management console targeting. In some cases, XSS/CSRF come into play, connecting back to Chapter 2.

Chapter 6 leaves the world of Web technology to look at how mobile devices are being used and abused in the mobile workforce. Most of the chapter deals with stealing information from hotel networks and hotspots, and there are some tidbits concerning cell phone voicemail and physical attacks against cell phones.

Chapter 7 is one of the best phishing primers that I've ever seen. Example sites (that worked) are dissected in a way that makes it easy to see what was happening. An interesting take-away from this chapter is that successful phishers don't have to be techno-wizards; lame phishing pages still produce results. A decent intro into the e-crime underground is also provided, which provides a nice backdrop.

Chapter 8, an offshoot of Chapter 1, focuses on social engineering, showing how attackers use a small bit of information as a foundation to a profile. Things like calendars and social network sites can yield all that's needed to "buddy up," and several examples are discussed. Not being a fan of social sites, I feel vindicated after reading this book, although I'm not sure I buy into the viability of "sentiment analysis."

Chapter 9 provides a profile of steps for building a targeted attack. The chapter starts with an overview of motives, moves into information gathering methods, and finishes with attack options. Techniques such as penetrating the inner circle and "targeting the assistant" are discussed.

Chapter 10 gives two case studies that build from the previous chapters. Not very technical and pretty brief, but still realistic and representative.

Overall, this book is a triumph. Well written, solid material, and fun. Just what you'd expect from an O'Reilly book.

SCENE OF THE CYBERCRIME, SECOND EDITION

Debra Littlejohn Shinder and Michael Cross

Syngress, 2008. 744 pp.
ISBN 978-1597492768

REVIEWED BY KIM GRILLO

When I first read the title of this book I figured that after investigating cybercrime for the past five years as part of my job, this book would be too entry-level for me. However, the author introduces many tools and forensics techniques that were not covered either in my formal education or as part of my on-the-job training. The authors have a very good understanding of the issues that law enforcement and security professionals face when investigating cybercrime. These issues are introduced at the beginning of the book to lay the groundwork for the subsequent chapters. The book covers a wide range of information, including investigation of cybercrimes and security basics, best practices for preventing threats and implementing security, and collection of information for prosecution that would be of interest to law enforcement and IT professionals as well as those just entering the world of cyber investigations.

For example, Chapters 15, 16, and 17 spend a lot of time discussing the cybercrime legal process, which is a great introduction to the topic for someone who has never encountered it before or for the techie who needs to work closely with law enforcement. These chapters provide a good background on some of the obstacles law enforcement faces when working cybercrime, such as jurisdictional issues, and also provides an in-depth look into what to expect if testifying as an expert witness. There is also enough technical discussion of tools and tips to keep the technical crowd interested. Chapter 6 deals with Computer Forensic Software and Hardware, which provides a fair amount of information on disk imaging, file recovery, and Linux/UNIX tools. The chapter even includes a forensic software reference, over 30 pages of programs and utilities with brief descriptions of their uses.

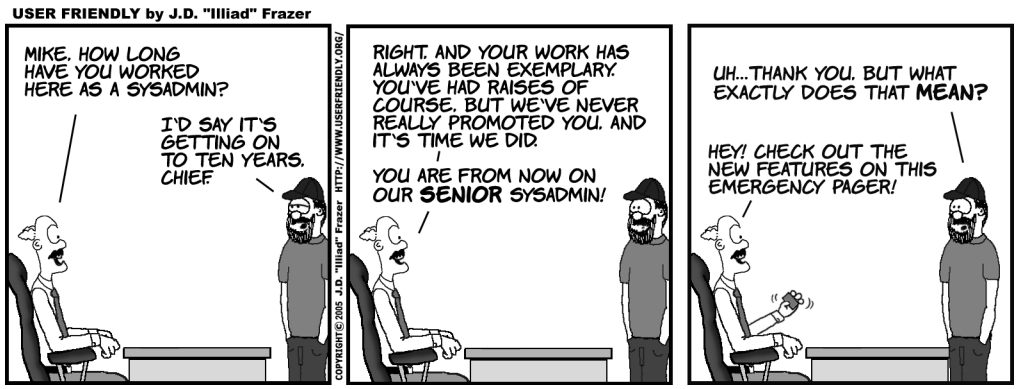
I think this book is targeted at law enforcement officers assigned to cybercrime cases with no formal background in information technology. The book probably covers more information than they would need to perform their job and at times might be a bit too technical for someone without an IT background. But this makes it very well suited to the technical individual who has responsibilities or an interest in cyber investigation, providing a good mix of the "known" (tools and tricks introduced in the Computer Forensics Software and Hardware chapter) and the unknown (learning about cybercrime legal process). Overall, this is a great book for anyone already working in the field of cybercrime as well as those just entering it.

Why the Semicolon in *;login:*?

The answer, as told to Peter Salus by Dennis Ritchie:

"The ; was utilitarian. During most of the early '70s the most popular terminal was the Teletype model 37. The sequence <esc>; put it into full-duplex mode so the terminal didn't print characters locally, but let the system echo them. So this sequence was put into the greeting message. Of course it didn't print when you used that terminal, but other terminals that appeared later didn't understand the message and so printed the ;."

—Peter H. Salus, *A Quarter Century of UNIX*, p. 69.



USENIX notes

USENIX MEMBER BENEFITS

Members of the USENIX Association receive the following benefits:

FREE SUBSCRIPTION to *;login:*, the Association's magazine, published six times a year, featuring technical articles, system administration articles, tips and techniques, practical columns on such topics as security, Perl, networks, and operating systems, book reviews, and summaries of sessions at USENIX conferences.

ACCESS TO ;LOGIN: online from October 1997 to this month: www.usenix.org/publications/login/.

DISCOUNTS on registration fees for all USENIX conferences.

SPECIAL DISCOUNTS on a variety of products, books, software, and periodicals: www.usenix.org/membership/specialdisc.html.

THE RIGHT TO VOTE on matters affecting the Association, its bylaws, and election of its directors and officers.

FOR MORE INFORMATION regarding membership or benefits, please see www.usenix.org/membership/ or contact office@usenix.org. Phone: 510-528-8649

USENIX BOARD OF DIRECTORS

Communicate directly with the USENIX Board of Directors by writing to board@usenix.org.

PRESIDENT

Clem Cole, *Intel*
clem@usenix.org

VICE PRESIDENT

Margo Seltzer, *Harvard School of Engineering and Applied Sciences*
margo@usenix.org

SECRETARY

Alva Couch, *Tufts University*
alva@usenix.org

TREASURER

Brian Noble, *University of Michigan*
brian@usenix.org

DIRECTORS

Matt Blaze, *University of Pennsylvania*
matt@usenix.org

Gerald Carter,
Samba.org/Likewise Software
jerry@usenix.org

Rémy Evard, *Novartis*
remy@usenix.org

Niels Provos, *Google*
niels@usenix.org

EXECUTIVE DIRECTOR

Ellie Young,
ellie@usenix.org

NOMINATING COMMITTEE REPORT, 2010 ELECTION FOR THE USENIX BOARD OF DIRECTORS

Rémy Evard,
Chair, USENIX Nominating Committee

The USENIX Association is governed by its Bylaws and by its Board of Directors. Elections are held every two years, and all eight Board members are elected at the same time. Four of them serve at-large and four serve as statutory officers: President, Vice-President, Secretary, and Treasurer.

Per Article 7.1 of the Bylaws of the USENIX Association, a Nominating Committee proposes a slate of Board members for the membership's consideration. As a practical matter, the purpose of the Nominating Committee is to balance continuity and capability so as to ensure that the incoming Board is composed of persons shown by their actions to be both dedicated to the Association and prepared to lead it forward.

The USENIX Nominating Committee is pleased to announce the candidates whom we have nominated for the upcoming USENIX Board of Directors election:

President: Clem Cole, *Intel*

Vice-President: Margo Seltzer, *Harvard University*

Secretary: Alva Couch, *Tufts University*

Treasurer: Brian Noble, *University of Michigan*

At Large:

John Arrasjid, *VMware*

David N. Blank-Edelman, *Northeastern University*

Matt Blaze, *University of Pennsylvania*

Jacob Farmer, *Cambridge Computer Services*

Niels Provos, *Google*

I am very pleased that all of these exceptional individuals have agreed to offer their time and talents to serving the USENIX Association and the advanced computing community.

THANKS TO OUR VOLUNTEERS

Ellie Young, Executive Director

As many of our members know, USENIX's success is attributable to a large number of volunteers, who lend their expertise and support for our conferences, publications, good works, and member services. They work closely with our staff in bringing you the best there is in the fields of systems research and system administration. Many of you have participated on program committees, steering committees, and subcommittees, as well as contributing to this magazine. We are most grateful to you all. I would like to make special mention of the following individuals who made significant contributions in 2009.

The Program Chairs:

Margo Seltzer and Ric Wheeler: 7th USENIX Conference on File and Storage Technologies (FAST '09)

James Cheney: First Workshop on the Theory and Practice of Provenance (TaPP '09)

Alexandra Federova and Jim Larus: First USENIX Workshop on Hot Topics in Parallelism (HotPar '09)

Rodrigo Rodrigues and Keith Ross: 8th International Workshop on Peer-to-Peer Systems (IPTPS '09)

Wenke Lee: 2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '09)

Jennifer Rexford and Emin Gün Sirer: 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI '09)

Armando Fox: 12th Workshop on Hot Topics in Operating Systems (HotOS XII)

Geoffrey M. Voelker and Alec Wolman: 2009 USENIX Annual Technical Conference

Wenke Lee: 2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET '09)

Sambit Sahu and Prashant Shenoy: Workshop on Hot Topics in Cloud Computing (HotCloud '09)

Jelena Mirkovic and Angelos Stavrou: 2nd Workshop on Cyber Security Experimentation and Test (CSET '09)

Dan Boneh and Alexander Sotirov: 3rd USENIX Workshop on Offensive Technologies (WOOT '09)

David Jefferson, Joseph Lorenzo Hall, and Tal Moran: 2009 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE '09)

Tadayoshi Kohno: 4th USENIX Workshop on Hot Topics in Security (HotSec '09)

Jennifer Bayuk: Fourth Workshop on Security Metrics (MetriCon 4.0)

Fabian Monrose: 18th USENIX Security Symposium (Security '09)

Adam Moskowitz: 23rd Large Installation System Administration Conference (LISA '09)

The Invited Talks/Special Track Chairs:

Erik Riedel: Tutorials at FAST

Geoff Kuenning: WiPs and Posters at FAST

Michael Freedman: Posters at NSDI

George Candea and Andrew Warfield: Posters at USENIX Annual Tech

Dan Boneh and Patrick McDaniel: Invited Talks at USENIX Security

Carrie Gates: Posters at USENIX Security

Sven Dietrich: WiPs at USENIX Security

Doug Hughes and Amy Rich: Invited Talks at LISA

Lee Damon: Workshops at LISA

John "Rowan" Littell: Guru Is In sessions at LISA

Gautam Singaraju: WiPS and Posters at LISA

Other Major Contributors:

Alva Couch for liaising with the Computing Research Association, with LOPSA, and with VEE, CHIMIT, and HotAC workshops co-sponsored by USENIX

Matt Blaze, Gerald Carter, Clem Cole, Alva Couch, Rémy Evard, Brian Noble, Niels Provos, and Margo Seltzer for their service on the USENIX Board of Directors

Rémy Evard for chairing the USENIX Nominating Committee for the 2010 Election for the Board of Directors

Jeff Bates, Steven Bourne, Clem Cole, John Gilmore, Timothy Lord, Jim McGinness, Keith Packard, and Niels Provos for serving on the USENIX Awards Committee

Paul Anderson, Mark Burgess, Bill LeFebvre, and Amy Rich for serving on the SAGE Awards Committee

Rob Kolstad and Don Piele for their work with the USA Computing Olympiad, co-sponsored by USENIX

Dan Geer, Theodore Ts'o, and Brian Noble for serving on the Audit Committee

Jacob Farmer of Cambridge Computing for his sponsorship of the "USENIX Education on the Road" series and for organizing the Storage Pavilion and Data Storage Day at LISA

CORRIGENDUM

In the April 2009 issue of *;login:*, an error was introduced on p. 5, the first page of Mark Burgess's article "The Cloud Minders." Paragraph 3, sentence 3 has been corrected in the online version to read as follows:

But perhaps you are thinking, "The gentleman doth protest too much": We are all stuck in the same mess, cheered on by broken funding politics and commercial exuberance; after all, this is only a sign that information technology has truly entered the marketplace.

conference reports

THANKS TO OUR SUMMARIZERS

LISA '09: 23rd Large Installation System Administration Conference 100

Mark Burgess
Leah Cardaci
Rik Farrow
John Hewson
Rowan Littell
David Plonka
Shaya Potter
Andrew Seely
Josh Simon
Chuan Yue

SPECIAL AWARDS AT LISA '09

David Blank-Edelman received the 2009 SAGE Outstanding Achievement Award for his many contributions to the sysadmin community over the past quarter of a century. See http://www.sage.org/about/outstanding.html.

