;login:
FEBRUARY 2014 VOL. 39, NO. 1
ELECTRONIC SUPPLEMENT

Conference Reports

Advanced Topics Workshop at LISA '13

Washington, D.C.
November 5, 2013
Summarized by Josh Simon

Tuesday's sessions began with the 19th annual Advanced Topics Workshop; once again, Adam Moskowitz was our host, moderator, and referee. We started with our usual administrative announcements and the overview of the moderation software for the one new and several long-absent participants. Then we went around the room and did introductions. In representation, businesses (including consultants) outnumbered universities by about 2 to 1 (down from 3 to 1 last year); over the course of the day, the room included five LISA program chairs (past, present, and announced future, down from 11 last year but much closer to our five- and ten-year rolling averages of 7 and 5.5, respectively).

Preventing Mistakes

For the first topic we talked about how to prevent the "I know what I'm doing" syndrome. One person had several outages over the past year caused by either development (code mistakes) or operations issues; examples in the industry include Knight losing $172M/sec (http://bit.ly/1a9Vapb) and Twilio's billing system sending out over-large bills (http://bit.ly/17egzzp). How do you make sure you (or your peers) don't cause unexpected outages?

This led to a lively discussion about processes and culture. One answer to the question was to have a backup so someone can go away (on vacation, a promotion, or departure) and the organization still has the skills. Another answer was to change the culture from "A checking up on B" to "A and B working together." If the culture is such that someone works with you to share the risk and responsibility, outages become less frequent. This requires putting in processes so nothing is ad hoc, such as "always do it in a test bed/sandbox first."

Document your changes (you do have a change management policy, right?). Include things like a change template: Do you need to update monitoring or other services? Is there a rollback plan? If the change is customer-visible, were they notified in advance? This leads to focusing on the customer experience; anything that's customer-visible should get greater scrutiny.

Also, document the policies and procedures. Consider having the policy include "Have a second set of eyes review it" before performing the task. Your processes also need to be written down and followed. Creating that documentation alone reduces the effort; even simple checklists can help prevent problems (at least as long as they're followed). This also acts as the bridge to automation; you should automate what you can.

Other comments included slowing down to do it right so it only needs to be done once, and then automate it; being willing to say No; reminding people that even simple changes can have catastrophic consequences; and remembering that some people learn from making mistakes.

Finally, if there is an outage, you need to have some form of after-the-fact discussion, but framing the questions there is key. "What did so-and-so do wrong" implies a blame culture, but "What did our system do wrong" implies a process culture. The latter is more likely to lead to positive results.

Favorite Tools

After that discussion wrapped up we took a poll: what's your favorite new-to-you tool over the past year? Answers included nine-foot tall cabinet racks, C++11, Colibri, CamScanner, dmarcian.com to analyze dmarc mail data, Docker, Emacs as a server on Mac OS, Evernote (premium subscription), Git, Graphite, a GPFS-native RAID appliance, having a good project manager, LaunchBar, logstash, a new office chair, online collaborative tools like Google Docs or Dropbox, packer, Quicksilver, S3 Glacier, and vagrant.

Fostering Communications

Just before the morning break, we discussed fostering communications and teamwork. One person had an environment where a group of developers created a product that used root for everything, including reformatting disks. He needs to cultivate a culture of looking for information instead of working in a bubble. The consensus around the room was that breaking down the silo walls between teams is important. Some suggested remediation included job shadowing; managing the developer/administrator role/skill gap; creating a moderating team with representatives from multiple groups, whose purpose is just to ask questions so that alternate perspectives are considered in the requirements phase instead of after implementation; "making rounds" like a doctor to see your customers and make sure their needs are being addressed; using other technologies such as a shared wiki and online chat; offsite social gatherings like lunch or bowling; and having the developers involve the sysadmins early in the process.