Luke S. Crawford won the Chuck Yerkes Award in recognition of outstanding individual contributions in online forums over the past year. See http://lopsa.org/node/1858.

LISA '09: 23rd Large Installation System Administration Conference

Sponsored by USENIX and SAGE in cooperation with LOPSA and SNIA

Baltimore, MD
November 1–6, 2009

Videos, MP3s, and PDFs of presentations are available online at http://www.usenix.org/lisa09/.

KEYNOTE ADDRESS

- ***Ahead in the Cloud: The Power of Infrastructure as a Service***
  Werner Vogels, CTO, Amazon.com

  Summarized by David Plonka (plonka@cs.wisc.edu)

Werner Vogels, who sometimes describes himself as the "system administrator of a large book shop," gave this year's LISA keynote address, an overview of Amazon's Elastic Cloud Computing (EC2) as the preeminent example of infrastructure provided as a service.

First, Werner introduced an exemplary customer of Amazon's services—"animoto" (http://animoto.com/), a company that developed an engine that automatically combines images, music, and video clips that you fashion into a polished production, something like a movie trailer. He said they are a New York-based enterprise that owns no servers. Instead, their product uses a four-step process (upload, analysis, rendering, and distribution) that employs Amazon cloud services: Amazon SQS to manage a queue of jobs, and Amazon EC2 to create and S3 to store content. When animoto launched a Facebook application for the service, they were able to immediately employ thousands of servers via Amazon EC2 servers to handle the influx of new users. He made the point that such a venture is revolutionary: you couldn't secure start-up funding for 5000 or more servers simply to launch a free Facebook application. Thus, Werner describes this as a "democratization of business": essentially, that the little guy can get the resources to launch something great.

Werner proceeded to describe how, in general, Amazon provides infrastructure as a service and that this is a significant foundation of Amazon's structure, such that Amazon's Web business, is a customer of this service too. Turning his comments to system administration, specifically, Werner said it is a myth that cloud computing puts sysadmins out of work. Indeed, sysadmins should have cloud computing in their portfolio, so that you can shift things there if and when you want. Since running an application on the cloud entails automation, cloud computing allows you to introduce more automation.

Werner prefers to use the term "infrastructure as a service" to "cloud computing" to disambiguate the concept from many things that are being called cloud computing, some of which go horribly wrong. The Gartner group's

definition of cloud computing is computing that is massively scalable, a service, over the Internet, and with multiple external customers. Werner added that infrastructure as a service needs two more characteristics: (1) resources available and releasable on demand (saving money and management troubles) and (2) the ability to pay as you go (so that there is not a massive expense when you are not using it).

Next, Werner described the implementation and evolution of each component of Amazon's infrastructure as a service to meet both Amazon's internal needs and those of its customers. The general infrastructure for Elastic Cloud Computing (EC2) runs on equipment managed using the Xen hypervisor. The storage offerings include Simple Storage Service (S3) for storage, SimpleDB for single index table DB operations, Elastic Block Storage (EBS) for operations requiring raw device access, and Relational Database Service (RDS) for custom MySQL instances that the customer can tune. Other offerings such as the Amazon Virtual Private Cloud (VPC) allow a customer to create a sort of "walled garden" using their own network addressing. Lastly, he mentioned that they've introduced "reserved instances" for customers wanting 24/7 use.

In closing, Werner outlined current usage trends for such cloud services, including load testing, collaborations, disaster recovery testing, large-scale analysis, marketing campaigns, and High Performance Computing (HPC). Amazon's cloud customers include Forbes for its stock quote streaming application, periodic streaming video for Indianapolis Motor Speedway events, Playfish for social network scaling, eHarmony to perform map/reduce matching daily, Netflix for video on demand, and Intuit for load testing to prepare for TurboTax downloads.

The session closed with questions from the audience. Rik Farrow asked, "What if MySQL goes away, given that it is the basis of Amazon's RDS service?" Werner replied that it is out of their control, but that Larry Ellison, CEO of Oracle Corp., the owner of MySQL, has committed to maintaining MySQL as a separate product. Brent Chapman asked about provisioning networks, and Werner replied that they don't currently expose this, even to allow customers to assign IP addresses, but there are a couple of tools for load balancing. Furthermore, each virtual machine runs its own firewall, and the customer doesn't have configurable control over all the network pieces. How does sensitive data (e.g., health care information) get handled in the cloud? Such security is an end-to-end issue and, generally, encryption is required to guarantee that data in the cloud can never be read by anyone else; there are strategies that involve using non-sensitive metadata as indexes to locate encrypted data. He also noted that some customers have achieved HIPAA and other regulatory compliance levels and that an Amazon white paper is available about HIPAA certification on cloud computing.

Contact information for Werner Vogels can be found at http://mynameise.com/werner.

THE HUMAN SIDE OF SYSADMIN

Summarized by Shaya Potter (spotter@cs.columbia.edu)

- ***Pushing Boulders Uphill: The Difficulty of Network Intrusion Recovery***

  Michael E. Locasto, George Mason University; Matthew Burnside, Columbia University; Darrell Bethea, University of North Carolina at Chapel Hill

Michael Locasto focused on experience gained and lessons learned from analyzing a large-scale intrusion event that occurred at a large research university. This problem of network intrusion recovery "is a particularly thorny exercise in researching, designing, and creating usable security mechanisms," because intrusion recovery is an art, not a science. Each attack is different and each place an attack is successful is different. This means that each attack has to be handled differently. One thing that can help is to know more cases where intrusion occurs on a large scale to learn from those examples. However, there's a large stigma to admitting that a breach has occurred, meaning there are fewer stories to learn from. Michael hopes his paper can contribute to the lore and add to the publicized experience.

While Michael's paper included three intrusion recovery stories, his talks focused on the one that occurred in December 2007. Early in the year, a research lab at the university received new machines with high performance NVIDIA cards. As no official Linux drivers existed for them at the time, the lab used unofficial drivers. On December 6, the machines started crashing regularly. Installing the new official drivers did not help. They then pushed all the updates to the new machines via rdist, including the latest kernel, but this did not help either. It was assumed that there was a problem with the machines. However, on December 13, the rdist machines started crashing as well, and when they attempted to recompile its kernel, mkdir returned an error for directories containing only numbers. This led them to believe the machines had a rootkit, which they were able to confirm.

So what did they learn from this? First, they only discovered the intrusion by accident. Until a conflict cropped up and machines started crashing, they had no idea the machines had been exploited. Second, computer forensics is difficult to achieve. There is a tension between disabling an exploited host and keeping it up, considering, on the one hand, the impact on the reputation of those who host the machine as well as the risk to confidentiality, integrity, and privacy of the data on it versus the desire, on the other hand, to observe what is going on to learn more about it, as well as to keep available the service provided by the machine. Third, institutional memory is weak, meaning that goals can be forgotten or misunderstood, causing security gaps after other repairs are performed. Fourth, in many cases, informal preferences seemed to have a large impact on how to proceed in handling the intrusion. Finally, and most importantly, improvisation was common, as good tools

did not exist to handle the challenges faced. This includes engineering challenges of repairing a network as well as management and usability challenges of dealing with people.

- ***Two-Person Control Administration: Preventing Administration Faults through Duplication***
  Shaya Potter, Steven M. Bellovin, and Jason Nieh, Columbia University

Shaya Potter focused on how one can apply the two-person control model to system administration. This is to help with administration faults that occur due to the fact that machines are complicated and complicated systems are prone to faults. Shaya focused his talk on two types of administrative faults that can occur, accidental and malicious. Accidental faults can be viewed as misconfigurations, while malicious faults are the result of an administrator leveraging privilege for malicious means.

To help prevent these type of faults from entering the system, Shaya proposed that the two-person control model should be applied to system administration, as this is known to be a good way to prevent faults in real systems, such as with nuclear weapons, bank checks, and pilots, all of which require two people to perform the same actions or just be available to observe the actions. To implement this, the authors created the I See Everything Twice (ISE-T) system to provide a general means of branching two computing environments and then comparing how they differ after they execute independently. This can be applied to system administration by branching the environments, allowing two system administrators to work independently and then compare their results for equivalence. If the environments are equivalent, the changes can be applied to the base system. Shaya noted that this can be an expensive solution, too expensive in many scenarios. However, he believes that the system can be used to improve other areas of system administration with little added cost, including the training of junior system administrators, providing an audit trail as well as combining it with configuration management.

ISE-T is implemented as a combination of three components. First, it includes an isolation mechanism that can be based on operating system containers, such as Solaris Zones, Linux Containers, or BSD's jails, as well as virtual machines such as VMware. These provide isolation for the cloned environment. Second, it includes a file system that both can branch itself quickly and can isolate the changes that occur after branch occurs, so that one can easily determine what has changed. In order to satisfy these requirements, they leveraged unioning file systems in order to enable quick branching, as well as isolating the changes to a new branch of the union. Finally, ISE-T includes a system service that compares the two environments for equivalence. This is difficult, because if one is limited to exact binaries, one will miss files that are semantically the same but differ in small ways, such as white space in scripts. However, for their prototype, they basically stuck to requiring that the files having the same binary content.

The authors created a prototype which they used to test the feasibility of capturing changes and comparing for equivalence over a number of administrative tasks, including installing software, upgrading systems and making configuration changes, as well as having people create back doors. In general, the configurations were equivalent, except in places where they expected differences to occur, such as with the creation of encryption keys and back doors. In the few cases where other differences were detected, ISE-T was able to clearly show differences between the two administrations.

Shaya was asked about how this compares to using a source code control system for configuration management systems. He answered that this formalizes and enforces behaviors that might not exist with a source code control system.

- ***The Water Fountain vs. the Fire Hose: An Examination and Comparison of Two Large Enterprise Mail Service Migrations***
  Craig Stacey, Max Trefonides, Tim Kendall, and Brian Finley, Argonne National Laboratory

Craig Stacey spoke about his worst week as a system administrator. Argonne used to have a simple mail infrastructure. Mail came in and was handed off to a mail server using a system that lasted from 1998 to 2008, running an old version of Cyrus IMAP. The system worked very well until 2006, when they thought about replacing it, but other things took time away. Later on they piloted Zimbra to provide calendaring to the lab. Then they decided that as Zimbra can provide mail as well, they should consolidate everything onto it. They felt the deadline for moving was far away and wouldn't be hard to hit, as IMAP is simple: it's just mailbox data.

Plan A was to use imapsync to move data between the machines. However, the IMAP server was old. Its libraries were too old, and it was so overtaxed that it couldn't handle the load and they feared it was going to fall over. So they set up a separate machine to pull everything off. However, it turned out that they could only run two syncs in parallel without affecting service, which was very slow and wouldn't finish in time.

Plan B was to use rsync to sync the machines and then scripts could import the changes into Zimbra, and imapsync could then sync the mailbox flags between the two machines. However, the implementation did not hold up. The server they were trying to sync the mailboxes to was too slow, due to NFS problems. And there were many namespace collisions in Zimbra, due to its flat namespace structure, but they learned a lot in their test system.