Software-Defined Networking (SDN)

After the morning break we talked about Software-Defined Networking (SDN). Someone introduced it to us at last year's workshop, and one of our cohort has thought about it for the last year and hoped vendors would come out with more hardware. One problem is that not everyone agrees about what SDN needs; to some it's running config management on the switches, to others it's running OpenFlow, and to still others it's having Perl or Python APIs to access the switch in real time. It seems to be gravitating towards defining it as the separation between the

data layer (protocols) and the control layer. The hardware is being consolidated into the ASIC manufacturers like Broadcom; the chips are integrated into the switches (either on-board or with add-on cards). OpenFlow (v1.3 or later) promises better control for sampling on the fly, such as with an intrusion detection system. Someone sees its major use cases as bandwidth management (between multiple datacenters with quality of service) and multi-vendor cooperative partnerships (such as managing hundreds of thousands of virtual machines on thousands of physical machines so they need VXLAN on top of the switches to manage things). However, another of our cohort talked to his networking team to get SDN in-house, and got a lot of push-back because "it's just using Python to control the switches." He had to educate them; they don't see the bus that's coming up on them.

Manufacturing IT

That segued into our next topic, Manufacturing IT. Ben Rockwood gave a talk about looking at IT from an Ops Management perspective a few years back, and applying manufacturing's building concepts to IT: moving systems to single-feature deployments, reducing cycle time, and so on. The question was whether there is any practical use for it yet. One person used the analogy of sysadmin-as-mechanic talking to the customers-as-drivers; it's not just about fixing the car, but about making it more efficient.

One person recently had a gratifying experience; they're migrating from an old job scheduler to a new one on their cluster. The most immediate and effective way to communicate the new capabilities and kindle the users' excitement was to sit down with them, watch what they're doing, and improve the user experience. Sysadmins need to remember that not everyone understands the system in the way that we do. Another noted that asking the users direct questions was essential. We need to make sure the problem we're trying to solve is actually worth solving now; you may find that other areas need the resources focused on them instead. It was also noted that there's both the design and operations levels, and how you work at each is different. At the operations level, once you know how things should be done you should automate it as much as possible to free yourself up to think about more at the design levels.

One person noted that we're focusing on the one aspect of the DevOps concepts here. A counter-example is that we have a lot of operations automation, so what's the value-add of change control? What's the model for where the efficiencies should be? He's disappointed that we're not going beyond what we're looking at and looking at other industries. It was pointed out that Alva Couch can't get system administrators and operations researchers involved in each others' areas.

Personnel Issues

Our next topic was about personnel issues. One person, now effectively a CIO of his company, had a person whom he called "the single worst sysadmin manager ever." This person trained his staff to believe "the fastest way to get fired is to get noticed," would do the work and not explain why, and would take tasks away from the technical people. This CIO inherited this team, fired the manager in question, and tried to retrain the SAs, but unfortunately wasn't that successful. Despite providing opportunities to learn and grow, only two people of a nine-person team looked into it (and got spot-bonuses and promotions, both publicly announced, and commensurate salary increases). What could he have done better?

One person said it comes down to autonomy, mastery, and purpose, and with the previous manager all three were broken. Reinstilling one is possible, but reinstilling all three may be impossible. Starting with purpose is best; once you have purpose you can build mastery. Another noted that the sysadmins themselves need to feel responsible for something to be engaged in or with it. There's certainly a fear of blame (if something goes wrong), and a lack of responsibility leads to the "who cares?" thought. As someone else noted, one can teach a toddler to respect edges by letting them fall off a couch but not out the second story window.

Someone suggested developing a post-mortem culture. Any and every time something goes wrong, a post-mortem is required. Another suggested that for major projects they should hold a post-mortem even or especially if everything went well. Make it clear that it's not an issue of blame but of learning. Also, having a Wheel of Disaster meeting where the team is given a list of scenarios and then choose one, with someone running it (as the gamemaster) and someone else as the on-call person, and do a dry-run of the troubleshooting "on paper" in the meeting.

Another asked if the sysadmins this manager hired were the right ones to begin with, or if they hired great people and broke their spirit? The response was that it was about a 50/50 split. Money and title is not always enough motivation and reward, and for some, money can even be a demotivator. Be careful to be supportive and not to blame someone if something goes wrong. As an example, were the sysadmins given incorrect or incomplete instructions?