In the end they had to start the real migration two weeks before the switchover date in April. Things weren't ready when the switchover came, but they threw the switch anyway to deliver to the new mail server while the sync was continuing. In the end, they created a new mailbox for each user and showed users how to access both mailboxes and move what mail they needed from one server to another if it was needed.

Craig noted that hindsight is 20-20 and that they could have done it more slowly with more testing, but that they were really afraid the system was going to fall down. Leveraging users to do the migration is useful, as not everything needs to migrate and the individual users know what should and shouldn't be migrated. The most important lesson is that one shouldn't get complacent when a system seems to just barely work, for if you don't keep it up to date on hardware and software, it will get to the point where you can't make any changes to it without fear that you'll break the system.

INVITED TALK

- ***How to Build a PB Sized Disk Storage System***
  Raymond L. Paden, HPC Technical Architect, IBM Deep Computing

  Summarized by John Hewson (john.hewson@ed.ac.uk)

Petabyte-sized disk storage systems seemed unthinkable, but today they are increasingly common. Vendors are responding by making larger, less expensive disks and controllers that manage over a PB of data. Yet datacenter managers and application developers do not truly appreciate the complexity of these large storage systems. Too often they approach them as being peripheral rather than integral to the overall processing system. They give detailed attention to other things, but simply ask for a storage system like the ones they had before, only bigger. This often leads to an overly costly storage system that is a performance bottleneck. This talk will help you avoid the pitfalls and develop an efficient PB-sized system.

Paden talked about the issues encountered when building a petabyte (PB) sized storage system, and the problem that existing paradigms do not scale to the new sizes required. The key message of the talk was that to implement a PB storage system a number of questions need to be answered first: what is the I/O profile? Is a LAN or a SAN used? Can NFS or CIFS be used, or is a specialized file system needed?

Paden said that it is necessary to understand the profile of the system's users and their working set: what is the optimum working set size and cache locality? Temporal and spatial locality were examined, as were the implications of the storage access patterns of the users, highlighting differences such as streaming large files, small transactions for I/O processing, and transaction processing with temporal locality. Paden concluded that most environments have a mixed usage pattern and that it is best practice to develop use cases prior to making a purchase. Use cases provide a model of realistic use and require more time to evaluate than simple performance benchmarks; however, they provide a less synthetic method of evaluation.

Paden introduced the building-block strategy, where the smallest common storage unit consisting of servers, controllers, and disks is defined. This block is then used to construct a datacenter, ensuring that each device in the building block is balanced and optimized. Large building blocks are recommended for PB-scale systems.

The limitations of SAN file systems were mentioned, with LAN-based systems recommended as a cost-effective alternative. Paden described a taxonomy of file systems commonly used with clusters: conventional I/O, asynchronous I/O, networked file systems, network attached storage, basic clustered file systems, SAN file systems, multi-component clustered file systems, and high-level parallel I/O. Choice of the correct file system depends on the user profile.

Management of risk was briefly outlined, including disaster recovery and avoidance of single points of failure, such as the use of RAID 6 with SATA disks. The question and answer section identified managing the large number of cost-effective disks as a key issue for the future.

Doug Hughes, the Program Chair, asked if Paden had played with Lustre. Paden said he had not but Hughes was the second person to mention it. Hughes described Lustre as easy to install, reliable, and fast. Someone asked about flash drives and SSD. Paden replied that he was under non-disclosure but could say that these technologies are undergoing a period of rapid flux. Someone asked about the next big challenge to building file systems: It is managing the number of moving parts and having the tools to do so. The problems are "Texas big." As the session closed, people queued up to ask more questions.

INVITED TALK

- ***Eucalyptus: An Open Source Infrastructure for Cloud Computing***
  Rich Wolski, Chief Technology Officer and co-founder of Eucalyptus Systems Inc. and Professor of Computer Science at the University of California, Santa Barbara

  Summarized by Chuan Yue (cyue@cs.wm.edu)

Professor Rich Wolski gave a wonderful talk on Eucalyptus, an open source infrastructure for cloud computing. Wolski first mentioned that cloud computing is a reality. People not only talk about it, but also spend money on it. Many companies, such as Amazon and Google, have tremendously powerful infrastructures in place today. Using Amazon as an example, Wolski pointed out that two important features make cloud computing work. One is SLA (service level agreement), which tells you what level of services you will get when you purchase. The other is the transaction, which has really driven the interest in cloud computing.

Wolski then explained why they decided to build Eucalyptus. People are using public clouds daily, but what really happens inside the public clouds is not transparent. There absolutely are many issues that distributed computing researchers should think about. Therefore there should be an open source infrastructure that allows people to experiment, play with, extend, and understand cloud computing. This infrastructure should not be a refactorization of previously developed technology: "It has to be developed from

the first principle to be a cloud." This infrastructure should be open source so that its openness and exposure to the community can drive the technology forward.

Wolski emphasized that they borrowed something from the idea of the Turing test to make Eucalyptus a real cloud. What they did is to emulate an existing cloud—Amazon AWS—and support its APIs to the point that a user or a program cannot tell the difference between Amazon AWS and Eucalyptus. They built Eucalyptus to foster greater understanding of cloud computing and provide a technical overview platform for public clouds. But Wolski emphasized that Eucalyptus was not built to compete with or replace Amazon AWS or any other public cloud service. Eucalyptus needs to correctly implement Amazon's cloud abstractions and semantics, which are defined for scale. Eucalyptus also needs to be simple, scalable, system portable, and configurable.

The Eucalyptus infrastructure they have built is a bunch of open source Web service components that have been stuck together to make a cloud out of whatever resource users have. Eucalyptus has a layered architecture. The top layer is the translator that renders various client-side APIs into the internal representation of the cloud abstractions. The middle layer includes the VM controller and the storage management service. The bottom layer is the resource management.

After describing what they have done, Wolski went on to share what they have learned from building Eucalyptus. First, the notion of private clouds does not really exist; almost all the deployed private clouds are hybrid clouds. Second, storage architecture is hard to change in a company, because institutional knowledge and policy are embedded in the storage architecture. Third, cloud federation is a policy mediation problem. Last, a really new thing in cloud computing is that an application can measure the dollar cost associated with its execution. Wolski also argued against two myths of cloud computing: "Cloud computing is nothing more than virtualization" and "The cloud is grid." Using their careful performance comparison results, Wolski demonstrated that the third myth, "Cloud imposes a significant performance penalty," is also not true.

Finally, Wolski showed that Eucalyptus has been downloaded over 55,000 times from all around the world, and there are many real systems running on top of Eucalyptus. Wolski also showed their open source distribution effort and roadmap.

An audience member asked why open source is that important in Eucalyptus, and yet no open standard has been developed around it. Wolski replied that they say open source is important not only because they personally have benefited from open source, but also because they really think innovation will come from open source. It is a mistake to standardize too early; whether it is worth it to standardize should depend on users' needs. What are the challenges with eventual consistency in cloud computing? It is tricky but possible to put an SLA on the consistency semantics, and there is a tradeoff between this SLA and the scale of your applications. Is SLA supported in Eucalyptus? Not yet, but they certainly plan to support it.

INVITED TALK

- ***The Advanced Persistent Threat***

  Michael K. Daly, Director of Enterprise Security Services, Raytheon Company

  Summarized by Rik Farrow

Daly made a no-nonsense presentation from the perspective of a security officer of a defense contractor, something that he told us would have been impossible just a few years ago. What's changed is the threat landscape, as well as how large US companies now feel about sharing security information. Part of the sea change includes the involvement of nation states in espionage and attacks.

Daly launched into a description of the Advanced Persistent Threat (APT) by describing a scenario in which a malware-infected PDF file is downloaded from the USENIX Web site. The file contains a trojan that upon opening installs malware that begins beaconing back, using an HTTP request. Once acknowledged, the beacon packets become rare, perhaps once a day, once a week, or even once every six months. Eventually, the attacker uses the beacon to connect to your infected system and start using your machine. The malware includes download capability, so it can update itself. The malware can do anything you can do, as well as use your system as a hop site (relay), a malware repository, an infection site, etc. For 2009, using data from F-Secure, Daly broke down the exploited software as: Acrobat 48.87%, Word 39.22%, Excel 7.3%, PowerPoint 4.52%. I personally thought that Flash belongs in the list as well, and Daly does mention Flash a bit later in his talk, with the use of zeroday exploits using Flash in Firefox.

Gh0stnet was a good example of APT, targeted specifically at the Dalai Lama and other Tibet-related groups, where 1300 systems were remotely controlled. Daly noted that he was not picking on China, but using a public domain report. You can find public documents about Computer Network Attack (CNA) and Integrated Network-Electronic Warfare (INEW) in China, and it's expected that this is going on in most other countries as well.

Daly provided a bullet list of things you can do to protect your networks:

Focus on traffic in and out of your networks, using network analysis tools; initiate awareness training, including targeted training for specific people; compartmentalize your environment and watch inputs and outputs via the network; drive down the dwell time, that is, the amount of time between malware installation and discovery; share and collaborate, working with other groups.

Daly pointed out that dynamic DNS gets used a lot for malicious purposes and that every site on the Russian Business Network can be considered bad. He's seen DNS used as a covert channel. Raytheon has blocked some popular attachment types, such as .zip, and people get used to it.

He also suggested looking for regularity in beacon packets, or for User-Agent strings in HTTP requests that have subtle changes, by using Pareto searches. Daly mentioned using Web proxy server logs to uncover systems that have visited malware download sites.

Rik Farrow suggested that dangerous apps only be run in sandbox environments, and Daly said they were working on that now. Tom Limoncelli worried about working with China, as Google does. Daly recommended segmentation, that is, not opening everything up to untrusted partners. They also have a clean laptop loaner program for visits to other countries, and they set up special Web-based email accounts for trips. If somebody demands your password, you can give it to them, because it won't work from the outside. Another person said that if the US government makes us take off our shoes and toss our water bottles before we can fly, they should be able to outlaw software that is too complex to run safely. Daly replied that people want cool features and that the government tries to keep costs down by using COTS software. Carson Gaspar asked if they are seeing obfuscated email and attachments, and Daly said they are even seeing steganography. Julie Bauer of AMD wondered how she could find out more about travel issues, and Daly suggested the US State Department Web site.

PLENARY SESSION

- ***Google Wave***

  Daniel Berlin and Joe Gregorio, Google, Inc.

  Summarized by Chuan Yue (cyue@cs.wm.edu)

Joe Gregorio explained that Google Wave is a collaborative communication environment. They started from the question: "What would email look like if it were invented today?" Unlike email messages that are sent back and forth between users, every conversation in Google Wave is a collaboratively edited document. People involved in a conversation can add information to the document and see how this document evolves. The user interface of Google Wave is similar to Gmail and it has inbox, contacts, waves, and controls. Google Wave supports various content such as text, markup, image, and gadget, and it is more than just a collaborative text editing environment.

Gregorio differentiated the Google Wave product from underlying wave technology: "Wave is to Google Wave as email is to Gmail." The important part is federation, which is a technology that enables different wave providers to share waves with each other. Google Wave Federation Protocol (http://waveprotocol.org/) is open and iterating. It includes specifications and white papers. Federation is important both in avoiding fragmentation and in encouraging adoption. For example, Novell Pulse (http://www.novell.com/products/pulse/) has adopted the Google Wave Federation Protocol and can work seamlessly with Google Wave.

Going into the details of Google Wave, Gregorio described the wave data model. A wave is the container of wavelets, while each wavelet is actually a conversation. A wavelet contains a list of the conversation participants and a set of XML(ish) documents and annotations. A document is a linear sequence of items. It looks like an XML document but is not, because element and attribute names are well beyond what XML allows. Annotations are associated with the items in a document to provide various functionalities. The wavelet is the unit of concurrency. The wavelet is also the unit of federation.