Someone asked if the CIO talked to the individual sysadmins one-on-one as to why they didn't step up. Sometimes people don't want to say anything publicly. Their manager did and got "I had to do this other thing instead" as the response. Another added that trying team consensus brainstorming on what the team's top priorities should be to allow them input to the decision may make them more likely to buy into it. People need to take pride in their work, not necessarily take responsibility for it (which can be an onus). For recognition and awards, one environment gives out peer-nominated awards quarterly across IT, as voted on by the managers. They don't have a post-mortem culture but a preventative action culture so you can make things better before the disaster.

Finally, classes that teach what the knobs and widgets do but not why you might set them one way versus another aren't that helpful, so several write their own Best Practices documents that start with the why before going into the how. That way, when you check to see if something is still working, you can see whether the why changed.

Career Management

Next we discussed topics under the umbrella of "career-type stuff." One person is nearing retirement age. Another works for a small company acquired by a bigger one and isn't sure what's happening with his role. Yet another is being pushed into more of a mentor role than a hands-on technical role. The general question amounts to, "What should I do?" To lend perspective we took some quick polls. Of those in the room, half were born between 1950 and 1969 and half between 1970 and 1989; two people have over 30 years of experience, nine had 21–30 years, and seven had 11–20 years of experience.

The consensus was that you should enjoy what you do; focus on the opportunities provided by change, instead of on any fear; let go of control, which can be hard; look at the big picture; make sure your boss both knows and approves of what you're doing, be it hands-on technical or mentoring or direction; and work with your manager to explore areas of personal interest that are relevant to the organization's goals.

One person has the title of Director and actually gives direction, acting as the wise elder (but not the wise guy), setting direction but not dictating the details of how to get there. Write up your own job description and poke holes in it to see what's missing. For those in acquisitions consider the opportunity as a new job. Another notes that you can't grow within your comfort zone; growth requires moving outside that zone (and eventually expanding it).

On the subject of handing off tasks (as one approaches retirement, or is promoted, or otherwise leaves one's existing role), several worried about what will get left behind and possibly dropped, because in the specific case their coworkers are just putting in their time and won't do anything extra. Getting people out of this mind-set is hard. The consensus here was that while "document it all before you go" isn't enough, you can still set up your former area to succeed, but once you're gone they can choose otherwise and it's not your fault.

Another question was whether anyone else found that they want to do what they like on their own time and just have a job to pay for it? The short answer was Yes: one person is around a lot of actors who would love to do acting full time but can't make a living at it. Another has a job mostly writing code, and meetings are there for getting the requirements to write the code. And yet another moved to a job that's more in line with his interests and does the things he hated at his old job for fun and with less politics on the side.

Whither LISA?

After the lunch break, a USENIX Board member asked us to discuss the LISA conference itself: is it where it should be, or should things change, and if so how? One person's sense is that LISA both needs to change and is changing; they wondered how we think it should change, and what purpose it serves.

From a marketing standpoint, many agree that the conference has needed better marketing for a long time. One person (involved since 1997) found out about the conference on their own, and asked how others found out about it. Answers in general were from a boss, coworker, or friend (about half the room); exposure to other USENIX conferences (four people); higher education, such as a computer science department's posting; unrelated events local to a past conference (two people) and several of us simply couldn't remember. Only one person first heard about the conference from USENIX. There are also more events these days, both regional (Cascadia, LOPSA-East) and specialty (PyConf). Should we better integrate the joint conferences (for example, a Puppet track instead of Puppet on the side)? More marketing needs to happen through more venues; word of mouth just isn't enough.

As for the content, the consensus was that it's fine, though there's always room for improvement. One person notes that many years ago, many of the things he found out about at LISA were things he could introduce to his workplace right away, and he's seeing much less of that. The refereed papers seem much more niche-specific. It's not clear whether this is because the papers and talks have changed, or if it's because we've gotten older. As for what USENIX can do to make it more relevant to those of us in the room: over the years, people who come to the workshop are here to give back to other people and they get workshops out of the conference. Someone noted that (by design) this workshop is for the more senior people, so between the hallway track, the ATW, and giving back, we're probably not the audience to answer the question posed.