Gregorio explained that in Google Wave, federation means sharing wavelets. Federation begins when a user adds someone outside the user's domain to a conversation. Wave servers run operational transforms to share the updates of the wavelets. Operational transforms incorporate operations made to a wavelet and then send transformed operations to each user so that all the users can end up with the same shared state. The Google Wave federation system sits on top of the XMPP technology: a wave server is an XMPP server, and federation is a service inside the XMPP server.

Gregorio mentioned that, so far, they have published two draft specifications: Google Wave Federation Protocol and Google Wave Conversation Model. A Java open source implementation of the specifications is also available (http://code.google.com/p/wave-protocol/). Gregorio showed that they have opened up a federation port on http://wavesandbox.com/, but it is still highly experimental. Gregorio also gave a few demos to illustrate the previously introduced concepts such as conversation, wavelet, and document. Finally, Gregorio said that they will open up an open federation port on http://wave.google.com/ once the specifications become stable; meanwhile, he mentioned that Google wants more people to participate in this project.

Someone asked whether they have explored the idea of making Google Wave completely peer-to-peer rather than using a client-server model. Gregorio replied that he was not involved in the early discussion of Google Wave, so he has no idea whether there is a discussion about peer-to-peer. But he argued that there has to be a centralized server to put together all the changes from the clients and then ask the clients to apply the changes before sending an update. Ari Redkin of UC Berkeley asked where the certificates used in Google Wave come from. Wave servers signed the certificates. Therefore, when a client tells the server to make changes to a document, the server knows where the request comes from. Gregorio acknowledged that currently a client is tied to a particular wave server because the server has the private key of the client. Has there been any thought or discussion about dealing with spam, malware, and virus in Google Wave? Certainly work needs to be done, and a white

paper will come out soon talking about that. In Google Wave, at least all the operations are signed.

PLENARY SESSION

- ***Cosmic Computing: Supporting the Science of the Planck Space Based Telescope***

  Shane Canon, Group Leader, Data System Group in NERSC, Lawrence Berkeley National Laboratory

  Summarized by Leah Cardaci (lcardaci@cs.iupui.edu)

Shane Cannon discussed the Planck project and NERSC's work supporting research on the resulting data. Cannon began with a disclaimer that he was not an astrophysicist, cosmologist, or rocket scientist.

Cannon first presented an overview of the science of the cosmic microwave background (CMB) and why it is studied. The CMB is one of the first observable conditions from the Big Bang. Details about it can provide information about the Big Bang, the geometry, composition and shape of the universe, and the dynamics involved in the evolution of the cosmos. It was accidentally discovered in 1965 by Penzias and Wilson, physicists working for Bell Labs looking at types of background signals. They noticed an independent background signal that they could not eliminate. Eventually,they realized the signal was coming from space. In 1978, the pair won a Nobel Prize for their discovery. Since its initial discovery, the CMB has been studied in ground-based, balloon-based, and space-based projects.

Planck, a joint mission of the ESA and NASA, is a new space-based project that will look at fluctuations in the CMB to study the Big Bang and some basic properties of the universe. It will provide the biggest datasets for the CMB to date, with 10^{12} observations. The Planck satellite has both a low frequency and a high frequency bank of instruments. Advanced cryogenic shielding is needed to control the heat generated by the instruments. The ESA launched the Planck satellite on May 14, 2009. It is in orbit around the second Lagrangian Point (L2), a stable point relative to the Earth and the Sun. The results from Planck will allow for a more detailed map of the CMB to be built. As the dataset has grown, the analysis has had to move to more iterative methods that scale in a more sustainable manner.

The data is beamed from Planck to various ground stations. After some initial analysis, the low and the high frequency data are split off to separate processing groups. At this point, that data is sent to various places, including NASA's IPAC. NERSC is not a formal part of this pipeline but supports a lot of the analysis and handles both the high and low frequency datasets. CMB data is analyzed with time order data that is used to generate maps. Analysis of the CMB data is primarily concerned with how to remove the noise present in the measurements.

NERSC is the flagship computing center for the Department of Energy's Office of Science. It began in 1974 at Lawrence Livermore and moved to Berkeley Lab in 1996. It serves a large population of diverse interests. The center focuses on high end computing, storage, and networking. Because of the variety of groups using the resources, NERSC systems need to be flexible to meet multiple clients' needs. There are several groups of systems involved in this support. The flagship system is Franklin, a Cray XT4 massively parallel processing system with 9,740 nodes. In addition, a new Cray-based system is presently being built. NERSC also has some smaller clusters, a 400 TB global file system, and an archival storage system. The GPFS-based global file system was created to permit clients to avoid the burden of moving data to various systems but still allow high-performance access. It is a large SAN that connects to the various NERSC systems in a variety of optimized methods.

NERSC is supporting a variety of projects involving big datasets. Some projects are beginning to require data handling that almost outpaces the system's current abilities. Recent projects are expected to generate petabyte datasets. In addition, these datasets will be analyzed long after the original observations, creating a need to preserve the data for the long term. Cannon provided a select list of example big data projects at NERSC, pointing out that some new projects may require more high performance data support and not as much high performance computation support. KAMLAND is a neutrino detector experiment that has generated 0.6 TB of data in six years. ALICE is a soon-to-deploy collider experiment that is looking at QCD (quantum chromodynamics) matter. It is expected to generate around 600 TB of data in the first year, with a long-term estimate of 3.8 PB of data. Other future data-intensive projects include trying to build a model of global climate from 1871 to the present, building cloud resolving climate models, and the Joint Genome Institute looking at microbial DNA.

Cannon finished by relating general observations about dealing with data-intensive projects and handling large datasets. Successful projects use a shared data model, employ parallelism in multiple levels of the process, design to deal with failure, avoid the I/O bottleneck whenever possible, and consider the true lifecycle of the data. Technologies that enable such projects include scalable archive systems, parallel file systems, data management middleware, and visualization technology. Challenges include the continued imbalance between capacity and bandwidth, the fact that common utilities are not designed with the new storage approaches in mind, the lack of improvement in bit error rates, and the need to deal with the new requirements for long-term storage of these large datasets.

What is the name of the new NERSC computing system? Cannon said it was named Hopper. Had NERSC made any customizations to GPFS for their global file system? They basically ran the system out of the box, but had collaborated with IBM to meet their needs. However, they have customized the Cray system.

INVITED TALK

- ***Storage and Analysis Infrastructure for Anton***
  Mark Moraes, Head of Anton Software Development and Systems Software, D.E. Shaw Research

  Summarized by David Plonka (plonka@cs.wisc.edu)

Mark Moraes introduced LISA attendees to the Anton supercomputer, a new machine meant to impact biology and chemistry by using advanced computational methods and hardware. First, to put their achievement in context, Mark introduced us to the fascinating world of computational chemistry and molecular dynamics (MD). MD involves simulating what molecules do, in the study of proteins, for example. Basically, they simulate the molecule surrounded by a cube of water to observe behaviors such as wiggling and folding that are key to the molecule's function. Simulation is an important technique because the alternative, experimental method, is difficult and involves error-prone purification and crystallization. The molecules themselves are complex: a modeled protein could become a 50,000 atom system when the water is modeled explicitly. To understand further why simulation on biological timescales is hard, he informed us that most organic molecules are held together by bonds that vibrate on the femtosecond (10^{-15} of a second) timescale, while other important behaviors happen on the timescale of milliseconds.

Through the use of new MD algorithms that could scale to a large number of interconnected cores, they saw the possibility of running simulations 100 or 1000 times faster than previously feasible. However, modern commodity CPUs have limitations as a building block for such a system, since not much of the chip area is devoted to computation (lots of it is cache instead). Since MD computations involves a relatively small number (tens of thousands) of atoms, D.E. Shaw Research decided it was worth building a new machine based on custom-designed ASIC-based chips; while the resulting supercomputer may be less flexible, it would be dramatically faster for MD applications.

Thus Anton was born, coming online last year; it is used to study molecules, small proteins, and DNA via simulation. Its performance, as measured by a comparative benchmark, shows that it dramatically outperforms its predecessor, Desmond. For instance a 1024-core, 512-node cluster running Desmond might achieve about 500 nanoseconds of simulated behavior per day, whereas the Anton supercomputer, with its 512 ASICs, can achieve about 16,000 nanoseconds (16ms) of simulated behavior per day. Such dramatic performance gains in simulation are expected to change the way research chemists do their work. Indeed, in response to Anton's performance, one chemist remarked, "I'm going to have to think faster!"

Continuing with a system administration–specific portion of this talk, Mark described the significant infrastructure that supports Anton's supercomputing capability. For control and management, their front-end machine uses Linux running CentOS 4.x, and software such as syslog, ganglia, PostgreSQL, dhcpd, and tftp. There are also some custom Anton management components that employ JSON-RPC and the slurmd job queue system. The I/O and storage subsystem relies on NFS with the data resulting from runs accumulating into the hundreds of gigabytes to a terabyte per day. A custom file-based organization avoids having to search unnecessarily for data in volumes consisting of thousands of terabytes of storage. The parallel analysis portion of their infrastructure is a framework that they developed called HiMach; it is inspired by Google's MapReduce but is specific to the MD data structures they use. There are also Linux-based control processors on boards within the machine, which assist in monitoring and managing the supercomputer's custom ASIC processors.

Mark wrapped up his presentation with a fascinating animated movie clip, based on Anton simulation, of the folding of the villin protein headpiece. These and other sample animations dramatically show the behaviors of these microscopic structures at fine timescales, and researchers appear to be about to arrive at new scientific results that seemed impossible with the prior slower simulations. Thus, it appears that this new instrument, Anton, will allow researchers to arrive at useful results much more quickly and, in some cases, arrive at results that no one had the patience to develop by simulation before.

At the close of the session Mark fielded questions from the audience: How programmable are the ASICs employed in Anton? There is a flexible component, coded in C, that is often changed, especially for force fields; about half of the ASICs (12 cores) can be changed. The pipelines also have tables that allow their function to be changed somewhat. Is visualization performed in real time during a run? They can do this, but typically just a little bit of visualization is used during the run to determine if it's working correctly. Are custom or standard compilers employed for the programmable cores? They license four general-purpose cores from Tensilica, which provides customized compilers for these cores. D.E. Shaw Research designed some other components themselves, with custom instructions, and they now have a gcc 4.3 port that generates pretty good code for them. Mark added that it takes a long time to port gcc, so if you're in this situation, you should start early. How does Anton differ from the MD-GRAPE supercomputer? MD-GRAPE involves pipelines, deals with distant forces in hardware, and ignores the Amdahl's Law bottleneck. Anton ties computation together end-to-end.

NETWORKS, NETWORKS, NETWORKS

Summarized by Rik Farrow

- ***Crossbow Virtual Wire: Network in a Box***
  Sunay Tripathi, Nicolas Droux, Kais Belgaied, and Shrikrishna Khare, Sun Microsystems, Inc.

  Awarded Best Paper!

Nicolas Droux explained that virtualized environments and services need to share physical network interfaces. In this eight-year-long project, the big focus was on performance and also being able to take advantage of hardware that has multiple rings, DMA channels, and classifiers. Security was also important, so there is real isolation between flows and no ability to sniff or inject traffic into another flow.

The project built on previous work, such as nTop, streams, and Nemo, but needed to improve on management, which had been difficult in the past. Crossbow uses the notion of Virtual NICs (VNICs). Each VNIC is assigned a hardware lane that may consist of NIC resources (like rings and channels), and this lane, and any threads and interrupts, gets bound to a specific CPU. This binding avoids context changes and improves cache coherency. To control bandwidth in a VNIC, interrupts can be disabled and replaced with pulling chains of packets. Priority Flow Control (PFC) allows the use of VNICs with services instead of just VMs. The VNICs themselves connect via virtual switches, and these switches and the use of VNICs allow the modeling of a physical network within a single Solaris (or OpenSolaris) system. Droux pointed out that a student could be sitting in a cafe with a network model on his laptop, designing the next routing protocol.