On the subject of content, one person asked if the refereed paper model is obsolete, as that may be part of the problem. USENIX tends to come from the academic side of things where papers are all, but LISA tends to come from the practical side of things, and these concepts are at best orthogonal. Several of the workshop attendees have never written a paper, with the "Nothing I do seems to be of sufficient interest to other people" rationale.

As for the tutorials, there seems to be an impression that seniors see them and think they shouldn't need to spend a whole (or half) day on that subject just to get to the one chunk they want late in the tutorial session. There is a perception, at least among those in the room, that tutorials tend to be more for the junior to intermediate administrators (more as topic introductions); some should be aimed at seniors, for deep dives into a more narrow subject. LISA seems to be marketed more to the beginning through intermediate administrators, but the senior people

definitely add value. One suggestion was to partner with or at least visit and snoop at other conferences like Velocity or SCALE to see what they're doing that we're not. We took a quick poll of the PC members in the room to see how many went to a competing conference this year and more than usual had.

Another person noted that students are missing. There used to be a significant visible student presence and that's no longer true. The role LISA has played, in addition to where to learn about the new things, was to provide a common language and a framework for discussion. There are more specialists now (DevOps, only-servers, only-storage, and so on), and LISA is a place for generalists and for bridging the gaps between the specialties. Someone else concurred; one of our competition conference's attendees skew much younger than LISA. How do we bring in more (and younger) people? Are we aiming for more juniors, or just more people? The conference has multiple tracks, but it's not always clear who the tutorials are for (junior to intermediate), and who the workshops and tech sessions are for, and where the senior folks fit in. Getting coherent content can be hard.

That led to a brief digression about giving back. Someone believes that many of us are still attending LISA after many years to give back to the community. We took a straw poll and only eight of 20 in the room at the time actually think they're giving back (including the five past LISA chairs). Some of that's from being in the Hallway Track and being available for others to ask questions, and others are teaching. One notes that especially for people who haven't been attending for long, the Hallway Track can be very intimidating. We need more informal ways of meeting people other than "let 'em loose in the same room."

Other Jobs

Our next topic was what we did outside of our day job as our "other job." Answers included being in a leadership role at church, bicycling, building (doing general contractor work), contributing to public software development projects, cooking, making art, officiating hockey games, parenting and grandparenting, photographing, teaching Highland wrestling, volunteering for both technical and nontechnical organizations. That's interesting but what's the cross-over? What do we learn in the one that helps out in the other?

One of the cooks notes that from cooking he got mise en place, staging everything before the stove goes on, and he applies that to datacenter moves. The wrestler learned humility. He used to wrestle and had fun, and he did well . . . until he was matched up against someone nearly half again his size in the last heat. He translates this to the technical arena by looking at how the younger people were coming along and he can't keep up—and the humility in not keeping up was almost an epiphany. The photographer now understands how to prevent people from abusing his time. People are hitting him up for free photo shoots, not realizing what time is involved in doing stuff. That translates well to IT work: time is valuable, and "just this quick thing" is often neither quick nor appropriate for the existing project. Several think books could be written about how parenting translates into dealing with people. One now appreciates the years of couples counseling and the relationship with their spouse, especially in contentious situations. Another notes that in acting, especially in comedy, you have to give the audience what they want. Timing, listening, and understanding what they want is critical both there and in technology.

What to Follow Next Year

Next we had one of our traditional polls: "What's the thing you'll be following in the next year?" Answers included bring your own device (BYOD), OpenStack, Software-Defined Networks (SDN), changing federal appropriations for IT in the wake of the healthcare.gov launch, dealing with internal politics, developing a process to handle end-of-life (such as Windows XP), making things enterprise-worthy in their kludged ecosystem, managing people's expectation of "everything works and never crashes," the density of flash memory, the role of system administrators in national security, and the state of encryption. Several also talked about cloud technologies, from deploying smaller more-local cloud technologies, to the security (or lack thereof) in the cloud in the wake of the Snowden revelations, to the debate on using the public cloud versus a mixed (public/private) cloud.