Tom Limoncelli (Google) asked the only question, wondering if the bandwidth limits were absolute or if they allowed bursts? Droux answered that they were looking at bandwidth guarantees instead of limits, and when they have that, bandwidth use would become more flexible. See Peter Galvin's column on p. 79 for more details about Crossbox.

- ***EVA: A Framework for Network Analysis and Risk Assessment***
  Melissa Danforth, California State University, Bakersfield

Melissa Danforth described EVA (Evolutionary Vulnerability Analyzer) as an attack graph tool that supports a multitude of analysis modes. Host-based vulnerability scans produce information that is limited to each host. With EVA, the user can start by imagining an attack that provides a foothold within a single system, and see ways that the attack may spread to other systems.

Danforth used a diagram, built with EVA, that showed two groups consisting of a total of six systems. Two Internet-facing systems provided the initial foothold, and via vulnerabilities found on four internal servers an attack would eventually produce privilege escalation to root on the internal servers. EVA does this by producing attack graphs, where the nodes represent systems and the edges represent successful exploits. Exploits can be chained together to form templates: for example, a template for a vulnerable SSH server that provides both the compromise and privilege escalation (to root).

Nessus is used to scan for host-level vulnerabilities, and several attacker models are used: for example, an insider or an external attacker with no privileges. The Java Expert System (JES) is used to create the graphs, although the process is not automatic and requires some user input. The tool has been used to encourage sysadmins to patch systems they have been ignoring by showing how failing to install a patch can lead to many systems being exploited. EVA can also be used for forensics by uncovering possibly exploited systems and for network design.

- ***An Analysis of Network Configuration Artifacts***
  David Plonka and Andres Jaan Tack, University of Wisconsin—Madison

David Plonka and his co-author had the good fortune to be sitting on top of a 10-year repository of network configuration changes. When his university began building out their network, they also began using RCS as a revision control system. In this paper, the authors mined this repository for insights.

David said that they borrowed heavily from the world of programming, where many studies of source code changes had already been done. First, they converted the RCS records into CVS so they could use tools such as statCVS-XML and cvs2cl during the analysis. Then they began to pry out details of who made changes, when they made them, the types of changes made, and how quickly new revisions were made.

The campus network has over 3800 devices, with many being access layer (switches and wireless). Web interfaces allow 300 authorized users to make some changes, and other smaller groups, about 64 people in total, have root access. Still, 75% of all changes were made by just five people, all part of the network engineers group. They also found that 90% of revisions were interface changes, something that could be done by authorized users, and that VLANs were another good target for their Web interface. They could also see that there were many short-lived revisions as people tried changes, then backed them out or changed them, apparently because they didn't work. David pointed out that network management is different from programming because you are working on live systems.

One person asked about the use of RANCID (Really Awesome New Cisco confIg Differ), as RANCID doesn't include comments or the author of changes. David said he only mentioned RANCID and preferred his own scripts. What is the difference between the campus info and info gathered from a provider network (large ISP)? The difference is that the campus has lots of access-level devices.

INVITED TALK

- ***Searching for Truth, or at Least Data: How to Be an Empiricist Skeptic***
  Elizabeth D. Zwicky

  Summarized by Leah Cardaci (lcardaci@cs.iupui.edu)

Elizabeth Zwicky opened the talk with a warning that all of the numbers in the talk were made up, but all of the stories were true. The talk was addressed to system administrators who look at information about technology. Skepticism towards data is a trait that good system administrators and good security people have in common. It involves the desire to learn about something, the ability to understand the difference between appearance and reality, and an ability to understand numbers.

As an example of the difference between appearance and reality, Zwicky related how a former coworker who knew how to pick locks was asked by the company CFO to break into her office. The coworker pointed out this wasn't really a lock-picking problem, popped out a raised floor tile, and used a coat hanger to pull the handle and open the door. While the office appeared to have a solid wall, it was easily bypassed.

Taking this approach to finding and considering data can prevent logistical nightmares, is part of troubleshooting and security, is fun, and prevents you from falling for pseudo-science. When looking at potential data, determine if it is really data, consider what it is data about, and ask what conclusion can be drawn from the data. Zwicky went through several examples of potential data, such as "Brand A's router has an error rate 200% worse than Brand B" and discussed the value of each example. Hearsay, numbers without context, and conclusions are not data. Observations, numbers with context, and self-reports are data.

Basic statistics can help one understand whether given numbers have appropriate contexts. When looking at averages, it is useful to know what kind of average is meant. One common average is the mean. Mean is interesting when discussing a bell curve graph that is fairly symmetric, but not with other distributions. Graphs related to machines do not usually have that shape, so mean is not a useful measurement. Other measurements such as the median, quartiles, and percentiles, or the entire graph are more relevant. If only a mean is available, looking at the standard deviation will show how the distribution is skewed.

When given a rate, asking "Compared to what?" can provide information with needed context. For example, a 200% increased risk of being hit by a meteor is still very low, because the initial odds are incredibly low. Correlation does not equal causation. Two correlated events can both be affected by other unmentioned factors. If someone is looking to make a point, they will likely be able to find some correlation to support the claim.

Zwicky showed an example comparing numbers of users versus network usage per month at an ISP. While the initial months appeared to be following a clear predictable curve, the seasonal activity surrounding Christmas caused a change in behavior. Two significantly different predicted curves for the activity show how people can interpret the same numbers in different ways. Without the appropriate context, it is not easy to know what the given data is really showing. Seeing data without knowing what the data is really about does not provide much information.

There are a variety of ways to gather data. You need a programming language to process the data, preferably one that manipulates text well. You need some tools to look at the internals of what is happening with programs and network traffic. Some examples of such tools are trace, dtrace, truss, wireshark, tcpdump, and Windows Sysinternals. Spreadsheet programs, GraphViz, and gnuplot can be used to make pictures of the data you gather. Some basic knowledge for handling data is basic statistics, SQL and XML, and writing regular expressions.

There are multiple ways to find data sources, including mining existing data or learning new data. For example, look at logfiles. If there are no existing sources, collect new data. Data can be collected with logging, tracing or sniffing, or running tests. Simulate data or extrapolate data from some known information. See if data can be gathered from published sources and colleagues. If all else fails, "make stuff up" by guessing. This process can be at least slightly improved by collecting a variety of guesses, basing guesses on known information, and testing various guesses. When collecting data about people, be prepared to go through a Human Subjects Board. It can be difficult to design an unbiased survey. In such a case, it may be better to gather descriptions rather than numerical measures.

Once you have the data, the interesting part may be obvious, but you will likely need to analyze it. Sometimes you need to check the data to be sure it makes sense and measures what you want to measure. Once you have the data it is important to know what questions you want to ask when analyzing the data. Humans are good at certain types of pattern recognition, such as recognizing abrupt change, noticing correlation, and seeing faces. They are not good at understanding probability, seeing when things aren't related, noticing slow change, and understanding a delayed correlation.

To show data well, know what the message is and limit any extra facts shown. To avoid lying with graphs, understand how people perceive graphs. Humans do not perceive area well. For this reason, pie charts are not an effective tool.

Zwicky gave a detailed example of the problem measuring performance of a help desk. Time to completion was not an effective metric, because it encouraged workers to prematurely label a job or try to hand off jobs to others. An alternative measure was a customer satisfaction survey. When

considering the results, it was important to think about the behavior of the people who filled out the form. Most would be people who felt strongly about the help desk service and not those with an average experience. As the majority of employees were not likely to follow instructions for no reason, the survey results were skewed. Zwicky showed a variety of different graphs of the survey results. Looking at the data in multiple ways showed information that was not visible in the most basic view of the results.

One person asked how to deal with management's insistence that a known bad metric is better than no metric at all. Zwicky suggested trying to replace the bad metric with a better metric. Making the new metric more appealing in some way can help.

SECURITY, SECURITY, SECURITY

Summarized by Shaya Potter (spotter@cs.columbia.edu)

- ***Secure Passwords Through Enhanced Hashing***
  Benjamin Strahs, Chuan Yue, and Haining Wang, The College of William and Mary

Chuan presented PasswordAgent, which is meant to provide secure passwords for Web site usage. This is important, as passwords are the most common online authentication method and will remain dominant for the foreseeable future. However, passwords are crackable if weak, and vulnerable to theft, especially as users use the same password at many sites.

Many approaches have been used to try and secure passwords, including password managers, but those lack mobility, and single sign-on systems, but these provide a single point of failure. Password hashing, taking a weak password and making it strong by hashing it with other data, is what the authors built on.

PasswordAgent is based on PwdHash, a Web browser-based solution that creates a unique password for each site based on a hash of the password and the domain name of the site being accessed as a salt. PwdHash is meant to focus on phishing attacks, since it would give a phishing site an incorrect password from a different domain name. Rather than using the domain name to hash together with the password as PwdHash does, PasswordAgent creates random salts that it hashes together with the password. It stores these salts in a salt repository, accessible from many different machines, and in multiple repositories so that one doesn't have a single point of failure. PasswordAgent is built as a Firefox extension and hooks into password fields, enabling the password to be replaced when the password is protected by PasswordAgent.

PasswordAgent protects passwords in many different scenarios. For instance, it doesn't have a master password to steal. Furthermore, even if the plaintext password passed to a Web site is compromised, the real password is still protected as long as the salt remains secure. It also reduces the risk of weak passwords, as hashing them together with the random salt increases their security. Even if an attacker could guess the user's password, they would have to iterate against every possible salt. Finally, PasswordAgent protects against phishing by notifying users when they enter their passwords into sites that have not been set up to be protected by PasswordAgent. If the user expected this site to be protected, it's indicative that this is a phishing site.

Limitations of PasswordAgent include vulnerability to malware on the system that can see the salts as well as the passwords before they are hashed, as well as its dependence on the salt repository. If the repository becomes unavailable, one will not be able to create the hashes.

What happens if domain names change, such as one Web site being purchased by another company? The password would have to be manually reset, which would generate a new salt.

- ***SEEdit: SELinux Security Policy Configuration System with Higher Level Language***
  Yuichi Nakamura and Yoshiki Sameshima, Hitachi Software Engineering Co., Ltd.; Toshihiro Tabata, Okayama University

Yuichi presented SEEdit, a tool to improve and simplify the configuration of Security Enhanced Linux (SELinux), which provides least privilege via type enforcement and mandatory access control. However, while useful, SELinux has a bad reputation; in fact, many recommend disabling it if one has problems. The reason for this is that security policy configurations are difficult.

Refpolicy is the current way to configure systems. It is developed by the community, policies for many applications are included within it, and it works well when the system is used as expected. However, it fails when used in other ways. Furthermore, because it's such a big policy, it's very difficult to understand how to change it to fit one's needs when they differ from the expected scenarios.

SEEdit tries to fix this problem by letting one write SELinux policies in SPDL, which is a higher-level language that hides many of the complexities of SELinux's policy language. After writing the policy in SPDL, it's translated into SELinux's own policy language. Instead of having to create type labels manually, SEEdit automatically generates types for permission rules listed in the SPDL language.

Yuichi demonstrated that SEEdit is able to create complete configurations using many fewer lines than are required by Refpolicy to describe the same security policy, making it much easier for a user to read and grasp, as well as verify. Furthermore, smaller policies enable SELinux to work in embedded environments where space can be at a premium. For instance, Refpolicy-based security policies can take a few megabytes of space, while a SEEdit-created policy only took up 71k of space.

However, the problem with SEEdit is that its current approach integrates multiple SELinux permissions into one

higher level which are merged, reducing granularity. For instance, reading a regular file and symbolic links are a single permission in SEEdit. They would have to expand SEEdit to understand more permissions.

Could SEEdit be used to manage the Refpolicy itself once its few issues are worked out? It would be possible to use SEEdit to manage the policy. A question was asked about the difficulty of debugging misconfigurations if one's configuration doesn't work as expected. The bug could be a result of the conversion into SELinux's language, so one might not know which SPDL rule created the SELinux policy rule that caused the problem. Yuichi agreed that this is important and needs to be worked on.