Cloud

Since we mentioned the cloud, that was our next topic of discussion. For some people their professional world is all in the cloud, others are still all local and think "cloud'll blow over," and then there are people whose users are dealing with both. Technical integration and expectation management are questions. For example, one institution is moving mail to the public cloud, thinking "It'll all be the same, just on their servers not our servers," but really it's "Our 100,000 users are distributed to some vendor's cloud." How does the help desk handle that? This clobbers the customer expectations.

While outsourcing infrastructure to the cloud isn't different from outsourcing anything else to the cloud, in that you always have to take the service-level hit, and the tradeoff needs to be calculated in the initial deal, the decision-makers often just hear, "Oh, it'll save money" and didn't do any other research into the technology or its security. Marketing needs to happen: "Doing this to lower our costs" is the common refrain, and sometimes lower cost will have side effects like a lower level of service (to keep tuition down, to give raises, and so on). The help desk staff will receive service calls for things they can't control or affect, and have to handle the users. There may often be no recourse beyond "call the vendor and hope."

Someone else comes at this from a different perspective. A software company providing what became Software as a Service on premises, then became a cloud provider, and one of the biggest things they learned is that customer support is critical; custom-

ers are much less invested in your product than formerly, since switching cloud vendors is possible (and likely, if you screw up your service).

Enterprise Monitoring

One person had over a hundred alerts from his monitoring system today and is seeing over 700 per week on average for the past month. They're obviously getting ignored by the entire team. If you're getting alerts that clear themselves and users don't notice, should that monitor be disabled? What are some best practices in monitoring?

Answers included alerting on "the right stuff"; being selective about how you monitor, as "95% full" on a 1 PB disk array might not be relevant, and perhaps the rate of change is more important; only monitoring actionable items; monitoring services, not just components, such as caring about the end-to-end user experience and not that a particular process or server is down; only emailing for important issues and using dashboards for less-urgent ones; opening tickets to yourself when you see gaps in what should be monitored but isn't; paging the developers for their service alerts to incentivize them to fix their own problems (false positives); providing context, such as letting the users of a service receive alerts and telling the sysadmins if action needs to be taken; putting the sysadmins on change management emails, so they can tweak the monitors where appropriate; turning off monitors that are unused or ignored; and using easily filterable subject lines in the alert emails.

It was pointed out that in Nagios the retry-check-interval and number-retries can be changed ("It's not a real problem until it's been 30 minutes"). It was also mentioned that you should be monitoring services, what's about to break or whether the service is down, not so much the server or component. In Nagios 4, you can attach a "wait" to an individual check: "Don't tell me something's wrong until X is also wrong."

One person wants three categories in a monitoring system: thresholds (number of occurrences or time length of the event), priorities (e.g., critical, major, minor, information, warning, and wtf), and escalation policies. Some alerts are just for reporting and don't send email; "1000 memory errors" isn't a problem over a year but is over 10 minutes. Dependencies are also important; if the switch goes down, you don't care about alerts for the 20 servers behind it.

The original questioner was happy he's not the only shop seeing these sorts of problems.

To-Do List Tools

After the afternoon break, we held another poll about what tools we use to track our to-do lists and whether that tool actually works. Answers to the former included Cozi, email, Evernote, GitHub, GTD, GoTasks as hooked to Google Calendar, JIRA, Lotus Notes, OmniFocus, paper notepad, spreadsheet for tracking the team's tasks, text editor of choice, and trouble ticketing system such as RT. Answers to the latter were either "yes" or "mostly," with a couple of "meh"s thrown in.

Asperger's Syndrome

Our next topic was Asperger's Syndrome, which the DSM-V has collapsed with autism and other concerns into the "autism spectrum disorder" catch-all. The question was asked whether we have any coworkers that have (or that we suspect have) Asperger's Syndrome, what challenges we have in dealing with them, and what would most help us address those challenges. The goal is that if you understand how someone's mind works you might be able to work with them more efficiently.

Asperger's is an autism-spectrum condition characterized by difficulties in social interaction and non-verbal communications, and restricted behaviors and areas of interest (very depth-first). Also, there's atypical use of language; one individual (recently diagnosed with it) tends towards the pedantic and literal, and his sarcasm detector runs about 10 seconds slow. He self-describes as an idiot savant: there are some things he does really well yet some basic things he does very poorly. It has to do with how his brain handles associations. If someone says, "Something green on the ground," most default to "grass"; his brain goes breadth-first (green CD case, spray paint, etc.), so jumping to the obvious doesn't happen—which can be an advantage for evaluating possible options that the neurotypical brain might discard. Most people's brains track social cues and proxemics and kinesics; Aspie brains tend not to. The upshot is that he wants to improve awareness of this sort of thing, so he's wondering if we have experience with coworkers who may have it and would resources (such as a BOF) be helpful?

Several others identified that they or their families have some form of autism spectrum disorder or ADHD (five in the room), and others (nine attendees) know or suspect it in their workplaces. In workplaces in general, for this and other hidden disabilities, people won't feel bad if you point out or help with issues they know about. For example, reminding an ADHD person that something's time-sensitive helps them break out of a loop. The speaker's Aspie friends have said the same thing about social cues, and being reminded would be helpful. That doesn't relieve people of the social requirement to do that reminding in a socially acceptable way. Be sensitive providing that feedback.

In an engineering-specific work environment, one of our cohort is sure there's tons of it (e.g., "wear the same shirt for a week, but write bulletproof code"). Now as a parent he's more tuned into the possibilities and can nudge them along.

Programming Beyond Scripting

Our next discussion was on programming beyond scripting. Some believe that once a site reaches a certain size you have to use custom software to manage it, beyond just off-the-shelf software and a configuration management system. Given some uncited research studies and anecdotal evidence that implies some people simply don't and can't be taught to think algebra-

ically (for example, "x=2, y=3, set y=x, now what's x?"), the question posed was if most sites have system administrators who code in a programming (not scripting) language, is there agreement that some people just can't make the shift from scripting to programming languages?

One expressed dissatisfaction with someone with a sysadmin degree from a particular institution who can't code, but another noted it might be luck of the draw as they have four people from that institution who can and do.

One noted that coding ability is a continuum, from none through some to all. One speaker thinks he's a pretty good programmer but doesn't have to code often (about 10%); he's happy with his physical, infrastructure, and monitoring gig. There's room for non-coders in the profession. Several people agree that they don't code often enough to feel comfortable with it.

The question can be applied more broadly to all sorts of skills: does a system administrator need a Computer Science degree? Within a group you need a little bit of everything; scripting is programming, just in a different language. A group should have experience and exposure to both sorts of languages. But when it comes to code, it's not that unusual for generalists to look at someone's code in a language you don't know, so exposure to the concepts is necessary. You have to be able to think logically and analyze while troubleshooting or looking for a bug; it's all problem-solving. There is, however, a difference between troubleshooting and debugging someone else's code and creating the code. While there are differences between debugging and creating, you still need to understand algorithms.

Given the advanced automation tools (like Hadoop, Nagios, CFEngine, and so on) that require some level of programming, is there something today that doesn't require programming? One environment has a requirement that all code pass code reviews and readability testing, and they have the same (strict) testing coverage for operations code as for the business-specific code. Another was trying a decade ago to remotely manage machines beyond turn on/off, and now that there are APIs that the hardware vendors support, he can. And now that those integration functions have software packages, it may not be necessary to write this stuff any more. As more and more automation tools have higher-level languages, there's less need for scripting tasks. In several of them you can say what to do, not how to do it. There's still a need for programming but perhaps not a strict requirement.

Workshop

The next subject was the workshop itself. We traditionally have some trouble hearing each other. One of the other workshops uses wireless microphones and speakers. Only about ten of the attendees favored using wireless microphones. One person was worried about germ transmission. Another suggested standing up to speak to project better.

Consensus was that there should be another ATW next year. One attendee wanted both fresh blood, people who haven't been to the workshop before, and fewer graybeards. Our recent new members have mostly been referred by existing participants; we as attendees need to get new people in by talking to them and Adam. Part of John Schimmel's (founder of the ATW) original purpose was for senior people to speak to each other without juniors interrupting. Most of our cohort, however, were against having a listening-only role for juniors. Someone noted that even senior people who see "position paper" in the sign-up are too intimidated to write one. Several thought that's still a low bar, as the CFP includes an example (which amounts to "write a proposed problem, not its solution"). Others thought that the position paper was a hurdle, that people need to think they'll get a benefit from the workshop before they do the work of writing the paper. Consensus was that the writeup in the CFP should be different, and Adam asked us to send him suggestions, some of which are below.

One member noted that different people can contribute differently. One possibility is that already-insiders can nominate and sponsor one person per year. Several agreed that would be a good idea; however, there is a 30 seat plus 2 (Adam-as-moderator and Josh-as-scribe) hard limit for the workshop.

Another noted that some may be more wary of the image of ATW than the reality, and wondered whether it's the old boys' club. Several thought that there's a perception problem outside this room. The response was that the writeup is in ;login: every year and that there's more than just the CFP writeup to help someone determine whether to apply. One suggestion was to link to the previous year's writeup from the CFP description. Someone noted that this writeup is appreciated, but we often don't continue the discussion (either at or after the conference) after the writeup gets published. Helping people understand what we discuss may have value. Many people aren't self-confident enough to stand up to "20+ years of experience."

One first-time attendee said he'd been to some other LISA workshops and thought ATW was by far the most functional workshop he's been in; many of us shared new-to-someone ideas and agreed that the ATW is both run well and well-organized.

Security

Security was up next. The question posed was "Is the enemy of my enemy really my friend?" One person's environment has an IT Security group that dictates policy but doesn't actually interact with or seek input from the system administrators, and frequently in companies you have Security making a recommendation to fix a problem but Sales saying they need it so you can't fix it. Do the sysadmins get involved in security audits? How well, or not, do your sysadmin and security organizations work together?

In one case, there was animosity between IT Security and Sysadmin. By reading the actual security regulations one Opera-

tions person was able to better understand Security's motivation and could therefore help build the bridges between the two groups. The two groups need to support each other, but we need to recognize the importance of both sides. In another case, Security would invite specific Operations staff to a multi-day training course and then keep those people as liaisons.

One environment would use the position of Security Manager as a patronage because doing real security well is hard but faking it takes no effort at all. The more security theater a company has, the less inclined the sysadmins are to be helpful. In another environment, Security and Operations are co-located, eventually reporting to a common manager, and the relationship is very collegial and not at all adversarial. Someone pointed out that an adversarial relationship can develop from how each group measures or defines its success: if Auditing is measured on the number of issues identified, for example, but Operations is not measured on the number of issues resolved.

Some environments are very checklist-driven, some are balkanized as opposed to centralized, levels of enforcement often vary (does Security have the authority to turn off a compromised Operations box?), and it's not always clear what kinds of exceptions are allowed (e.g., "patch every 30 days" is at odds with "we have a 2-month release cycle").

What's Coming Up?

Finally, we had our last lightning round: what's the biggest or most important thing coming in the next year? Answers included being in a team doing automation and writing code, building a Web cluster, building out their cloud, changing the billing model, continuing to get better with politics, driving the IT manufacturing concept, evangelizing for and implementing configuration management for both infrastructure and application spaces, extending dependency resolution into spinning up machines, finding a new job, finishing the six-month Hadoop rebuild, getting a monitoring solution he doesn't hate, improving security in general, increasing staffing, revamping and modularizing his 20-year-old dot files, ripping out and replacing their build/automation/CM system and putting in something quality, solidifying DNSSEC, surviving a reorg at work, taking all the different opinions about CFEngine into a coherent plan, deciding whether to retire in March or June, working on the next generation of supercomputers, and writing a refereed paper for LISA'14.