- ***An SSH-based Toolkit for User-based Network Services***
  Joyita Sikder, University of Illinois at Chicago; Manigandan Radhakrishnan, VMware; Jon A. Solworth, University of Illinois at Chicago

Jon Solworth spoke about securing user-based network services (UBNS) easily. UBNS involves authenticating users, encrypting communications, and authorizing and customizing the services based on the user authenticated. Different users have different access permissions.

In general, password authentication is used to authenticate to these services, but it isn't that secure. For instance, Dovecot mail service has a good reputation for security. It was built using four different process types for privilege separation, and about 37% of the source code is just implementing authentication and encryption, ignoring external libraries such as OpenSSL. This portion was implemented by a security expert and is not so easy to implement for regular programmers.

They've developed the SSH-based UBNS toolkit, which makes it much easier to provide all the requirements of UBNS with minimal changes to existing service applications. It doesn't require any knowledge of cryptographic libraries. It isolates the UBNS functionality into address spaces separate from the service functionality to have the OS enforce the isolation. By building on top of SSH, it doesn't require a global namespace, but instead allows each user to create their own public/private key pair.

The implementation is a modification to SSH. In a traditional SSH tunnel to a running service, the running service has no direct knowledge of the user who set up the tunnel, but with UBNS this information is available to the service. On the server side they provide an inetd type super-server, unetd, to manage connection to their managed UBNS services and require a simple modification to applications in the accept() function to make it UBNS aware.

On the client side, no modifications have to be made; instead the client connects to a local port on the client, which initiates an SSH connection to the unetd super-server to instantiate the requested services as the correct user and sets up the tunnel

PLENARY SESSION

- ***Towards Zero-Emission Datacenters Through Direct Reuse of Waste Heat***
  Bruno Michel, IBM Zurich Research Laboratory

  Summarized by Rik Farrow

Bruno Michel began by telling us that the energy consumption in datacenters has doubled in the past two years. Using a chart from the International Technology Roadmap for Semiconductors, Michel pointed out that energy leakage, manifest as energy wasted as heat, will only get worse as chip features continue to shrink. He also mentioned that he is a mountaineer and has personally seen the shrinkage of glaciers in the Alps, an obvious effect of global warming.

There have been improvements in energy usage, spurred on in part by the awareness of the impending crises. The Green 500 ranks supercomputers by the amount of useful work produced per watt. The number one supercomputer in 2002 produced 3.4 Mflops/watt, while the current fastest one produces 445 Mflops/watt, while being ranked number four in the Green 500 list.

The thrust of his presentation had to do with using water for cooling. The circulation of blood is efficient in transferring both nutrients and heat. Water itself has tremendous thermal capacity, much higher than refrigerants. IBM began using water to cool chips with its 3070 mainframe back in the '80s. Each processor chip (including those that controlled data transfer like today's northbridge) had a piston that rested on the chip, with an armature sitting above the set of pistons for circulating water.

Working from biology for modern design, Michel described fluid channels built right into chip carriers, with tiny channels nearest heat-producing parts flowing into larger channels above. Chip design needs to consider the position of cooling channels for areas that will be the hottest sections of chips. IBM has planned for stacked chips, with vertical interconnecting pins, interlaced with water cooling, providing shorter signal paths and much more efficient cooling.

Using this design allows water to reach 60°C (170°F), a point where it becomes feasible to resell the heat produced as a side-effect of computing. In parts of Europe, they can sell this "waste" heat for about half the cost of producing the heat via electricity or gas. Even where the climate is hot, like Dubai, waste heat can be used to preheat seawater for evaporative desalination. The goal is to reach a PUE/reuse ratio of less than one. Prototypes built using these designs are expected to cost 30% more, 10% when mass-manufactured, but these costs should come down to perhaps 3% over time.

Andrew Hume wondered if we will be needing plumber's putty when pulling boards? Michel said that admins will hardly notice the difference, as there will be connections for water and electricity. Hume asked if they had considered other fluids, because of bacteria? Michel said that corrosion

is a problem, but that they have used systems like this for over ten years.

Someone asked about radially oscillating flow cooling being used in cell phones, but Michel pointed out that the cost would be quite high. Hamil Howell asked what C4 technology meant. Michel said this is a technical term, controlled collapse chip connect, which uses solder balls that melt to make connections instead of wires. Doug Hughes of D.E. Shaw Research asked about using outside air: what's the overall efficiency? Michel said he was not an expert, but it is better to build a datacenter where you can disable the heat pump and use outside air. Doing so requires a large enough gradient for this to work, but you also have to filter out dust, which requires more power to pump the air. Steven Spackman said he grew up in Quebec where they use hydroelectric power. Michel pointed out that electrical energy is more valuable than heat energy, and that you need to understand the concept of exergy. DCs have 100% exergy, but if you can heat houses, you can get your exergy down to 10%.

ON THE FRINGE

Summarized by John Hewson (john.hewson@ed.ac.uk)

- ***Federated Access Control and Workflow Enforcement in Systems Configuration***
  Bart Vanbrabant, Thomas Delaet, and Wouter Joosen, K.U. Leuven, Belgium

  Awarded Best Student Paper!

Bart Vanbrabant discussed the current situation, in which most system configuration data is stored in source control repositories, with limited, directory-based access control being highlighted as a weak point in security or reliability. Examples were given, such as testing or development repositories, and shared configuration across grid computing sites.

Vanbrabant introduced ACHEL, a tool to integrate fine-grained access control into existing configuration tools. Using this tool, changes that are checked in to a source code repository are checked before being uploaded to a server. ACHEL is mostly language-agnostic, only requiring administrators to create regular expressions in order to apply access control to existing code. An email workflow is provided so that repository admins can approve changes made requiring higher privileges. The implementation of ACHEL was presented, with language-agnostic components discussed, as well as the language-specific abstract syntax tree parser component. Vanbrabant presented a prototype implementation as a test case, using Mercurial source control, Bcfg2 deployment, and a simple custom configuration language. A sample junior and senior sysadmin workflow was created. Finally, a larger federated example from BEGrid was presented.

The question of whether access control could be integrated into existing configuration languages was raised. Vanbrabant said that there could be better language integration but that it was desirable for access rules to be separate from the specification itself.

- ***CIMDIFF: Advanced Difference Tracking Tool for CIM Compliant Devices***
  Ramani Routray, IBM Almaden Research Center; Shripad Nadgowda, IBM India Systems and Technology Lab

Ramani Routray gave an overview of the DMTF Common Information Model (CIM), a standard for vendor-neutral exchange of management information, including the underlying XML, and its use in Web-based management. The problem of identifying meaningful semantic differences between the XML documents containing the CIM data was discussed, and the goal of performing meaningful difference tracking was identified. Routray then presented CIMDIFF, a tool that identifies semantic differences between devices which implement CIM, allowing system administrators to discover differences between systems.

Routray then summarized details of the implementation of CIMDIFF with overviews of the hash maker, knowledge base, and difference tracker.

- ***Transparent Mobile Storage Protection in Trusted Virtual Domains***
  Luigi Catuogno and Hans Löhr, Ruhr-University Bochum, Germany; Mark Manulis, Technische Universität Darmstadt, Germany; Ahmad-Reza Sadeghi and Marcel Winandy, Ruhr-University Bochum, Germany

Luigi Catuogno gave an overview of the problems with existing data protection on mobile devices such as memory cards and USB sticks. He discussed the issues of untraceability, lack of effective security, and management overhead on users. The authors offer Trusted Virtual Domains (TVDs) as a way to attain free and transparent deployment of mobile storage within an organizational network.

Catuogno introduced TVDs as a coalition of virtual machines with mutual trust, an enforced security policy, and the ability to span physical infrastructure. Catuogno introduced an extension of TVD which covers mobile storage devices, adding device identification and dynamic device management, as well as transparent mandatory encryption of sensitive data stored on mobile devices. Encryption keys are stored in a centralized key management database, and access control is integrated into the TDV policy. Offline access to mobile storage is provided by delayed re-encryption and delayed revocation of encryption keys. As a further reference, Catuogno mentions a prototype implementation based on the Turaya security kernel.

INVITED TALK

- ***Visualizing DTrace: Sun Storage 7000 Analytics***
  Bryan Cantrill, Sun Microsystems

  Summarized by Rik Farrow

I had heard that Cantrill was an interesting speaker, and that turned out to be an understatement. Cantrill both entertained and enlightened us with his funny, fast-paced talk on DTrace.

Although ostensibly a talk about DTrace in the 7000, Cantrill started with the story behind the creation of the tool. In 1997 he was part of a team working to tune a Sun Enterprise 10000 (E10K), a million-dollar server with up to 64 UltraSPARC CPUs, to run a TPC benchmark. The E10K worked well for a while, then performance "went sucky" for about four minutes, before resuming at a benchmark record-breaking level.

Cantrill wrote kernel modules to debug the problem, not something Sun's ordinary customers would even dream of doing, and eventually discovered that the mainframe-like machine had been misconfigured to act as a backup router, and would do so when the real router would crash. This killed performance until after the router rebooted.

Cantrill went on to write DTrace with Mike Shapiro and Adam Leventhal (see their 2004 Annual Tech paper at http://www.usenix.org/event/usenix04/tech/general/full_papers/cantrill/cantrill_html/), a project that earned the STUG award for them in 2006—not that Cantrill mentioned this, as he was much more interested in demonstrating DTrace on the Mac he was using for his presentation.

Cantrill started with a simple DTrace command, dtrace -n syscall:::entry'{trace(execname)}', which lists the names of programs executed. But this gave him a way of pointing out that DTrace has an awk-like syntax, with a probe pattern, followed by an optional predicate, and an action, surrounded by curly braces, that executes only when the probe triggers. Cantrill next demonstrated aggregation and histograms using slightly more complex examples. While typing, he quipped, "Using DTrace is a way of seeking food, not mates. It is not an aphrodisiac. Just ask your wife while reading the manual out loud."

Learning that the action expressions used a C-like language, called D, that borrows features from awk actually helped me to understand the previously impenetrable DTrace examples.

Cantrill continued his presentation by firing up the demo VM of the 7000 storage appliance so he could show how DTrace helps debug storage system issues. He told the story of the famous YouTube video that shows a Sun engineer shouting at a disk array (Just a Bunch of Disks, JBOD), resulting in longer latency. They had observed some unusual latency and tracked that back using DTrace tools within the 7000 to a single drive in the JBOD which turned out to have three of the four mounting screws missing. Replacing the screws fixed the latency issue. But to reproduce it they loosened all the drive screws, tried various ways to vibrate the array using synthesized sound, then finally discovered that simply screaming at the array produced increased latencies that clearly show up in the 7000 GUI. Cantrill pointed out that the camera is clearly shaking during the video because he is still laughing.

Cantrill concluded that looking at latency issues really helps uncover problems and that every aspect used in the GUI is there because they needed to understand performance.

Someone asked if the Sun 4500, an obsolete member of the Enterprise line, could be used as a NAS appliance? Cantrill answered that this is not supportable as a NAS appliance, but with their new masters they were going to look at how they actually make money. Was the 7000 was going to support FC (Fibre Channel)? Cantrill said he was working on an FC target that should be ready by Christmas. Mark Staveley asked about FCoE (FC over Ethernet), and Cantrill went off about how this was possible but something he considered ridiculous, as it meant replacing your FC infrastructure investment, so why not just go to 10 Gigabit Ethernet?

INVITED TALK

- ***Above the Clouds: A Berkeley View of Cloud Computing***
  Armando Fox, Reliable Adaptive Distributed Systems Lab, University of California, Berkeley

  Summarized by John Hewson (john.hewson@ed.ac.uk)

Armando Fox introduced the idea that the datacenter is the new server, albeit one requiring a large investment in infrastructure. He outlined the RAD Lab's five-year mission to enable a single individual to develop next-generation Internet apps. He introduced cloud computing as a new concept, separate from SaaS, which he describes as predating Multics. Pay-as-you-go utility computing, with the illusion of on-demand infinite resources, was identified as being the novel discriminator between SaaS and cloud computing. Fox described the advantages of provisioning in the cloud—better matching between capacity and demand makes a cloud datacenter considerably more efficient than a traditional datacenter, with perhaps 20% average resource utilization, where remaining resources are reserved for peaks.

Fox also presented some unique advantages of clouds. Cloud computing transfers risk for large capital expenditures where resource demand is unknown; cost becomes associative, as the cost of 1000 servers for an hour is equal to the cost of one server for 1000 hours, allowing academics to perform experiments on larger numbers of servers than was previously possible. He used the example of Animoto, which scaled from 50 to 3500 servers in three days and then scaled back down, using Amazon Web Services. Notably, the economies of scale from existing large infrastructure such as Amazon and Google, and their operational

expertise, are cited as the prime reason for the current trend toward cloud computing.

Fox next mentioned challenges and opportunities, with the primary challenge being the incompatibility of different types of cloud: low-level instruction-set VMs such as Amazon EC2 at one end, and Google's framework-based AppEngine at the other. There is scope for vendor lock-in, but also opportunities for the development of open APIs and free and open source software. Other uniquely cloud-based issues were presented: the costs of moving data, including physically shipping it, and the proliferation of non-relational scalable storage such as Cassandra, Hypertable, and Amazon SimpleDB.

Fox discussed deciding whether or not to "cloudify" an application. Authentication and data privacy are paramount when data is placed in a public cloud, and it is unlikely that the weakest link will be technical. For those considering building private clouds, he notes that other than efficient utilization, they are unlikely to be as cost-effective as public clouds. Overheads, such as a billing system, and incentives to release idle resources also need to be addressed.

The role of academics in cloud computing is seen as promising. Fox mentioned UC Berkeley's own 40-node Eucalyptus cluster, with a workload that can overflow onto Amazon EC2, on which they routinely perform experiments on hundreds of servers. He commented on the difficulty of incorporating cloud computing into the funding/grants culture and providing accounting models for cloud usage. Statistical machine learning is an area for future research, as the difficulty of resource optimization increases.

Mark Burgess asked whether weak coupling between cloud components was desirable and whether a change in software writing strategy is needed. Fox agreed that scale independence in software was both necessary and desirable. Alva Couch asked whether low cost would override privacy concerns, with Fox replying that legislation is always behind technology; however, for certain applications a new high-trust cloud might be needed, as the public cloud is out of bounds.

PLENARY SESSION

- ***Frank Lloyd Wright Was Right***
  Daniel V. Klein, Consultant

  Summarized by Mark Burgess (Mark.Burgess@iu.hio.no)

Closing sessions at LISA have been varied since the demise of the LISA Game Show, but Dan Klein is known for his snappy and erudite lectures that dance between playful analogy and serious commentary. With an impressively polished script and dazzlingly crisp diction, he didn't disappoint this time in his flawless execution of the closing session.

The talk was subtle but firm in bemoaning a lack of leadership we often exercise in design or repair of the infrastructure around us. "We," he suggested, "are collectively responsible for the messes we make, and by golly we are responsible for some corkers."

Dan likened our efforts to implement computer security, in particular, to the Byzantine ruminations of civil engineering in Pittsburgh and to US tax legislation, as well as to the bizarre Heath-Robinson-like contraptions we build from Web services using today's so-called state-of-the-art technologies. He showed a humorous battery of examples and illustrations of things gone awry.

The talk was "cantilevered" with examples and quotations from visionary architect, philanderer, and borderline fraudster Frank Lloyd Wright, who suggested, like Ayn Rand's fictional Howard Roark, that rather than feign "hypocritical humility" in building the world, "honest arrogance" might be a cleaner approach to infrastructure building. Thus he proposed to "raze it and start over" in the face of mistaken design, rather than keep people in ungainful employment, perpetrating "feeping creaturism" that creates monsters from incessant patching.

Wright proposed to merge form and function seamlessly from the start, to keep design as simple as possible, but no simpler. He posited that "less is only more when more is no good." We build, Dan suggested, some comically inept systems that suffice often more by luck than judgment, and we could take a lesson from Wright and try to build systems that work *with* rather than *against* the environment they live in.

The talk was a fast and festive romp through cultural and technical folly. There were more than a few laughs from the appreciative audience. The undercurrent was essentially this: as engineers, we should exercise more leadership in mending broken designs rather than using plasters and chewing gum to keep them together. Raze it and start over is an underestimated strategy, Dan suggested. We needn't fear it if we phase in major change incrementally, because, as Wright exclaimed, "Belief in a thing makes it happen."

LISA '09 WORKSHOPS

- ***University Issues***
  Workshop Organizer: John "Rowan" Littell, California College of the Arts

  Summarized by Rowan Littell (rowan@hovenweep.org) and Josh Simon (jss@clock.org)

The fifth annual University Issues workshop had six participants (down from 15 last year), three from the US, two from Canada, and one from Norway, with institutional sizes from hundreds to tens of thousands of users. The format of the workshop was a semi-structured roundtable discussion.

In the morning session, after brief introductions, we started with a discussion of personnel management issues. Some of us have managers who don't understand technology or can't communicate or don't trust (either their own people or

other groups). We had a mix of centralized and decentralized institutions, which affects the management styles; one example was of a central IT group that takes over faculty- and departmental-run technologies.

This segued into a discussion on project management. Smaller groups tend to either not have any or to self-manage projects, and larger institutions tend to have a dedicated project management team. Project management tends to be expensive, since there is a big time and communications commitment in doing it well. There are also issues involved with funding: is project management funded centrally or is the funding grant- or project-based? In general, management, both personnel and project, needs to communicate with the employees or users in order to make the right decisions for the needs of those employees and users.

We next had a brief discussion on virtualization. Most people are using either VMware or Hyper-V. Several places are looking at offering virtualization as a service. This segued into uses for virtualization. Eeducational institutions have liability and security concerns (in the US, the Family Educational Rights and Privacy Act (FERPA) is the big one to worry about); the issues and risks of exposing confidential information are certainly concerns when virtualizing on a third-party vendor. There are also the usual reliability and security concerns. Outsourcing IT to virtualization services may be good for smaller environments.

After the break, we moved from soft topics to more technical ones. We continued discussing virtualization technologies, touching on guest OS and network interactions (particularly network storage) and patch and update management; it was noted that Solaris zones make patching the guest zones at different schedules difficult. The topic of virtualization as a service led to a discussion of how to provide administrator rights to such systems, which and how many operating systems to support, and how to integrate hosted virtual machines with the rest of the department's computing resources.

The discussion then moved into various security concerns. Phishing scams and password management were noted as prominent concerns. Some institutions have implemented outbound rate limiting on email and scanning messages for known passwords. Password policies, including change frequency and complexity, were also discussed. Some attendees reported having to deal with students or other parties who have captured and cached passwords, usually for non-malicious purposes such as creating services that let people log in to multiple Internet services at once.

The workshop ended by moving back to a discussion of money. American institutions have, of course, been struggling under the current economy, foreign ones less so (with the Norwegian representative noting that students do not pay tuition per se and funding comes from other state sources). Money is easier to access for hardware purchases than for personnel, raises, or training, although some have changed how they budget for hardware (e.g., leasing machines so that the cost is part of the operations budget rather than the capital budget).

Although the workshop closed at lunch with a half-day format, all attendees agreed that it had been beneficial. There was a short discussion about the small size of the workshop: people agreed that a larger attendance would have been better but that the small group allowed for good discussions.

- ***Government and Military System Administration Workshop***

  Workshop Organizer: Andrew Seely

  Summarized by Andrew Seely (seelya@saic.com)

The Government and Military System Administration Workshop was attended by representatives from the Department of Defense, Department of Energy, NASA, Food and Drug Administration, National Oceanic and Atmospheric Administration, Raytheon, and Science Applications International Corporation. This was the second year the Gov/Mil workshop has been held at LISA.

The workshop concept was to create a forum to discuss common challenges, problems, solutions, and information unique to the government sector, where participants would be able to gain and share insight into the broad range of government system administration requirements. LISA was an opportunity for diverse government and military organizations to come together in a unique forum; it's not common to have highly technical staff from DoD, DoE, FDA, NASA, NOAA, and industry at the same table to candidly discuss everything from large datasets to organizational complexity. All expected to find similarities and hoped to be exposed to new ideas, and no one went away disappointed. The day's specific agenda was developed in the weeks leading up to the workshop, with each attendee identifying specific topics for the workshop to address. The agenda was adjusted as the workshop progressed in order to capture emergent issues.

The day started with roundtable introductions and a reminder that the environment was not appropriate for classified or sensitive topics. For system administrators outside the government sector this could seem like an unusual caveat, but for people who work in classified environments it is always a good idea to state what the appropriate level of discussion is for any new situation, especially when the discussion is about government systems and capabilities. The group agreed that the day would be strictly UNCLASSIFIED and that no For Official Use Only or higher material would be discussed.

The day was loosely divided between technical and organizational topics. Technical topics discussed included PKI and identity management systems, integrating authorization and authentication systems, and managing large datasets. More detailed and wide-ranging this year were topics centering on policy, procedure, and organizational structure, with heavy focus on industrial security and government/military security policy implementation. Detailed discussions on

certification and accreditation highlighted the surprising differences between government agencies.

All attendees presented what types of personnel their respective sites or companies are seeking to hire. Over half had positions to fill, and almost all required security clearances. DoE and DoD were generally hiring, while FDA, NASA, and NOAA were not. Hiring information and career Web sites were shared.

The final topic of discussion was to determine if there would be sufficient interest in this workshop to repeat it at LISA '10. It was agreed that it was a valuable experience for all attendees and that all would support a follow-on workshop. A Gov/Mil wiki was established at http://gov-mil.sonador.com/ to provide a collaboration area to help develop this workshop into a viable community of interest.

- ***Advanced Topics***

  Workshop Organizer: Adam Moskowitz

  Summarized by Josh Simon (jss@clock.org)

Adam Moskowitz was the host, moderator, and referee of the Advanced Topics Workshop once again. We started with our usual administrative announcements and the overview of the moderation software for the three new attendees. Then, we went around the room and did introductions. In representation, businesses (including consultants) only outnumbered universities by about 2 to 1 (down from 4 to 1); over the course of the day, the room included six LISA program chairs (past, present, and future, up from five last year) and nine past or present members of the USENIX, SAGE, or LOPSA Boards (the same as last year).

Our first topic was cloud computing. We discussed the various takes on it, and one of the clearest issues is one of definition: Technologists and non-technical end-users have different definitions of what it means. Comparisons to grid computing were made; the consensus was that grid is for high performance computing (HPC), cloud computing isn't, and that grid and cloud are solving different problems. When discussing cloud computing you need to determine if your definition includes the server/OS (be it physical or virtual), the applications, or the data itself. Then there's the issue of the data: who owns it, maintains it, backs it up, is responsible for restores as needed, and deletes it when you're done with it. So far, in general, cloud computing is good in that you and your company can save money on hardware and possibly on support (licensing, maintenance, and staff) costs, but so far we've ignored the security aspect. Contracts are all well and good, but there are legal and regulatory and security issues regarding certain types of data (student records, health records, personally identifying information, access control for research data, and so on) that make it a bad idea for some environments, industries (health and financial), and applications. How do you audit your cloud provider?

Next we did a lightning round of cool new-to-individuals tools or technologies. The most common response was a programming language (Erlang, Python, and Ruby); others were Bugzilla, iPhone, memcache, nfswatch, RRDtool, XMPP for system-to-system messaging, and ZFS. One person mentioned his new Viking 6-burner range.

After our morning break, we resumed with a discussion of file systems. Some are looking for alternatives to NFS that scale better; most seemed to like GPFS, and others mentioned OCFS2 and GFS2. In all cases, you need to look at your requirements to find the one that best suits your needs; for example, OCFS2 doesn't scale beyond 7 or 8 nodes in a cluster of virtual machines, but if you only have 3 or 4 it might be sufficient. This segued into a distribution discussion regarding what needs to be local and what can be remote, as well as what needs to be read-write (more expensive) versus read-only. From there we segued into charge-back. Can you charge back to other departments or users the cost of your file services (and, indeed, other services), and if so, how? Most people are looking at tiered models, such as "dumb SATA is free; if you want RAID or backups it costs more." The problem is that end-users can add cheap disk to their systems and not see the difference between disk (the physical device and its data) and storage (the infrastructure for availability, retention, and recovery). Some folks are charging back what they can even though it's not enough to cover the hardware costs, let alone the staff costs. It was stressed that you have to proportionally reflect your costs or the users will game the system, and you have to be careful not to oversubscribe.

Our next major discussion topic was career paths. Management is still the most common option career path for ever more senior people. In education, it's pretty much the only option, as you have to become a manager to grow into any CTO/CIO/Dean/Provost roles. In industry there's no well-defined career path; there's junior to intermediate to senior, but then it can tend toward either management or architecture/design. One possibility is "minister without portfolio," where you're known internally as a senior resource and various departments bring you the hard problems for advice if not outright solution, and otherwise you just do what needs doing. Some noted that manager-or-techie may be the wrong view. Leadership is the issue: does your organization provide a way to foster and grow leadership skills? It seems that "architect" is the "leader who isn't a manager" title. In addition to growth, the concept of job satisfaction came up. Some are satisfied more by title, some by compensation (salary or benefits), some by growth, and some by having interesting problems to solve. Where are you on that scale, and can your current organization satisfy you? If not, it may be time to find one that can.

After our lunch break, we had a discussion on automation. We talked about some of the differences between host and network based configuration tools, and how at the baseline you need to get a set of consistent configuration files for all the devices at a given point in time. The next problems are to get that configuration information to those devices, then

move from that set to another set from a different point in time. Do you keep the configuration data and metadata and state all in the same system or not? Are the tools topology-aware or not? We should move away from the procedural specification and more toward a declarative mode (e.g., "build a VPN between point A and point B"), letting the tools figure out the "right" way to do it for your environment. Abstracting up to a declarative level will be helpful in the long run but getting there is going to be challenging. The mental model for automation sits at the intersection of "how people think about their systems" and "what data the tool provides" or "what function the tool performs."

We next had a quick survey on the hot new technologies or big things to worry about in the coming year. Answers included automating failover and self-healing automation; changing the way people think; chargeback and resource allocation; cloud computing; finding a new job, getting out of the rut, having new challenges; getting useful metrics; outsourcing; politics at work; and rebuilding community and improving communications between IT and their users.

Our next topic was communications, both between technical and nontechnical (be they business or faculty as relevant) and between groups within an organization. Having an advocate for IT in the remote business group has been helpful for some people; holding tours of the datacenters for non-technical users has helped others. Empowering users to help themselves, such as with self-service Web sites or kiosks, helps as well. To get IT recognized as helping the business accomplish its goals and not as obstructions or obstacles, IT has to understand those business goals better. It's not that IT should say "No," but, rather, "Here's a better way to accomplish that" or "Here's what I need before I can say yes." Technical people need to remember that just because someone isn't technical doesn't mean they're stupid. One additional note is that, like nurses, we often see people on the worst day of their lives: something is broken, they have a deadline, and we have to fix what's wrong so they can get on with it.

After the afternoon break, we resumed with a discussion on mobility. Laptops and mobile phones are commodities now, so what policies do people have for managing them? There was a reminder that if the policy is too complicated, then it'll just be ignored or worked around. Most places have the management of laptops (both corporate and visitor) controlled by now, but handhelds are a newer problem. In general, the policy needs to scope what is and isn't allowed and to focus on what is and isn't within the control of the people enforcing it. This completely ignores the supportability aspects. Are VPNs the answer? DMZs for unauthenticated devices? As with everything else, "it depends."

The security issues involved in managing mobile devices segued into a discussion of identity management; it seems that many people are falling for phishing despite education, outreach, and announcements. Several have implemented email filters to look for personally identifying information in outbound email to try to prevent account compromises.

Security in general is about the same as a year ago (one person said, "It's better now"). It's still often an afterthought for infrastructure projects. We tried to brainstorm on how to get people to incorporate security. You need management buy-in and to change the culture, whether it's for regulatory reasons or not. It helps if there are policies to point to and guidelines to follow. Security is a method or a process, not a result. It does get better when more people understand *and* follow it.

Next we discussed our preferred scripting languages. Perl, Python, and Ruby continue to be the big winners, with shell scripting (and VBscript for those on Windows) trailing behind. Others include Erlang and Haskell.

We next discussed outsourcing. There's been a rise in many places of the percentage of finance-based managers who don't understand engineering or information technology. Outsourcing is only good in those cookbook situations where there's easily identified cause and effect, and specific tasks to accomplish. Companies don't think they're giving up control when they outsource. There are two different kinds of outsourcing: ongoing operations, where automating may be better (but since that's hard, it's next-better to implement it via API as request ticket to your outsourcing company), and project-based, where a particular project is given to the outsourcing company. However, it should be pointed out that in India, the average time-in-post is only three months, so a one-year project means you'll have four different project teams on average. That gives you lower quality, and going through a non-technical project manager gives you even less control over the implementation.

Companies are good at looking at the next quarter but not at long-term costs. One trick mentioned is to realize that the outsourcing company has limited bandwidth which you can fill up with less important projects, showing yourself to be willing and getting goodwill, even though you're keeping the more important projects in-house; and it's good to use some metrics to show how excellent you are at those in-house projects. Finally, we recommended that you keep your core business in-house; anything else is asking for trouble.

Our last discussion was a lightning round about the biggest technical or work-related surprises this past year. Some of the surprises included company closures despite profit and growth, company relocations, retiring executive management and changes in management or reporting structures, and responsibility changes.

Due to software issues, this year's Talkies Award could not be awarded. Last year's winner was D.J. Gregor, but he was not present this year.

Save the Date and Get Involved!

LISA'10 24th Large Installation System Administration Conference

www.usenix.org/lisa10 November 7–12, 2010, San Jose, CA

Important Dates

Extended abstracts, experience reports, and proposals for invited talks, workshops, and tutorials due: *May 17, 2010*
Notification to all submitters: *July 1, 2010*
Final papers and reports due: *August 24, 2010*
Poster proposals due: *September 17, 2010*

Program Chair

Rudi van Drunen, *Competa IT and Xlexit Technology, The Netherlands*

Overview

The annual LISA conference is the meeting place of choice for system and network administrators and engineers. The conference serves as a venue for a lively, diverse, and rich mix of technologists of all specialties and levels of expertise. LISA is the place to exchange ideas, sharpen old and new skills, learn new techniques, debate current and controversial issues, and meet industry gurus, colleagues, and friends.

Attendees have a wide range of administration specialties, including system, network, storage, cloud, and security administration, to name a few. They hail from computing environments of all sorts, including large corporations, small businesses, academic institutions, and government agencies.

We strongly encourage informal discussions among participants on both technical and nontechnical topics in the "hallway track." Experts in different fields are very approachable at LISA and are available to discuss your issues and questions. LISA is a place to learn and to have fun!

Get Involved!

The theme for LISA '10 is "Share your experiences, both real-world and in research."

Experts and old-timers don't have all the good ideas. We welcome participants who will share their experiences/lessons learned and provide concrete ideas to implement immediately, as well as those whose research will forge tomorrow's infrastructures. We are particularly keen to showcase novel solutions or new applications of mature technologies. This is your conference and we want you to participate.

Here are examples of ways to get involved—and/or propose a new idea to the program chair by sending email to lisa10ideas@usenix.org:

- NEW! Practice and Experience Reports: These will describe a substantial system administration project that has been completed and will consist of a 20-minute talk, with a 10-minute Q&A session immediately following. Talks must be highly practical, showing do's and don't's. Submissions will consist of written proposals no more than 4 pages long. Proposals will be refereed by the program committee.
- Refereed Papers: These are published papers, 8 to 18 pages long, describing work that advances the art or practice of system administration. Presentations are limited to 20 minutes.
- Invited Talks: Invited Talks are one-hour presentations on a single topic of interest to system administrators. Talks may be historical or focus on the latest hot technology, be serious or funny, cover a spectrum of related issues or dive deeply into one specific topic.
- The Guru Is In Sessions: For the Guru, these sessions are a chance to share your expertise with your fellow system administrators; for the audience, these are a chance to get your questions on a specific topic or technology answered by an acknowledged expert.
- Workshops: Workshops are half-day or full-day sessions for small groups (typically not more than 30 people) to share ideas and knowledge. Workshops are intended to be participatory, not instructional, and familiarity with the specific topic/area is expected of the attendees.
- Training Program: Tutorials are also half-day or full-day sessions but, unlike workshops, tutorials are intended for a single expert to share knowledge, not to be open discussions. Even if you're not an instructor, you can suggest classes.
- Work-in-Progress Reports (WiPs) and Posters: This is your chance to share an idea that could turn into something more formal at next year's conference. Both WiPs and posters are a good way to make your first presentation at LISA.
- Birds-of-a-Feather Sessions (BoFs): BoFs are informal gatherings held in the evenings. Topics range from use of a particular software package or product, through folks wanting to talk politics, to people interested in a particular aspect of computing.

USENIX Federated Conferences Week

June 21–25, 2010 • Boston, MA

http://www.usenix.org/events/#june09

USENIX Federated Conferences Week will feature:

- **2010 USENIX Annual Technical Conference (USENIX ATC '10)**
- **USENIX Conference on Web Application Development (WebApps '10)**
- **3rd Workshop on Online Social Networks (WOSN 2010)**
- **2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '10)**
  Paper submissions are due March 23, 2010; see http://www.usenix.org/hotcloud10/cfp.
- **2nd Workshop on Hot Topics in Storage and File Systems (HotStorage '10)**

USENIX is seeking suggestions and proposals for offerings (e.g., sessions, workshops, tutorials) for co-location. For more information, please contact the USENIX Executive Director, Ellie Young, at ellie@usenix.org.

Opportunities for Community-Building and Staying Informed

Stay Informed: Check out the USENIX Update Blog. You'll find the latest in conference announcements, submissions deadlines, available proceedings, new multimedia, and much more: http://blogs.usenix.org/

Subscribe to the RSS feed and stay current on all USENIX info: feed://blogs.usenix.org/feed/atom/

You can also follow us on Twitter: http://www.twitter.com/usenix

Stay Connected: USENIX has groups on LinkedIn and Facebook. Joining the groups will help you connect with other USENIX members.

- LinkedIn: http://www.usenix.org/linkedin
- Facebook: http://www.usenix.org/facebook

USENIX
The Advanced Computing Systems Association

Connect with other members and keep up to date on the latest USENIX news!

Expand Your Mind and Your Bottom Line

UNIFIED COMMUNICATIONS

VoIP

SECURITY

DATA CENTERS

@it 360°

Presenting

CIO Thought Leader Forum

Co-located with

CLOUD COMPUTING CONFERENCE

AN IT PARADIGM SHIFT

ASTERISK OPEN TELEPHONY CONFERENCE

IT360 SILVER SPONSOR: MEUK / RDBASE.NET

IN COOPERATION WITH: CIO Executive Council — Leaders Shaping the Future of Business

MEDIA PARTNER: itWorld Canada — IDG CANADA

ASSOCIATION SPONSOR: USENIX

PREMIER MEDIA SPONSORS: COMPUTERWORLD, NetworkWorld, THE GLOBE AND MAIL, CDN

PLATINUM ASSOCIATION SPONSOR: CATAlliance

CERTIFICATION SPONSOR: Procept ASSOCIATES LTD.

CONFERENCE SPONSORS: INFO-TECH research group, CIO CANADA, TORONTO Economic Development, Culture & Tourism

IT professionals and business executives maximize your success: **www.it360.ca**

METRO TORONTO CONVENTION CENTRE, CANADA

CONFERENCE & EXPO : 7 APRIL 2010

USENIX
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send Address Changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES