

Conference Reports

In this issue:

LISA '10: 24th Large Installation System Administration Conference . . . 55

Summarized by Theresa Arzadon-Labajo, Julie Baumler, Mark Burgess, Fei Chen, John F. Detke, Rik Farrow, Gerald Fontejon, Robyn Landers, Scott Murphy, Tim Nelson, Matthew Sacks, Andrew Seely, Josh Simon, Shawn Smith, Rudi Van Drunen, and Misha Zynoviyev

LISA '10: 24th Large Installation System Administration Conference

San Jose, CA
November 7–12, 2010

Opening Remarks and Awards

Summarized by Rik Farrow

Rudi Van Drunen opened the 24th LISA conference with the usual round of acknowledgements to the PC and USENIX staff. Van Drunen said that putting LISA together took him a couple of meetings and about +400 emails, with the staff handling setting up the conference. Then he announced the Best Paper awards. Fei Chen et al.'s "First Step Towards Automatic Correction of Firewall Policy Faults" won the Best Student Paper award, and Paul Krizak, of AMD, won the Best Paper award with "Log Analysis and Event Correlation Using Variable Temporal Event Correlator (VTEC)." Andrew Mundy (NIST) won the Best Practice and Experience Paper award with "Internet on the Edge."

Philip Kizer, President of LOPSA, announced the 2010 Chuck Yerkes Award winner, Edward Ned Harvey, for providing significant mentoring and participation in electronic forums.

Keynote Address

The LHC Computing Challenge: Preparation, Reality, and Future Outlook

Tony Cass, CERN

Summarized by Rik Farrow (rik@usenix.org)

Cass began by quipping that CERN had yet to destroy the universe, but if the many-worlds theory is true, perhaps the other 50% of the worlds have been destroyed.

Cass described some of the daunting requirements for operating the LHC. The LHC needs a vacuum with 10 times fewer particles than exist in the moon's level of vacuum. The superconducting coils that create the collider's magnetic steering fields must be kept at 1.9 Kelvin (-271 C). It was a failure in

cooling that forced the shutdown of the LHC last year. Cass showed images of what happens when the two beams of positively charged particles stray: a stripe of fused metal, and a hole through a solid copper plate used as a target. When two streams of particles collide, the energy is comparable to two trains colliding at 350 miles per hour.

The collisions are the whole point, and the LHC has 100 million data collectors. There are four detectors and 40 million collisions per second, producing 100–1000 MB/s, or around 23–25 petabytes per year of data. On site, they need to archive data, as well as reduce the data before it gets passed onto remote Tier 1 sites for further distribution and research. Cass went on to describe some of the challenges they have faced so far:

Capacity provisioning: together with other sites, the LHC is the world's largest-scale computing grid.

Box management: they use PCs with their own software (Quattro and Lemon) for node management.

Data management and distribution: over a gigabyte per second that must all be saved to tape, with enough redundancy for backup to fill three full SL8500 tape robots per year.

Network management: monitoring the flow to Tier 1 sites, as well as all the equipment used—they have 20 Gb/s links from CERN to Tier 1 sites.

LHC uses Oracle RAC (11g) pushed to the limits. Each day they run one million jobs on the grid, about 100,000 computer days per day, with a reliability of about 98%.

Failures are frequent, with about 200 failures a day, mostly disks. Infrastructure failures are a fact of life, but networks and software have proven reliable. They try to get 100% utilization using virtualization, which is fine for CPU-intensive apps, but expensive (10% penalty) for I/O-intensive applications.

In conclusion, Cass said that they had been preparing for these challenges since the late '90s, when solutions like Hadoop didn't exist and networks were very expensive. There were also sociological challenges, but they have been successful, supporting many thousands of people doing research.

Doug Hughes, of D. E. Shaw Research, asked about data integrity issues with so much data generated. Cass replied that they do use checksums and compress data. If data fails to decompress, they immediately know something is wrong. They also re-read tapes checking for errors. Mario Obejas of Raytheon wondered whether, since they were using Siemens SCADA (PVSS in German) software, they were affected by the Stuxnet worm. Cass replied that he didn't know, but that they have networks separated by firewalls. They are more

concerned with power plant issues, and thus carefully protect their control networks. Paul Krizak of AMD asked how Tier 1 sites know that data is correct. Cass responded that the experiments themselves provide software to do that. The Tier 1 sites also provide off-site storage for remote backups. Hugh Grant of the University of British Columbia asked about lessons learned. Cass said, "Don't believe what people say about requirements. They will underestimate things and over-complicate them. Use what you know, exploit what you can, make sure you can scale at least an order of magnitude over what they request."

Refereed Papers

Summarized by Fei Chen (feichen@cse.msu.edu)

A Survey of System Configuration Tools

Thomas Delaet, Wouter Joosen, and Bart Vanbrabant, DistriNet, K.U. Leuven

Thomas Delaet first identified gaps in the current state of the art and then presented a comparison framework for system configuration tools, which helps system managers decide which configuration tool should be bought for managing their system. There are a lot of such tools, with different purposes and characteristics, so it is very difficult to make a wise choice.

This work built a comparison framework including four categories of properties: properties of the input specification, properties of deploying the input specification, process-oriented properties, and tool support properties. In total, the authors defined 19 properties and, based on these, evaluated 11 existing open source and commercial system configuration tools and summarized their findings. The authors use this evaluative framework to provide guidance on choosing a tool and comparing tools.

Someone pointed out that this work requires the predefined workflow on top of the configuration. However, in general, for many configuration tools, there is no such workflow. Delaet responded that they have a scheme to define a language to define such workflow.

High Performance Multi-Node File Copies and Checksums for Clustered File Systems

Paul Z. Kolano and Robert B. Ciotti, NASA Ames Research Center

Paul Z. Kolano presented their design of mcp and msum, as well as detailed performance evaluation for each implemented optimization. The copy operation is one of the most common operations in computer systems. Because of backup, system restore, etc., files are usually being

moved from one place to another. Hence, maximizing the performance of copies as well as checksums for ensuring the integrity of copies is an important problem.

This work leveraged three major techniques to improve the performance of copies: multi-threading, multi-node cooperation, and hash trees. The authors' experiments show that mcp causes a more than 27-fold improvement in cp performance, msum improves md5sum performance by a factor of 19, and the combination of mcp and msum improves verified copies via cp and md5sum by almost 22 times.

Corral wondered about the user application of this work. Kolano replied that they hadn't deployed it for users yet and mainly have been using it to migrate users between file systems. Skaar (from VMware) asked about system overhead. Kolano said they hadn't specifically measured it, but showed an earlier slide where performance was identical to cp, indicating minimal overhead. Can this code be used for any systems? Yes, this is the general code.

Fast and Secure Laptop Backups with Encrypted De-duplication

Paul Anderson and Le Zhang, University of Edinburgh

Paul Anderson presented a fast and secure algorithm for backing up personal data on laptops or home computers. However, conventional backup solutions are not well suited for these scenarios in terms of security. The solution is really ad hoc. For example, people use external hard drive, DVD, or cloud storage to back up their data.

This work prototypes a new backup algorithm to back up personal data for Mac OS X. The algorithm takes advantage of the data that is common between users to increase backup performance and reduce storage requirements. The algorithm also supports two major functionalities. First, it supports per-user encryption, which is necessary for confidential personal data. Second, it allows immediate detection of common subtrees to avoid querying the backup system for every file.

Someone asked if they use the same key for the same file across different laptops. One user may reveal the files of the other users. Anderson said that's right. If a user has a file, it is possible to tell whether someone else has the same file (but not necessarily *who* has that file). Peter asked how file permissions are handled. Anderson answered that file permission attributes are separated from the files themselves. Why not allow the server to do the encryption? The primary requirement is to not allow the server to know the data.

Invited Talks I

IPv6: No Longer Optional

Richard Jimmerson, ARIN

Summarized by Julie Baumler (julie@baumler.com)

Richard Jimmerson started by explaining that he was going to cover IPv4 depletion, including when it would occur, why, and a number of related issues. He explained that his expertise comes from his experience at the American Registry for Internet Numbers (ARIN), one of five regional Internet registries (RIRs). He started with some historical background: in the mid-'90s people realized that IPv4 is not designed for the global commercial Internet. He mentioned that IPv6 addresses have been issued since 1999, and this became a recurring theme in his talk. The primary factor driving IPv6 adoption is the pending full depletion of IPv4 addresses. As of October 18, 2010, there were 12 /8 blocks containing 16 million addresses each. Registries are issued one or two /8s at a time by the Internet Assigned Numbers Authority (IANA), which holds the free pool for all five RIRs. There is currently a large demand in the Asia-Pacific region and large numbers of existing devices which could use IP addresses but aren't. It is very likely that the free pool will fully deplete in the first quarter of 2011. ARIN expects their free pool to deplete within 1 day to 6 months from the IANA free pool depletion date. The top 10 Internet service providers could deplete a /8 in one day with legitimate requests. Additionally, ARIN will set aside a /10 from the last /8 they receive from IANA to be used only for IPv6 transitions (i.e., to allow new organizations to have IPv4 gateways and things like DNS servers), and these will be allocated in /28 to /24 blocks only.

Jimmerson acknowledged that the end of IPv4 addresses has been announced before and did not happen. CIDR and NAT saved us from depletion in the '90s and there were some false "cry wolf" statements early in this century. He emphasized that this depletion is for real.

A common question is how underutilized blocks from the 1980s and 1990s will affect depletion. ARIN is constantly trying to reclaim them. They are still getting class As and Bs back, but that won't extend the depletion date by much, as an /8 only extends the free pool by a few weeks. There is also a new policy that creates a market in IPv4 addresses by allowing specified transfers of IPv4 addresses from organizations that aren't using them to organizations that meet existing requirements for issue of IPv4 addresses. ARIN has created a limited listing service to support this, and there are already listings.

Another important issue to keep in mind is that IPv4 and IPv6 will both be necessary for many years. For instance, all

content is currently on IPv4 and it will need to be made available to users of both IPv4 and IPv6.

There is currently very little visibility of IPv6 deployment. Jimmerson primarily attributed this to very low incentive for people to share what they are doing. People don't want to publicly be the first. Also, in many cases there is a potential for a huge market share win if a company supports IPv6 and their competitors don't. This means that many people are not deploying IPv6 visibly or are not marketing the fact that they have done so.

Jimmerson outlined a number of different issues and ARIN recommended action plans for different sectors of the IP address-using community, such as broadband providers, ISPs that provide services to business customers, content providers, and equipment vendors. Some common threads are the need to be ready and the need to have content and core services such as email available on both stacks. More details on these recommendations are available in the resources recommended below. ARIN has also been involved in raising awareness of the issues in the government arena.

Jimmerson recommended some resources for further information: <http://TeamARIN.net> includes a free slideshow and other information to use for educational purposes, and <http://getipv6.info> is a wiki that includes deployment experiences and information on where to get started. He also mentioned the social media links at <http://www.arin.net>. Jimmerson emphasized that anyone is welcome to participate in ARIN at no cost; further information about this is available at <http://www.arin.net/participate>.

Several questioners asked how IPv6 would affect security, system administration skills, and compliance. Jimmerson pointed out in each case that although IPv4 and v6 will form logically separate networks in most cases, the tools and issues are the same. He recommended that system administrators shouldn't be afraid of IPv6; they should just get educated and start playing around with it and testing it.

Someone asked whether there is anything that will preclude using NATs forever. Jimmerson acknowledged that you can build NATs on top of NATs and it will happen, but it's going to get messy, and at some point you will find data that people prefer to serve or receive over IPv6. This will be particularly true for latency-sensitive data. Another questioner asked how we get rid of IPv4 altogether so that we don't have to run both protocols forever. Jimmerson said that the most difficult part of this is that there is no flag date. He feels that for IPv4 to disappear, 99.9% of content will need to be available on IPv6 and you will be able to buy all types of IPv6-supported network equipment in stores. There are working groups coming up with suggested dates for this transition.

Invited Talks II

Storage Performance Management at Weta Digital

Matt Provost, Weta Digital

Summarized by Rik Farrow (rik@usenix.org)

Weta Digital is best known for the *Lord of the Rings* trilogy (LOTR) and *Avatar*. The first LOTR movie required 1.5 TB of storage total, while they had that much RAM when making *Avatar*. Provost suggested that people see *Avatar* in theaters, as the image is 298 GB, compared to just 4 GBs on DVD.

Weta Digital has a grid of 4700 servers, what they call a "renderwall." Producing *Avatar* required 56 million computer hours, or about 11 days of computing per frame. Rendering is the step that takes the work of artists and turns it into finished frames. Artists' work is the most expensive commodity, and providing file storage that performs well is key to creating a movie.

They use Perl scripts based on templates to create new directories for each shot. Shots themselves get spread out over filers to avoid hot spots. While a shot's frames may appear to be in a single directory, they use symbolic links to manage storage behind the scenes. They created a system called DSMS to build the link farm and use a MySQL database to store this info. The file system remains the canonical view, not the database.

Provost mentioned that some people wondered why they don't use Lustre, and he explained that Lustre 2.0 came out in 2002 (LOTR began in 1999) and requires a lot of space for metadata storage. They had 3.2 million files for *Avatar*, and most of those files are only 64kb, so the system they use has lower metadata overhead.

Running out of space on a filer causes serious slowdowns, so they monitor disk space. They also reserve space on NetApps (used by artists) and use a script to migrate data that is acquiescent (based on atime) when needed, and change the symlinks when this is complete. NetApps FlexCaches were brought in to help with performance during *Avatar*. They did use flash as well.

Performance is monitored on clients by watching `/proc/self/mountstats`, and they can trace bottlenecks back to filers by using the pathname combined with queries to the link farm. Provost pointed out that what they had was a combination of HPC and HA. Even artists' workstations are used for rendering at night, and they can't afford downtime.

Provost mentioned that while *Avatar* was shot in high definition, better cameras and 3D will mean that frame sizes may grow from 12 MB/frame to 84 MB/frame. Higher frame rates

and 3D also add to storage demand, so one second of a future movie may require 8 GB.

Don Johnson of NetApp thanked Provost for helping send his kids to college and then asked about the difference between BlueArc and NetApp filers. Provost replied that BlueArcs are really good for write performance, which is required to keep up with the renderwall. NetApp filers are the only thing they trust with their human-generated data. Deke Clinger of Qualcomm wondered if they had problems with the Linux versions of NFS and the automounter. Provost said that they are still using version four of the automounter, although they have their own fork. They can get big mount storms when rebooting the renderwall. They always use TCP with NFS. Jim Kavitsky of Brocade Communications asked if they track communications on the network, and Provost said that they do. They also store this in a database, so they have a historical record. Matthew Barr of MarkitServ wondered if they have looked at pNFS, and Provost said they have already solved a lot of the problems pNFS tries to solve, and that pNFS tends to work best with larger files.

Refereed Papers

Summarized by Fei Chen (feichen@cse.msu.edu)

The Margrave Tool for Firewall Analysis

Timothy Nelson, Worcester Polytechnic Institute; Christopher Barratt, Brown University; Daniel J. Dougherty and Kathi Fisler, Worcester Polytechnic Institute; Shriram Krishnamurthi, Brown University

Timothy Nelson presented the Margrave tool for analyzing firewall policies. Configuring and maintaining firewalls is always a challenging and difficult task, due to the complexity of firewall policies. It is very useful to develop a tool that can help sysadmins to configure and maintain firewall policies. This work describes Margrave, a powerful tool for firewall analysis, e.g., change-impact analysis, overlaps and conflicts detection, and security requirement verification.

Margrave embraces both scenario-finding and multi-level policy-reasoning in its model. It divides a policy into small policies and then analyzes each small policy. Therefore it provides more exhaustive analysis for richer policies and queries than other tools. Timothy Nelson presented the evaluation results on both network-forum posts and an in-use enterprise firewall.

Someone asked: Are you looking at analyzing the routing table? Nelson: Yes, we do want to do that. Matt Disney: Can you say more about how you guarantee exhaustiveness? Nelson: Not all Margrave queries result in simple scenarios like the ones we saw; in those cases we may still be able to make

guarantees, but if not, the user can provide a size that they want to check up to. Disney: It would be interesting to collect the system logs. Then sysadmins do not need to query the firewall manually. Nelson: This is a very interesting idea. We may look into this idea.

Towards Automatic Update of Access Control Policy

Jinwei Hu, University of Western Sydney and Huazhong University of Science and Technology; Yan Zhang, University of Western Sydney; Ruixuan Li, Huazhong University of Science and Technology

Jinwei Hu, who had to record his presentation on videotape because of a visa issue, presented RoleUpdater, a tool for updating access control policies automatically due to new security requirements. Manually updating access control policies is tedious and time-consuming. Updating is a key component of maintenance in the RBAC life-cycle. RoleUpdater is a very useful tool for sysadmins to manage their access control policies.

The key idea of RoleUpdater leverages model-checking techniques to update the RBAC policies. RoleUpdater first transforms update problems into a model-checking problem. Then a model checker takes a description of a system and a property as inputs and examines the properties of the system.

There was no Q&A, because the authors were not present.

First Step Towards Automatic Correction of Firewall Policy Faults

Fei Chen and Alex X. Liu, Michigan State University; JeeHyun Hwang and Tao Xie, North Carolina State University

► *Awarded Best Student Paper!*

Fei Chen presented an approach for automatically correcting firewall policy faults. Wool's studies have shown that most firewalls are poorly configured and contain faults. Manually checking each rule in a firewall policy and further fixing the fault is a difficult problem and impractical because a firewall policy may consist of thousands of rules.

This work first proposed a fault model of firewall policies, which includes five types of faults: wrong order, missing rules, wrong decisions, wrong predicates, and wrong extra rules. For each type of fault, Chen presented a technique to fix it. Then Chen presented a greedy algorithm that utilizes these five techniques to fix the firewall policies faults automatically.

Tom Limoncelli: Is it possible to use some AI system to automatically filter the packets? Chen: To the best of our knowledge, there is no such AI system, due to security requirements. Matt Disney: What further work is planned?

Chen: We are looking at applying our approach to a faulty policy repeatedly and seeing how much we can fix the policy.

Invited Talks I

Storage over Ethernet: What's in It for Me?

Stephen Foskett, Gestalt IT

Summarized by Theresa Arzadon-Labajo (tarzadon@ias.edu)

Stephen Foskett's entertaining talk, sprinkled with anecdotes and jokes, provided a lot of information about storage over Ethernet. Foskett began by saying that convergence is the marketing topic, and another trend is the rise of open systems. Finally, even though IP and Ethernet have been around a long time, they are becoming the "must-haves" of IT.

There are a few reasons why convergence is happening. First, virtualization is the biggest driver of storage. Systems were optimized for sequential I/O, but virtualization throws it in the I/O blender. Storage companies preach the message of virtualization, consolidation, and converged networking because they can sell a lot of SAN gear. Secondly, there is consolidation from a port count perspective. Converged networking allows you to deal with the spaghetti problem. You can allow your servers to breathe; all those cables interfered with air flow. The mobility of virtual machines allows you to move a running system somewhere else and it can still be the same system. You can't do that with conventional cables. Third, performance is driving convergence because of all the applications that need massive I/O.

Stephen showed graphs displaying the trends of Fibre Channel (FCP), Ethernet LAN, iSCSI, Fibre Channel over Ethernet (FCoE), and Ethernet Backplane. Everything seemed to outperform Fibre Channel, which means it will eventually get left behind. In order to make Ethernet handle storage, the Data Center Bridging project created new protocols: Priority Flow Control (PFC 802.1Qbb), Bandwidth Management (ETS 802.1Qaz) and Congestion Management (QCN 802.1Qau). If all these things can be accomplished, Ethernet could be a decent protocol and SCSI traffic could travel over it. Contrary to Ethernet's PAUSE (802.3x), PFC allows the stop message to be applied to only one class of service and lets other traffic keep going. iSCSI doesn't need this, because it has TCP. Enhanced Transmission Selection (ETS) allows you to reallocate channels in a converged network to different applications. Switches weren't built to handle this, so another protocol was needed. Data Center Bridging Exchange (DCBX) allows devices to determine mutual capabilities. Congestion notification is not standardized yet, but it's in the works. Theoretically, it will allow end-to-end traffic management. There will be a pause in the beginning, but once both

ends are informed, traffic flows nicely. In the real world, you can double or triple throughput.

Stephen compared the FCP, FCoE and iSCSI protocols. iSCSI works great, is robust, is mature, and every OS is supported on the client side. Also, there is a nice transition from 1-10GbE. All you have to do is plug in a new cable! Pros of iSCSI are its performance, functionality, low cost, and availability. Cons are that you may want 1GbE for performance or else have an FC Estate. Reasons to go with FCoE are that you have a lot of Fibre Channel and you might want to make better use of that. You can incrementally adopt it and go with end-to-end FCoE later. You may want to consolidate on Ethernet and not want to buy iSCSI licenses and arrays. I/O consolidation and virtualization capabilities are focusing on FCoE, and vendors are pushing this hard. Cons of FCoE are the continued bickering over protocols, the fact that we already have 8Gb FC, and end-to-end FCoE is basically non-existent, unproven, and expensive.

Stephen briefly talked about NFS. NFSv4 pretty much fixes all the issues with NFS. Vendors support it and there are drivers for it, but hardly anyone uses it. The good thing about it is that it is one protocol with a few commands instead of several protocols with thousands of commands. Plus, there's no UDP! pNFS (v4.1) provides file, block, and object storage and is focused on scale-out.

Server, network, and storage managers each get something different out of converged networking. Server managers win because they don't have to care about storage anymore. They have better support for virtual servers and blades. Network managers get all the headaches, because they have to learn a whole new world of protocols and deal with storage, but they get more tools and can segment the network. Storage managers lose, because everything outside the array is taken away from them. But this can make them concentrate on the real problem with storage, which is that once people write data, they never read it and never delete it.

Stephen gave a counterpoint to Ethernet by stating that InfiniBand already exists, is supported, and is faster. Fibre Channel is kind of pricey and insane, so you might as well go with something that is fast and insane. He proposed that we should go with something else entirely, like Fibre Channel over Token Ring (FCoTR). It's already lossless and the packets match up. He concluded that Ethernet will come to dominate. iSCSI is growing, Fibre Channel is continuing, and NFS is still here and it's all over Ethernet.

Someone asked about the availability of dense 10GbE switches. Stephen suggested looking at what Force10 and Arista have going. Someone else asked how to help out with the growth of FCoTR. Stephen said that there's not much you

can do but have fun with it. FCoTR makes as much sense as FCoE. If we're doing one of them, why not do both?

The 10 Commandments of Release Engineering

Dinah McNutt, Google

Summarized by Gerald Fontejon (gerald.fontejon@gmail.com)

Dinah McNutt said that these 10 commandments are from sysadmins to release engineers, and that the commandments are solutions to requirements. She also stated that the title of the presentation should be "Build and Release." The ideals from this presentation are for all types of software, internal and external customers (Web applications and shrink-wrapped products). Dinah also said that the ideals from the presentation are her own, not necessarily her employer's.

Usually the release process is an afterthought, and the process is minimally managed to "get it done." Release processes should be treated as a products in their own right, and the release process should be a bridge between developers and the system administrator who implements the release. The build and release steps are: (1) check out the code from the source code repository; (2) compile the code; (3) package the results; (4) analyze the results / report accordingly; (5) perform post-build tests based on the results of the analysis steps (i.e., smoke tests, unit tests, system tests)

There is a set of required features within the build and release process: the process should be reproducible, have a method of tracking the changes, and have the ability to audit what is in a new version of the product. Within each build, there has to be a mechanism that uniquely identifies (e.g., a build ID) what is contained in a package or product. The build and release process should be implemented as part of a policy and procedure, and if the automated build and release process has been bypassed, there has to be some documented reason why the process was disrupted. Included in the build and release process is the management of upgrades and patch releases.

Dinah laid out her 10 commandments:

- I. Thou shalt use a source code control system.
- II. Thou shalt use the right tool(s) for the job.
- III. Thou shalt write portable and low-maintenance build files.
- IV. Thou shalt use a build process that is reproducible.
- V. Thou shalt use a unique build ID.
- VI. Thou shalt use a package manager.

VII. Thou shalt design an upgrade process before releasing version 1.0.

VIII. Thou shalt provide a detailed log of what thou hath done to my system.

IX. Thou shalt provide a complete install/upgrade/patch/uninstall process.

X. System Admin: Thou shalt apply these laws to thyself.

On her last slide, Dinah showed how the build and release process relates to system administration. She said, "I think a lot of these concepts apply to system administration and other disciplines, not just software engineering—because my thoughts are bits—and release engineering is all about taking those bits and figuring out a reliable way of delivering them where they need to go."

Paul Krizak asked, "What are your thoughts on some of the newer packaging systems? In particular, I'm thinking of rPath, which takes the build process beyond just making binaries, and builds the entire runtime from the operating system all the way to the application, all in one shot. Do you think that is moving in the right direction? Or is that overkill?" Dinah replied that it depends on the environment that you are working in. She added that it could certainly be overkill, but she also believes there are a lot of applications and situations where it could be beneficial.

Someone asked about deploying applications and dependencies in a Web application—what are the recommendations for the server-user-ID to be used for the release process and its associated location? Dinah replied that the less you can do as root, the better. The subject of location goes back to the discussion on relocatable packages. "I could install the software anywhere and it's going to work."

Practice and Experience Reports

Summarized by Rik Farrow (rik@usenix.org)

When Anti-virus Doesn't Cut It: Catching Malware with SIEM

Wyman Stocks, NetApp

Stocks explained that Security Information and Event Management (SIEM) dumps all your logs in and does event correlation, helping to make sense of 50 million events a day. He found that it really helped having SIEM when systems on their network became infected with Conficker.

Someone outside had noticed the worm traffic and informed them. They immediately rolled out patches, thinking that with AV and SIEM they were okay, but the problem per-

sisted. They started by manually notifying users, but after two weeks they had SIEM send out emails with instructions for cleaning up. They were seeing 30–50 machines a day infected at first, down to 4–11 after two weeks of automated alerts. They sent samples of the infections to McAfee, and saw more than just three Conficker variants.

Someone asked about disabling switch ports, and Stocks responded that people get really upset when you just disable their network port. Someone else wondered which Conficker variants they had, and Stocks said mostly B and C, as variant A was mostly caught by AV. The same person asked about the rules they were using with SIEM to discover infections, and Stocks said they had IDS rules for distinguishing command and control traffic over HTTP, and would look for 445/TCP (CIFS) scanning.

Stocks summarized their lessons learned: they needed to synchronize time across the enterprise, so logging timestamps matched; short VPN connections made it difficult to find infected users; when the volume of infections dropped, the false positive rate increased; finally, you will learn things about your network that may not be that useful. In the future they want to add more preventive measures, such as having DNS black holes for C&C servers, new firewall rules, better network visibility, and historical look-backs to determine attribution.

Matt Disney of Oak Ridge National Laboratory asked if SIEM has replaced any currently used security controls. Stocks answered that SIEM gave them capabilities they didn't have before and added that Windows Domain Controllers rotate logs so rapidly they quickly lose information if it isn't collected in SIEM. Disney asked what features to look for when SIEM shopping. Stocks suggested finding products that can parse events (and aren't limited to the same vendor's products) and are easy to use. You want a balance between flexibility and usability.

In-Flight Mechanics: A Software Package Management Conversion Project

Philip J. Hollenback, Yahoo, Inc.

Hollenback is the release manager for Yahoo mail (philiph@yahoo-inc.com) and led a team of six to convert over 7000 distributed servers for Yahoo mail to Igor. The goal was to upgrade server software with no downtime and to do this repeatedly. The user mail servers are grouped in farms, and each user's email lives on one farm, with hundreds of thousands to millions of users on each farm.

Yahoo has developed its own in-house software installation system, yinst, which is both a packaging system, like RPM, and installation software, like yum or apt-get. Upgrades were

made by sshing into a system and executing yinst to install packages. Igor is a state-based package management system already successfully used elsewhere within Yahoo. Hollenback said they thought that all they needed to do was to get Igor working with yinst, but as they worked on the project they discovered several problems.

One problem is that packages had been installed additively in the past, with the expectation in some cases that some key software would just be there, like Perl or a Perl library. Another issue was that they want to have a single set of packages, so configuration needed to be separate from packages. Finally, they also discovered that each farm could have unique, or local, configurations, which had to be documented before they could proceed. Hollenback found himself surveying farms looking for these differences.

In hindsight, Hollenback said they needed to have started with good audit tools to uncover the existing configurations. They also needed other tools, such as pogo, a type of parallel ssh that works as a push tool. Moving forward, they are still working to remove configuration from packages, improve yinst settings, and add central configuration servers. They can roll back upgrades, but this needs to be smoother, and removing configuration from packages is making this easier. Summarizing, Hollenback suggested keeping things simple: install the same packages everywhere, don't inherit system state, use configuration servers, and, basically, don't be too clever.

Paul Krizak of AMD asked about the scalability of upgrades, and Hollenback answered that the nice thing about the system is that it is well distributed. They have software distribution machines in every colo and they have reached the point where they can launch 10k machines at once. Krizak asked about the human element, the pushmaster who watches over upgrades. Hollenback said that is a problem, but pogo helps by doing lots of health checks before and after installs. Hugh Brown of UBC asked about not inheriting system states, and Hollenback explained that this means use package and configuration data, not a system's past state. Each machine has particular roles, and the packages and configuration control which roles. Matthew Sacks wondered if it would have been better to have improved auditing tools early on, and Hollenback said that they need to audit existing systems to see how they were configured. Now they have Igor and it provides the exact set of packages and settings.

Experiences with Eucalyptus: Deploying an Open Source Cloud

Rick Bradshaw and Piotr T Zbiegiel, Argonne National Laboratory

Bradshaw, a mathematics specialist, and Zbiegiel, security, co-presented this experience paper, with Bradshaw starting. They had both been involved with the Magellan Project, a medium-sized HPC, and were charged with discovering if clouds could work for HPC workloads. There are many commercial clouds to choose from, but they decided to work with Eucalyptus, as it is open source, compatible with Amazon's EC2 and works with Ubuntu Enterprise Cloud (UEC), and they could run with patches on top of the usual Ubuntu.

Zbiegiel explained a little about how Eucalyptus works: cloud controllers communicate with cluster controllers and storage controllers, which sit above node controllers (each VMM). There is also another tool, Walrus, which works only with storage controllers. They experimented with different cluster sizes and found that 40–80 nodes worked best, since Eucalyptus can get bogged down as it sends out commands serially and got “impatient” if responses to commands were slow. There was a hard limit to the number of VMs, somewhere between 750 and 800.

Zbiegiel explained that they had two security concerns: networking and images. By default, VMs can talk to any IP address and can also masquerade as cluster controllers, so it was difficult to tell who might be doing something bad. They needed to see if outside machines were attacking or their own VMs were scanning or attacking or running suspect services. They used iptables to control where VMs could connect, and monitored all traffic as it passed through cluster controllers.

Any user can upload an image which becomes visible to everyone in the Eucalyptus cloud, and this is the default. Zbiegiel wishes the opposite were the default. Also, sysadmins can install ramdisks and kernels, and this can be a source of problems as well. Every user on Eucalyptus is a sysadmin, no matter what their actual level of experience is.

Bradshaw explained that they had chosen community-based support because they had only one sysadmin to manage the Eucalyptus clusters. This meant wikis, mailing lists, and best-effort documentation. They discovered that there is a big difference between batch users and cloud users, as cloud users need to support the entire OS. The learning curve for users is steep. He concluded by saying that they do have a cloud and also have a small Nimbus (NASA) deployment and are looking at OpenStack (an open source combination of Rackspace and Nimbus software). He suggested that you shouldn't believe the cloud hype, that clouds are useful, but every stack has its qualities and faults.

Someone asked about system monitoring and adding servers. Bradshaw answered that Eucalyptus does no monitoring, except of user-facing front ends. Setting up new servers can be done using any kind of distributed build process, added Zbiegiel. Chris Reisor of Dreamworks asked where images are stored, and Zbiegiel replied that they are stored in Walrus, the Eucalyptus version of Amazon S3. You create a bucket for each image. Reisor then asked how well Eucalyptus does when things go wrong. Zbiegiel said that it depends; sometimes they can recover, but they have seen it fail in many more fantastical ways that require bouncing (rebooting) the entire cluster.

Invited Talks I

Commencing Countdown: DNSSEC On!

Roland van Rijswijk, SURFnet Middleware Services

Summarized by Rudi Van Drunen (rudi-usenix@xlexit.com)

Roland started off with some of the attack vectors used to attack the DNS system, which sparked the development of DNSSEC. DNSSEC provides authenticity to DNS records by adding digital signatures to the records and validation of those signatures in resolvers. We see that the adoption of DNSSEC has been on the rise since the root of the DNS system got signed.

Roland described how most of the resolvers currently in use already support DNSSEC. To get started with a validating resolver, a good tool to use is unbound (<http://unbound.net>). As DNSSEC uses public key cryptography, it uses more CPU power, but the impact is negligible. Roland continued by discussing how you have to be pretty careful in your setup in order to run a signed zone. Ideally, setting up a DNSSEC signed zone should be as easy as setting up a normal zone. Surfnet has integrated this in their DNS self-service environment. The infrastructure they use is OpenDNSSEC and a hardware crypto box/key store.

There have been a number of quirks in the recent past due to DNSSEC signed zones that were not operated correctly; these have led to serious outages of parts of the DNS system, so sysadmins and operators need to be aware of the additional issues that DNSSEC brings.

Some pointers to additional material: <https://dnssec.surfnet.nl>; <http://dnssec.net>; <http://www.dnssec-deployment.org>; <http://www.practicesafedns.org>.

Roland concluded with the following key points:

As DNSSEC deployment really is taking off, you are the one who has to act, by seriously considering enabling validation of signatures in your resolver. Then think about signing your

zones. Mistakes can (and might) happen; please learn from them. And, last but not least, if it works, you don't notice it's there.

How do you protect your laptop? Use a validating resolver on your end-user system (e.g., Unbound). How does that work with captive portals or other nasty DNS tricks? It will not work, so switch off your validating resolver and fire up your VPN, routing all traffic through your home office.

Invited Talks II

Postfix: Past, Present, and Future

Wietse Venema, IBM T.J. Watson Research Center

Summarized by Scott Murphy (scott.murphy@arrow-eye.com)

Venema observed that publicity can be both bad and good for a project. He reminded us of his past security research, specifically an unflattering 1995 *San Jose Mercury* article likening SATAN (Security Administrator Tool for Analyzing Networks) to “distributing high-powered rocket launchers throughout the world, free of charge, available at your local library or school.” This was in contrast to the release of Secure Mailer (Postfix) in December of 1998, which was accompanied by a *New York Times* article titled “Sharing Software, IBM to Release Mail Program Blueprint.” This was the fourth official IBM involvement in open source between June and December of 1998 and is recognized as the one that caused IBM management to realize that there was no existing open source strategy. A mandate to develop one ensued, leading to the 1999 announcement of an IBM open source and Linux strategy.

So why create another UNIX mail system? As a practical exercise for secure programming, this would be an ideal project. Venema displayed an architectural diagram of the Sendmail program and its monolithic model—the two programs Sendmail and mailer. Highlighted was the fact that root privileges are required for the system to perform its tasks. This was followed by a number of slides listing the CERT advisories on Sendmail and the `/bin/mail` program over a 15-year period. Two major observations are that one mistake can be fatal and result in privilege escalation and that there are no internal barriers to compromise. This leads to a system that is hard to patch and in which it's hard to determine the side effects of a patch. The Postfix model was then shown, showing three major blocks: input, core, and output, similar to an Internet router. The only points at which elevated privileges are required are at local delivery time and to send to an external transport. Major influences on the design included the TIS Firewall, qmail, Apache, Sendmail, and network routers.

After the architectural overview, Venema went on to some of the considerations in the implementation of Postfix. If you know that your error rate is 1 in 1000 lines of code and that Postfix was 20,000 lines of code, you see you are releasing 20 bugs. Postfix is about 120,000 lines of code now, so perhaps 120 bugs. You want to control this, and the distributed architecture reduces the impact, with fewer bugs per component. Optimization is a special case, as Internet-facing servers have the problem of the worst case becoming the normal case and vice versa. Venema was told to “just implement SMTP without screwing up.” As there are only a few commands in SMTP, how hard can it be? Well, multi-protocol, broken implementations, concurrent access, complicated addressing syntax, queue management, SPAM and virus control, and anti-spoofing systems quickly turned up the difficulties.

The official strategy was to divide and conquer by implementing a partitioned “least privilege” architecture, use (mostly) safe extension mechanisms, and let third parties provide the external applications. Several examples were then given, along with supporting architectural diagrams. As a final example, the implementation of Sendmail Milter into Postfix was shown, along with the press release from Sendmail Inc. awarding Dr. Venema a Sendmail Innovation Award for his contribution of extending Milter functionality to the Postfix MTA.

Over the years, Postfix has grown in size from its modest beginnings. Urged on by a friendly comment on the size of the Postfix source file, Venema decided to do an analysis of Sendmail, Postfix, and qmail source code. In order to accomplish this, comments were stripped (reducing Postfix by 45%), format conformed to “Kernighan and Ritchie” style (expanding qmail by 25%), and repeating (mostly empty) lines deleted. A graph showed that Postfix steadily grew up until it was considered officially “complete” in late 2006, after which it tapered off to a significantly slower rate. It surpassed Sendmail in combined size in 2005, while qmail has been essentially flat since its initial release. Venema attributes the lack of bloat in Postfix to the partitioned architecture, asserting that small programs are easier to maintain. Minor features can be added through modification of a small program, major features by adding a small program (for interesting values of small). Currently, Postfix consists of 24 daemons and 13 commands, with the SMTP daemon weighing in at almost 10k lines, or approximately half the size of the initial Postfix alpha release.

You can't really talk about success without including market share. This is a rather inexact item, as the number of mail servers in the wild isn't easy to determine, nor does it accurately reflect the actual users. In 2007, O'Reilly did a fingerprinting study of 400,000 company domains to

determine the mail servers in use. At the time, Sendmail was number one at 12.3%, followed by Postfix at 8.6%. Ten systems were on top, accounting for 65% of the results, other systems accounted for 20%, and there were 15% unknown. Using Google to search for mailserver query volume over time, we get a slowly declining graph for four of the open source servers: Sendmail, Postfix, qmail, and exim. What does this actually mean? We don't know, but Postfix queries exceeded Sendmail queries back in 2006. Today, they are all close together near the bottom of the curve. Searching Google trends has illustrated this as well. Tweaking the search terms in order to reduce result pollution has also shown a decrease in queries on MTAs over the years. This leaves only a couple of conclusions—the results are only as good as the queries, and only a declining minority of users are actually interested in mail servers.

Over the years, the essentials of email have changed significantly. Back in 1999, you built an email system on UNIX, so you did not have to worry about Windows viruses. New problem—your UNIX-based email system is now a distribution channel for Windows malware. New solution—out-source the content inspection to external filters. In 2009, you built a mail system that had world-class email delivery performance. New problem—your high-performance email system is spending most of its resources not delivering email. New solution—work smarter. Venema displayed a new chart showing some research from the MessageLabs Intelligence report for August 2010 indicating that 92% of mail is spam, 95% of which is from botnets. Zombie processes keep ports open, resulting in server ports being busy with nothing and not accepting email. RFC 5321 recommends a five-minute server side timeout which Postfix implements. Zombies own your server. If we assume that the zombie problem will get worse before it gets better, we have some options: spend less time per SMTP connection, handle more SMTP connections, or stop spambots upstream. This third option is slated for release in Postfix 2.8 in early 2011.

The new component, called postscreen, is designed to reject clients that “talk too fast” or make other blatant protocol violations and to utilize greylisting. It also uses black-and-white lists as shared intelligence to decide if it's talking to a zombie, as zombies tend to avoid spamming the same site repeatedly. Venema then displayed the workflow diagram for postscreen, going on to describe the initial connection for SMTP and how to detect spambots that speak too early, using the question, “How does a dogcatcher find out if a house has a dog?” Answer: He rings the doorbell and listens for a dog to bark. Postfix does this with zombies. Good clients wait for the full multi-line greeting, whereas many spambots talk immediately after the first line. He then showed charts

illustrating the pre-greet events at two European sites, followed up by some charts on spam load and time of day. This varies by receiver and time of day. A few more charts showed this, with the USA and China displaying atypical patterns from the rest of the samples. Pilot results for small sites (up to 200k connections/day) show detection via pre-greeting of up to ~10% of sites not on the DNS blacklist and an additional ~1% that pipeline commands. Additional protocol tests will be developed as botnets evolve.

Venema wrapped up the talk with a conclusion that reiterated some lessons learned: Don't underestimate good PR—it has enormous influence; don't waste time re-inventing—good solutions may already exist; build your application with stable protocols—use established standards; use plug-ins for future proofing—accept change as a given; optimize both the worst case and the common case—these tend to swap positions as you go; and, finally, don't let a C prototype become your final implementation.

He observed that Postfix has matured well, establishing that a system implemented by a series of small programs is extensible by either a small change or adding an additional small program. Extensibility is a lifesaver, as it means not everything needs to be solved initially, but can adapt over time. While Postfix may be considered stable and released, the battle continues. New technologies on the roadmap will assist in the fight to keep zombie loads under control.

Bill Cheswick (AT&T Labs—Research) asked, “What language would you use instead of C?” Venema answered that the original plan was to do something he had done before and implement a safe language in C, and this safe language would be used to configure the system. It was described as a simplified version of a Perl-like configuration language implemented in C. Unfortunately, he had to first understand enough of the problem and build enough to get mail to and from the network and to handle delivery and local submission. Norman Wilson (OCLSC Calico Labs) said that he always believed that the hallmark of a successful programming system is not how easy it is to extend but how easy it is to throw things out. Have you done this in Postfix? Also, do you feel that building out of cooperating pieces makes it easier rather than harder, not technically but culturally? Venema replied that, in principle, you can throw things out, as it's a loosely coupled system. A couple of things have happened. First, LMTP is a protocol similar to SMTP. At some point LMTP was forked from SMTP and evolved separately for several years. Eventually it was just too much trouble to support both, so he forcibly merged them, effectively discarding a protocol. Second, Postfix uses a table look-up interface for everything. If it's simple strings, use Berkeley DB; if it's tricky, use regular expressions, which is not a

great user interface but will do almost anything. He still has notes about an address-rewriting language that would have replaced the trivial rewrite daemon in Postfix, but this will probably never be written. Monolithic vs. several programs is hardly an issue.

Refereed Papers

Summarized by John F. Detke (jdetke@panix.com)

Using TCP/IP Traffic Shaping to Achieve iSCSI Service Predictability

Jarle Bjørgeengen, University of Oslo; H. Haugerud, Oslo University College

Jarle Bjørgeengen presented work applying TCP/IP packet-shaping methods to SAN traffic in an effort to achieve predictable service response.

The problem is that in the common SAN configuration, free competition for resources (disk operations) among service consumers leads to unpredictable performance. A relatively small number of writes had a large impact on read times. Most applications behave better with predictable, if slower, read performance.

The test setup consisted of four blade servers connected to an iSCSI SAN, with ipfiltering providing the ability to control the data flows. The number of random readers and sequential writers could be controlled while capturing and plotting performance data such as average read or write times and throughput.

Various throttling mechanisms were tested and it was found that adding a delay to the ACK packets resulted in a linear increase in read time as the delay was increased. The optimum delay varies with the workload, so setting the delay to a fixed value is not a good option. Manually adjusting the delay is not practical, due to the dynamic nature of real workloads.

Automating the delay throttling with a modified proportional integral derivative (PID) algorithm was investigated. This turns out to be an efficient method for keeping low read response times with an unpredictable and dynamic write load.

A demo was given showing the software running in the lab, graphing performance data in real time while the workload was varied. First, Jarle demonstrated that adding modest write workloads has a large negative impact on read operations. Next he showed how adding the ACK delay improves things. Finally, we saw how using the PID algorithm to automatically adjust delay results in predictable read responses times.

In summary, common packet-shaping techniques using available tools can be used to automate controlling IP-based iSCSI traffic to provide predictable and thus improved behavior in a SAN environment. Packet delay proves to be a better control mechanism than bandwidth limiting at ensuring fair resource sharing with multiple clients. Future work includes moving the control mechanism outside the iSCSI array to create an appliance that could be used on different arrays without needing details about the array itself. Additional throttling algorithms are being investigated to see if even better results are possible.

YAF: Yet Another Flowmeter

Christopher M. Inacio, Carnegie Mellon University; Brian Trammell, ETH Zurich

Christopher M. Inacio started with a short tutorial and history of NetFlow, including the basic data available, its historical roots in billing, and how it can help with security investigations.

Why build Yet Another Flowmeter? The authors wanted a tool that was compliant with IPFIX, could capture both talker and receiver, performed well, could do weird layer 2 decoding such as MPLS encapsulated on Ethernet, and had an open design that allowed for enhancements.

The basic architecture of YAF was described, along with the various methods available for capturing data. These range from high-speed cards to reading previously generated pcap (packet capture) data. A condensed IPFIX primer followed, discussing data structures and how templates are used to conserve bandwidth and storage requirements. YAF fits between traditional header-only NetFlow data and complete packet capture tools. Options allow tuning which data is captured and how much. This lets you balance issues such as privacy concerns and data storage requirements. Entropy analysis of captured data is possible, which is useful in determining if the data is compressed or encrypted.

Various protocols are understood by YAF; X.509 is being worked on. Understanding who is creating encrypted tunnels on the network can help identify malware.

Christopher talked about common YAF deployments and the type of environment it has been used in. Generally, the capture device running YAF is attached via an optical splitter, providing an air gap that limits vulnerability to attack. Carefully crafted packets and payloads may still be a source of vulnerability. The authors' typical installation involves high data rates (monitoring multiple 10Gb links), which requires high-performance databases.

A toolkit is provided to build your own mediators. The goal is to make it easy to capture and store the particular data that interests you. Future work includes adding protocols and improving data storage abilities, thus providing the ability to use back ends such as MySQL, which should be adequate in environments with smaller data capture needs.

YAF is available at <http://tools.netsa.cert.org/yaf/index.html>. It has been in development for four years, has been deployed to several sites, and has proven to be stable. The authors are interested in hearing how YAF has been used and what improvements are desired.

Comments and questions about YAF and other tools can also be directed to netsa-help@cert.org.

Nfsight: NetFlow-based Network Awareness Tool

Robin Berthier, University of Illinois at Urbana-Champaign; Michel Cukier, University of Maryland, College Park; Matti Hiltunen, Dave Kormann, Gregg Vesonder, and Dan Sheleheda, AT&T Labs—Research

Robin Berthier started his talk by thanking Christopher M. Inacio for explaining NetFlow so well. The authors felt there were no tools available that fit the gap between tools providing a detailed and a high-level view of network flows. Nfsight was designed to fill this gap. It does so by aggregating flows by host and port number and creating a high-level view along with the ability to drill down to see details.

The challenge the authors faced was to identify bi-directional network flows without requiring IPFIX, which was not yet in mainstream use. Heuristics were developed and analyzed as to effectiveness in identifying flow direction. The back end is a set of Perl scripts that processes Nfsen/Nfdump data and stores the result in both flat files and a database. The front end uses PHP and JQuery to visualize the stored data. Automated alerts can be sent by the front end, and you can tag hosts with notes for team members to document network activity.

The back end is composed of two scripts. The first identifies the bi-directional flows and client/server data and stores these in flat files and a database. A Bayesian inference is used to combine several heuristics to identify flow direction. Robin explained some of the heuristics used, such as time-stamps, port numbers, and fan in/out relationships.

By using Bayesian inference to combine outputs, Nfsight is able to improve the accuracy of identifying directionality. Heuristics have differing levels of accuracy, some varying depending on the NetFlow behavior, while none were able to reliably determine direction all of the time. By combining the output of several heuristics, they were able to determine direction most of the time.

The second back-end script provides a framework for writing signatures that identify malicious activity in the bi-directional flow data. The current set of signatures include three categories of flows: malformed, one to many (possible scanning), and many to one (possible denial of service attacks). The IDS signatures are evaluated by sending the top alerts to the signature's author along with links to visualize the flows and then mark the alert as a true positive, false positive, or inconclusive. The results of this voting are then used to evaluate the signature as good or bad at identifying actual attacks. Most of the false positives came from heavily used services, so a whitelist capability is being developed to limit false positives stemming from these services.

The front end is used to visualize NetFlow data stored in the files and database. Flow data can be filtered based on criteria such as IP range, port number, time, and type of activity. The filtered data is then displayed showing endpoints, flow metrics, such as number of packets, and a heat map of activity. The color intensity indicates number of flows, and the particular color shows the type of flow (e.g., client or server). Red is used to indicate that the flow could not be matched to identify a bi-directional flow. This often indicates network problems or scans. Clicking on the heat map drills down to a detailed view, and this is where hosts can be tagged with a note that is viewable by others.

A demo of Nfsight was given that showed the high-level views and drilling down to the different detail views. Several use cases were also presented: the first showed an example of a power outage, clearly indicated by gaps in the heat map. Which hosts were, or were not, affected by the outage was easy to spot. By selecting port number, the tool can visualize external scanning of the network and which internal hosts are answering those scans (and thus may need patching).

Visualizing the flows can also be used to identify distributed and synchronized activity. An example was shown of a simultaneous attack on 20 vulnerable SSH servers. Future work includes improving the IDS signatures and creating additional heuristics to identify the type of service. This could be used to find Web servers operating on ports other than 80.

Nfsight will soon be available as open source at: <http://nfsight.research.att.com>.

If you have questions or wish to be notified when the tool is released, send email to rgb@illinois.edu.

Invited Talks I

Visualizations for Performance Analysis (and More)

Brendan Gregg, Joyent

Summarized by Mark Burgess (mark@cfengine.com)

Brendan Gregg presented an invited talk based upon a recent article in *Journal of the ACM*. Gregg spoke first about performance measurement in general. He emphasized that measuring I/O performance (or “IOPS”) could be a misleading pursuit, since it is difficult to know exactly which layer of the software stack is responsible for the results. Better to study latency as a compound effect, since it includes all layers. If performance has a ceiling, for instance, we have to find the weakest link in the stack. He also emphasized the importance of workload analysis in a network, not just on a single host, and recommended a split between measuring load and architecture as soon as possible in a performance analysis in order to understand the effect of communications in the system.

Gregg promoted DTrace as a toolkit, claiming that it is “game changing” for measurement and showing how some of its data could be presented using a form of granular time-series called a heat map. A heat map is a combination of line graph with histogram over a bucket of time. It shows a rolling distribution, something like using error bars for repeated measurements, but using colors and two-dimensional space to represent the data. The heat map shows not only a single sample line but an accumulated distribution of values in a measurement interval that can indicate skew and exceptional behavior. A color-shaded matrix of pixels was used to show latency versus time. By using a false color palette, it is possible to see outliers in the histogram more clearly—the palette can be used to emphasize details, but the colors can become confusing

Gregg proposed that visualization allows us to use the human brain’s ability to pattern-match to maximum effect. Writing software to see the same patterns is very hard. He showed a number of examples based on complex systems of disk reads, showing interesting and even beautiful patterns, although he had no explanation for the patterns that resulted. In questions it was suggested that the patterns might be explained by a model of periodic updating. Gregg ended by suggesting that visualization could be used to monitor the cloud for performance and even for system administration—e.g., in measurement of user quotas.

Invited Talks II

Rethinking Passwords

William Cheswick, AT&T Labs—Research

Summarized by Rik Farrow (rik@usenix.org)

Cheswick pointed out that he was 98th on a list of the 100 most influential people in IT. He then moved on to address a problem that really needs fixing: passwords. We need to have passwords that work for both grandma and the technical audience.

Cheswick went through many examples of password rules (calling these “eye of newt” rules), exhibiting conflicting rules and widely varying lengths. He claimed that he at least shares some responsibility for the rules, as he and Steve Bellovin had suggested having rules in their 1994 firewalls book.

Cheswick then used a short animated clip from a Bugs Bunny cartoon, where a rather stupid Middle-Eastern-appearing character keeps guessing until he comes up with the magic phrase to open a door: Open Sesame. Stopping brute force password guessing attacks was the focus of the NSA’s Green Book, back in 1985. What gets ignored are attacks where passwords are stolen: keystroke logging, phishing, and theft of databases containing passwords. If brute force attacks can be limited by inserting delays, Cheswick wondered why we continue to have “eye of newt” rules?

Cheswick suggested that we have only one rule as an engineering goal: the don’t-be-a-moron rule. This rule prevents the use of your own name, permutations of your name, and dictionary words. At this point he mentioned that his grandmother had written a disk device driver for the Univac 1, so his standard for “grandmas” seems a bit skewed. Cheswick also mentioned the Schecter and Herley paper from HotSec ’10, where they suggest allowing any password at all, but only allowing about 100 people to use a particular password. This way, only 100 people could have the same password, although the “don’t-be-a-moron rule” still applies.

Cheswick had a list of suggestions to help prevent brute-forcing and to make these preventive mechanisms less painful for users: use less painful locking—the same password attempt twice counts as one attempt; make the password hint about the primary password; allow a trusted party to vouch for the user (a significant other); use exponential backoff for delays instead of locking accounts; remind the user of password rules (eye of newt), as this might jog her memory.

He went on to suggest better solutions, such as getting away from static passwords entirely. He likes hardware tokens and challenge-response, and had looked at RSA softkeys

on smartphones, saying that the software does not include enough information to reveal the PIN.

Cheswick then provided a lesson in password entropy. He used the example of the 1024 (2^{10}) most popular English words, and that using two of these words as your password provides 20 bits of entropy. I wondered about this, as this means the brute force space is only 1024 squared, but Cheswick is right (no surprise there). He went on to explain that Facebook's rules require at least 20 bits of entropy, banks in the 30s, and government and .edu rules in the 40s.

Cheswick has a history of experiments with passwords, and he talked about how baseball signals worked, and how this could work using a challenge-response scheme you could do in your head (and certainly so could his grandma). He then described other potential schemes, such as passpoints, passfaces, blurred images, passmaps, passgraphs, even using a point in a Mandelbrot set.

He concluded with advice for users: use three levels of passwords; write down your passwords but vary them according to memorized rules; write down the "eye of newt" rules. He also suggested using PAM tally, a module found in most Linux distros that does account locking. He likes near public authentication services such as OpenID and OpenAuth.

I started the Q&A by pointing out that devices can be left behind: for example, showing up in Australia on Monday and not having your hard token when it is Sunday morning back in the US. Paul Krizak of AMD said that many people had switched to the site key model, but he found it interesting. Cheswick replied that this was a nice defense to phishing, that grandma could do it, and it was actually a good idea. Jay Faulkner of Rackspace pointed out that anyone playing Final Fantasy gets a physical token, and Cheswick said, "Fine." Marc Staveley said he checked his personal password vault and found he had 200 level three passwords secured with a level four password, and Cheswick suggested that perhaps he spends too much time online. Staveley then asked how we get beyond this, to which Cheswick responded that we need to go to OpenID or Google, or various values thereof.

Practice and Experience Reports

Summarized by Rik Farrow (rik@usenix.org)

Implementing IPv6 at ARIN

Matt Ryanczak, ARIN

Matt began by saying that getting IPv6 to work really won't be that hard. IPv6 is about 20 years old (it was called IPng in RFC 1475), and back then, some people expected to have replaced IPv4 before the end of the '90s. He then went on to

explain his own experience setting up IPv6 connectivity for ARIN.

In 2003, he got a T1 from Sprint to do IPv6. Sprint was testing IPv6 to their own site, where it then was tunneled. He needed special device driver support in Linux since ip6tables worked poorly; he instead used pf under OpenBSD, which worked really well. He had a totally segregated network, with a dual stacks system set up to reach the IPv4 side of the organization.

In 2004, he got a second link from Worldcom, also not commercial, used a Cisco 2800 and continued using the OpenBSD firewall. ARIN joined an exchange in 2006, Equi6IX, peered with lots of people, and could get away from T1s. They went to a 100 Mb/s link and no longer had IPv6 to IPv4 tunnels getting in the way, which was much closer to the class of service found in v4.

By 2008, he found the first networks designed with IPv6 in mind. He had wanted to find colos, with dual stacked networks and Foundry load balancers (v6 support in beta). The amount of IPv6 traffic was low enough that a single DNS server was more than sufficient—and still is today. ARIN now had 1000 Mb/s to NTT TiNet, Dulles, and San Jose. In 2010, they have two more networks, in Toronto and St Martin, are still using beta firmware, DNS only, and plan on anycast DNS eventually. Matt said the IPv6 only accounts for a small amount of ARIN's network traffic. He broke this down by categories: .12% Whois, .55% DNS, .65% WWW.

Matt went on to cover a lot of the topics he wrote about in his October 2010 *login*: article: all transits are not equal, check for tunnels, routing is not as reliable (although it has gotten better). Sometimes Europe would "disappear for days," and parts of the Internet still disappear from IPv6 routing. Matt emphasized that you must understand ICMPv6, because of fragmentation issues. In v4, routers can fragment packets, but not in v6. The sender must receive ICMPv6 for path MTU discovery, and the sender must fragment. Dual stacks are a good thing, okay for security. It does make policy more complicated, and you need to maintain parity between firewall policies, for example. DHCPv6 is not well supported. Windows XP barely supported v6. Linux is okay, *BSD better, Solaris and Windows 7 work out of the box. Windows XP cannot do v6 DNS lookups.

There is no ARP on IPv6; it uses multicast instead. This is also great for scanning networks and DoS attacks and can be routed, providing a whole new world for hackers to explore. Read RFC 4942 for v6 transition, how to properly filter v6 to avoid discovery issues. Proxies are good for transition: Apache, squid, and 6tunnel are valuable. Reverse DNS is

painful; macros in BIND do not work in v6 for generating statements, may be broken, and are difficult to do by hand.

Carolyn Rowland asked about working with vendors. Matt said he has received great support working with Arbor Networks in the security area. Carolyn wondered about problems we saw with v4 stacks, like the ping of death (Windows 95). Matt said that there certainly could be more bad code around. Someone asked which versions of DHCP he was using. Matt replied that they were using the “out of the box” DHCP client software in various OSes, and only Windows 7 and Solaris worked well so far.

Internet on the Edge

Andrew Mundy, National Institute of Standards and Technology (NIST)

► *Awarded Best Practice and Experience Report!*

Andrew Mundy is a Windows network admin at NIST headquarters in Gaithersburg, Maryland. He was asked to provide network services for an experiment in autonomous robots—about a half mile away from any building on campus. As this request came from an outside contractor, it had to be a visitor connection. NIST has a visitor wireless infrastructure in place, but now they needed to reach the middle of a field, to locations that actually changed during the project.

He discovered that they can do this using a different type of wireless access point with an external antenna. But they must have a line of sight to the location, as well as access to a fiber link, and the one guy who could tap into the fiber works only on Thursdays.

Mundy’s team picked the rooftop of the administration headquarters. This building even had unused steel structures on the roof they could use for mounting the antenna. They then tried a Cisco 1240 AG with a directional antenna, but it can only provide 80 Kb/s. They tried a couple of Aironets and settled on the AIR ANT3338 with a parabolic antenna with a 23 db gain. They ordered two, one for the roof and one to be mounted on the support trailer, where they will provide local wireless secured with WPA2 and wired Ethernet within the trailer. The trailer used a Honda generator, which provides very clean power for computers.

On roof install day, they prepared by tying on their tools so they couldn’t drop them off the roof. They also prepared to ground the steel support structure by attaching heavy RG-6 copper cable to a grounding block. As they were about to enter the elevator, some guys exited carrying some steel pipes, parts of the roof structure they planned on using. They were dumbstruck, as getting the paperwork to replace the structure would take six months. Walking back to their building, they found a wooden packing crate. They “borrowed” it,

modified it, added a pipe for clamping on the antenna, and bought sand and sandbags to hold it in place.

With the antenna in place and working, the contractors decided on a new site about three miles away. They used Google Maps to compute a new azimuth (direction) and lost line of sight (some trees), but things still worked, as they had enough gain. They did have to make sure that the wireless power output, plus the gain of the antenna, remained within legal limits. (Editor’s note: For more on power, see Rudi van Drunen’s February 2010 *login*: article about “Peculiarities of Radio Devices.”)

Lessons learned include using a lot more sandbags, rather than worrying about a gust of wind ripping the box off the roof and dropping it on the Director’s windshield. They wound up with 500 lbs of sand. Next, remember that temporary solutions aren’t always temporary. Finally, be flexible (*Semper Gumby*). He never would have thought a wooden packing crate and Google Earth would have provided an enterprise network solution.

Jay Faulkner of Rackspace asked if the signal was strong enough to work during rain and Mundy said that they didn’t test it during rain. Someone from Cisco asked how much they had to reduce the wireless signal power to prevent exceeding the legal limit. Mundy said 30%, so as not to exceed the 30 db power limit. Carolyn Rowland asked what they would have done differently, to which Mundy replied they could have used bridge mode, which would have gone miles and miles.

Managing Vendor Relations: A Case Study of Two HPC Network Issues

Loren Jan Wilson, Argonne National Laboratory

Wilson began by asking if there were any Myricom users or HPC administrators in the audience. No hands went up. He then went on to describe *Intrepid*, which was the number three supercomputer in its time. It is built of IBM Blue Gene/P nodes, which are PowerPC CPUs with local RAM but no storage. There are 1024 nodes per rack and they had 40 racks. Each node requires booting when starting another program. Access to remote storage is key to keeping the supercomputer working, both for booting and for storing results.

The Argonne Leadership Computing Facility (ALCF) had perhaps 5 PBs of useful storage, some over NFS, but most via GPFS. The plan was to connect the nodes in a full bisection mesh. Every node link is 10 Gb, and they needed to connect them via 10 Myricom switches, which provided a 100 Gb uplink. Wilson showed a diagram of the *Intrepid* setup, then a picture showing thousands of cables connected to the nodes and switches.

Myricom, <http://www.myri.com>, has totally stupid switches, that is, the only management interface is via HTTP. Myrinet itself is a source-routed protocol, which means that every host keeps a map of the network which is used to route each packet. But the Myrinet switches kept breaking, with about 6% of ports affected by a random port death issue. At first Wilson just switched to spare ports, but then he started disassembling switches and noticed that the ports that died were attached to a particular brand of transceiver. They also had 1000 quad fiber connects fail, and these interconnects don't just fail: they also corrupted packets.

ALCF lost 375 days of compute time due to these network issues. Wilson blames a lot of that on his own failure to create good relationships with the vendors involved. He suggested not starting with "OMFG everything is broken," as it will take years to recover your relationship with the vendor. They got a good deal on the Myricom gear but should have paid more for people to help deal with the gear. As it was, the reseller was pretty useless. Also, Myricom got paid before they shipped a single piece of gear.

After a while, they had weekly phone meetings, and once they started to do that, things worked better. Wilson wrote a switch event collector in Perl, which helped. When disassembling switches, he noticed that it was Zarlink that made the bad transceivers, and not only did the Avago transceivers work well, their support was good. Zarlink never even responded to him.

Carolyn Rowland asked if ALCF learned from these lessons. Wilson said that he wrote this paper when he was working there. Since then, he and a lot of others had left, and ALCF had probably not learned their lesson. Hugh Brown of UBC asked if doing acceptance testing should have been part of the lesson. Wilson replied that you should do acceptance testing, and you should not skimp. He suggested that you come to agreement on how things are supposed to work.

Invited Talks I

System Administrators in the Wild: An Outsider's View of Your World and Work

Eben M. Haber, IBM Research—Almaden

Summarized by Tim Nelson (tn@cs.wpi.edu)

Eben Haber began his talk by reminding us that society depends implicitly on IT infrastructure, and that without system administrators it would not be able to sustain itself. Unfortunately, system administration costs are increasing (as a percentage of total cost of ownership) and so there have been attempts to create "autonomic" systems that would be able to self-configure as well as detect and repair issues on

their own. This effort has spurred an interest in how exactly sysadmins do their job.

Haber showed a video clip of two sysadmins making changes to a database table. After a week of preparation, a small mistake nearly resulted in disaster. The clip served as an example of how high-risk a sysadmin's job can be, as well as showing tools and practices sysadmins create themselves to handle the risk. There was also a high degree of collaboration, in spite of how outsiders may view the job.

Haber is writing a book detailing an ethnographic study of system administrators at work. After summarizing the book, he showed a longer series of clips taken from the first chapter. We see a junior sysadmin ("George") attempting to solve a configuration problem. We see George struggle with the issue, call technical support, and also work with his colleague ("Thad"). George finally sees the cause of the problem but misinterprets what he sees, leading to a fruitless search until Thad discovers the problem on his own. George resists the fix, and Thad must debug George's misconception. Finally, George realizes his mistake and fixes the error. Between clips, Haber pointed out communication issues, such as the use of instant messaging to discuss Thad's solution when a phone conversation or face-to-face talk would be better, and observes that we need better tools for accurately sharing system state during collaboration.

In closing, Haber summed up what they had learned about the practice of system administration: that the environment is large-scale, complex, and involves significant risk. The ways sysadmins cope with their environment were interesting and included collaboration, tool-building, standardization, automation, specialization, and improvisation. He then considered the future of system administration. He drew comparisons to the (now obsolete) flight engineer position aboard airplanes, noting that as automation outpaced increases in complexity, the pilot and co-pilot were no longer dependent on a dedicated flight engineer. So far, the complexity of IT systems has kept pace with automation technology, which is why the job is not getting any easier.

Jonathan Anderson commented on the trust relationships that develop between sysadmins and wondered about keeping the balance between personal development and just relying on someone you trust. Another audience member then observed that George even began with a "mistrust" relationship with tech support.

Someone commented that it can take weeks or months to decide on a course of action, but sysadmins get a far shorter window to implement changes. Phil Farrell noted that a large organization will have even worse communication bottlenecks than a smaller one. Alva Couch from Tufts commented

that people under pressure feel entrenched and are more resistant to “debugging.” Sysadmins are under pressure to balance open-mindedness with snap judgments.

Several audience members brought up the issue of executive meddling. Haber replied that they had not seen major instances of executive meddling, but agreed that social requirements can exert as much pressure on a sysadmin as technical requirements.

Someone pointed out an ethnographic study of technology workers in Silicon Valley (done in the late '90s at San Jose State). Someone from Google wondered if there were videos of senior sysadmins as well; Haber replied yes, but that their videos were not as compelling for a large audience.

Jason Olson asked whether they had looked at high-performing sysadmins and tried to find indicators and counter-indicators of high performance, such as whiteboards. Haber answered that the sample size was too small, but the whiteboard would be a good example.

Someone asked whether the video recording may have influenced the experiment and whether the subjects might have been nervous because of the recording. Haber replied that there was some initial nervousness but they seemed to ignore the recording process eventually.

Invited Talks II

Enterprise-scale Employee Monitoring

Mario Obejas, Raytheon

No report is available for this talk.

Refereed Papers

Summarized by Julie Baumler (julie@baumler.com)

Using Syslog Message Sequences for Predicting Disk Failures

R. Wesley Featherstun and Errin W. Fulp, Wake Forest University

Featherstun started out by talking about how as systems become larger and, particularly, become collections of more parts (processors, disks, etc.), failures of some sort become more common. If we can't avoid failures, can we better manage them? The key to management is accurate event prediction. Featherstun and Fulp looked specifically at predicting disk failures. Since pretty much every device has a system log, they decided to use syslog. Syslog messages represent a change in state. They wanted to use that to predict future events.

They originally used a Support Vector Machine (SVM) with criticality numbers from syslog to determine priority. Later they found that criticality numbers seem to be falling into disuse and, after some experimentation, switched to a vocabulary of about 24 words from the messages themselves. They used a sliding window so that older messages would be ignored.

For testing they worked with almost two years' worth of data from a 1024-node Linux cluster. Using this method, they were able to achieve about 80% predictability of disk failures. Featherstun discussed the various refinements on window size, lead time, and vocabulary to achieve these rates and how different changes affected the results.

Several questions were asked regarding current and future plans for this technology, and Featherstun suggested contacting Dr. Fulp (fulp@wfu.edu) as Featherstun is no longer working on the project.

Log Analysis and Event Correlation Using Variable Temporal Event Correlator (VTEC)

Paul Krizak, Advanced Micro Devices, Inc.

► *Awarded Best Paper!*

The original goal of Paul Krizak's project was to create a log analysis solution that was scalable to large quantities of logs, could take advantage of multiple processors, worked in real time, and would allow other developers to write rules. They had previously been using Swatch, but it did not scale or have good event correlation. They looked at Splunk version 1, which could not scale to index all their data, and at SEC. They felt that SEC's rules were too difficult to use to meet their goals.

The system is written in Perl. It keeps track of how often events occur and has timeouts to avoid multiple notifications. Syslog-ng forms a core part of the system, which consists of multiple rule engines, a variable server, and an action server. The variable server keeps common data so that the rule engines do not have to manage state. The rule engines process the filtered logs coming from syslog-ng and notify the servers as necessary. The action server both produces alerts and queues jobs to solve detected problems. Jobs and alerts can be run immediately or held for a later time, such as business hours for alerts or maintenance windows for repairs. Krizak also discussed some of the lessons learned in producing the system, such as using a language that everyone was familiar with for the rules engine. The system is currently in use and mostly in a “fire and forget” state.

Someone asked if the system was publicly available. Krizak replied that it currently belongs to AMD and much of it is

environment-specific. However, he is willing to work with AMD to make it open source if there is sufficient interest and he does not have to become the project manager.

Chukwa: A System for Reliable Large-Scale Log Collection

Ariel Rabkin and Randy Katz, University of California, Berkeley

In Hindu mythology, Chukwa is the turtle that holds up the elephant that holds up the earth. Since Hadoop's symbol is an elephant and originally a key goal was to monitor Hadoop, it seemed appropriate. Chukwa is optimized for a certain large to mid-sized monitoring need and allows for two different ways of gathering data: either what you can get as quickly as possible or gathering 100% of the data, which could mean waiting for data that is on down servers. It uses Hadoop and MapReduce for storage and processing.

Chukwa was originally a Hadoop project, is now in Apache Accelerator, and will be moving again to be a regular Apache project. The easiest way to find it is to do a search for Chukwa.

Alva Couch asked if this project was just for applications in the cloud. Rabkin replied that most users are not cloud users—in fact, the killer application seems to be processing blogs—but that Chukwa was designed to be cloud-friendly.

Invited Talks I

Flying Instruments-Only: Navigating Legal and Security Issues from the Cloud

Richard Goldberg, Attorney at Law, Washington, DC

No report is available for this talk.

Invited Talks II

The Path to Senior Sysadmin

Adam Moskowitz

Summarized by Theresa Arzadon-Labajo (tarzadon@ias.edu)

Adam Moskowitz laid out the steps he felt were important for one to become a senior system administrator. He pointed out that the talk was going to be career advice to achieve professional and personal growth and would help advance one's career. The talk was aimed at mid-level sysadmins, but could be used as long-range goals for junior sysadmins.

He broke down the steps into three categories: hard skills, squishy skills, and soft skills. Hard skills are the technical ones, the easiest to achieve for system administrators, and are mainly for the generalist system administrator, the ones who do everything. Squishy and soft skills are the difficult

ones and are important for everyone. The USENIX Short Topics *Job Descriptions for System Administrators* should be used as a reference.

An important hard skill is to know multiple platforms, because it's a big win for employers. Backups, email, and networking are the minimum things to know. Familiarity with all the commands on the systems is suggested. It is also necessary to understand the boot process. They should specialize in at least one implementation of backups, RAID, volume management, and authentication mechanisms. A sysadmin needs to have programming knowledge, at least for automation purposes. Shell, sed, and awk should be a starting point. Beyond that, one should have knowledge of at least one robust programming language. Adam recommended Perl, since it's what most system administrators use, but one should learn what is most common in a given environment. Sysadmins should be able to read and understand C. As a bonus skill, assembler can be useful, because the deepest parts of the kernel are written in it. As far as software engineering skills go, familiarity with version control is needed, either with a specialized tool or by hand, so you have a way of rolling back. When writing utility scripts, make sure not to hard code anything. Proficiency in a system configuration tool such as Bcfg2, Puppet, Cfengine, or Chef is suggested, as is the experience of having set one up from scratch in a real environment. Basic knowledge of networking protocols such as TCP/IP, UDP, switches, and routers is important. An in-depth knowledge of application protocols such as HTTP, FTP, imap and SSH is recommended, so that simple debugging can be performed. A sysadmin should have a reasonable understanding of firewalls and load balancers and be able to use a protocol analyzer. A triple bonus skill to have is knowing the kernel. It can be helpful when doing performance tuning. Whether they like it or not, someone who wants to be a senior sysadmin needs to know Windows and be familiar with the basic configuration and common Office applications such as Outlook or Lotus.

Squishy skills are technical skills that don't have to do with a specific technology. Some skills face out and deal with procedure and other face in and deal with career growth. One facing-out skill is being able to do analysis, planning, and evaluation. A senior sysadmin has the ability to look at the "big picture" when dealing with a project. They know how all the pieces interact, for example, knowing the requirements for networking, servers, and power when planning a data center. Being able to know how long a project will take and how to schedule it accordingly is significant. They should know how to perform roll-outs, upgrades, and roll-backs if things don't work out. Another facing-out skill is understanding how a process works. There should be rules on how things get

done, whether it be a formal change management procedure or just an email that is sent out 24 hours in advance. Also, knowing how much process is appropriate is vital, because process can get in the way of getting the job done. But, if done well, a rule-based process helps get the job completed and prevents mistakes from happening. Senior sysadmins should know how to deal with business requirements. They should know the prevailing standards for what they are dealing with (e.g., POSIX, IEEE Std 1003.x, Spec 1170). They must possess knowledge of the regulations that they have to work within (e.g., SOX, HIPAA, FERPA, PCI). Then they can work with experts to correctly apply those regulations to their business. Senior administrators are expected to interface with auditors and consultants, so they should be able to talk about the business. Service level agreements (SLAs) should be appropriate for what the business is. Things should only get done if there is a requirement to do it and not just because it's the cool thing to do. Any future growth should be already written in the business plan. Sysadmins should be able to go to their boss and explain why something is needed and be able to tie it into the business requirements. Budgeting is another skill that is needed. Knowing how to build a budget is not required, because that's what your boss does, but knowing what data goes into it is. A sysadmin should also be able to obtain reasonable quotes from vendors.

A facing-out squishy skill, that is, one that pertains to career growth, is that sysadmins need to learn where to find help, since their manager may not be the one who could help. Places to find help are at conferences like LISA, LOPSA and SAGE mailing lists, local sysadmin groups, and Facebook or Twitter. The personal contacts that are made are very valuable, so paying out-of-pocket or taking vacation to go to a conference is worth it. Sysadmins should have their own library and not rely on their employers to buy the books they need. If they change jobs, they should be able to drop the books on their desk and be ready to work and not have to wait several weeks for the books to arrive. Knowing when to ask for help is a very hard skill for sysadmins to learn. But there may come a point in their career when there won't be many people who can help them out. Pair programming can be a very good skill for senior sysadmins, so that they can explain what they are doing to someone else and make sure that they are not going to do anything bad to the system.

Soft skills are the hardest skills for sysadmins to learn. Understanding that their job is about the people and the business, not the technology, is key. If they got into system administration because they didn't want to deal with people, that may be okay if they are a junior sysadmin. But senior sysadmins deal with a lot of people all the time. An important soft skill is having a friendly and helpful attitude. If people

don't want to come to you with their problems, then you have failed at your job. On the other hand, if they like you and know you will fix things, then you will have happy customers. A senior sysadmin needs to be comfortable talking to management and explaining things to them with appropriate detail and reasoning. Respecting other people in the company is an extremely important skill. Sysadmins need to understand that it's not the worker's job to know about computers. Another critical skill is being able to get in front of small groups and make presentations. Senior sysadmins will be required to explain new products or procedures to people and meet with managers. Mentors and managers can help sysadmins work on their soft skills. They can point out what needs to be worked on and can track your progress. LOPSA also has a mentorship program that might be worth looking into.

John from UC Berkeley commented that conflict resolution is useful if the reason you are confrontational is because your manager rewards it unknowingly and you get more results than when you are nice and polite, which can point to larger problems in the organization. Jay from Yahoo! commented that sysadmins are held back by their inability to acknowledge they don't know something and are unwilling to use the resources available to them. He pointed out that having a network is really important. Jason from Google asked about strategies for team building. Adam responded that he didn't have much experience with that, but that Tom Limoncelli would be a good person to ask. Robyn Landers from University of Waterloo asked whether Adam felt there was any value to the Myers-Briggs personality test or other categorization exercises. Adam wasn't totally convinced that it is beneficial in a group setting, but it is more of an introspection thing. He felt that it was worth figuring out what the personality differences are, how they affect things, and how they affect you.

Refereed Papers

Summarized by Misha Zynovyev (zynovyev@stud.uni-heidelberg.de)

How to Tame Your VMs: an Automated Control System for Virtualized Services

Akkarit Sangpetch, Andrew Turner, and Hyong Kim, Carnegie Mellon University

Akkarit Sangpetch talked about automated resource management for virtual machines. He emphasized how virtualization simplifies the lives of system administrators. Consolidation of resources was named as one of the key benefits of virtualization. The problem that the paper and the talk were addressing is how current techniques for sharing resources among virtual machines on a single host fail to consider the application-level response time experienced by

users. Akkarit and his co-authors suggested a way to dynamically allocate resources for virtual machines in real time to meet service-level objectives.

The speaker showed an example of how their system works when a user accesses a blogging application run within a Web server and a database virtual machine. He presented a control system of four components implementing a CPU-sharing policy based on analysis of network packets intended for a controlled virtual machine. Akkarit concluded by explaining the graphed results the model had produced.

Paul Krizak from AMD asked on which virtualization platform the model was tested. Akkarit answered that it was KVM, but noted that there is no reason why it shouldn't work with VMware ESX or Xen. Kyung Ryu from IBM research was curious whether I/O operations and I/O contention could be taken into account with the presented approach. Theodore Rodriguez-Bell from Wells Fargo added that the IBM mainframe community was studying the same topic and asked if there was agreement on results.

Empirical Virtual Machine Models for Performance Guarantees

Andrew Turner, Akkarit Sangpetch, and Hyong S. Kim, Carnegie Mellon University

Andrew Turner explained how Akkarit Sangpetch and he started to work on the topic. The difference in their approaches lies in the starting points of their research tracks. Andrew Turner used the performance experienced by the end user as his starting point, while his colleague was looking at network packets in search of dependencies at a low level.

The presented model is multidimensional and covers disk and network performance as well as CPU. The model lets one infer how these resource utilization characteristics are affecting application performance over time. In the end, a system administrator should be able to distil from the model how resources should be allocated between particular virtual machines (VMs) in order to achieve a specified application response time with a specified probability. A control system that dynamically alters virtual machine resource allocations to meet the specified targets was described. Lastly, the speaker guided the audience through the results acquired with TPC-W benchmark and a three-tiered application for dynamic and static resource allocations.

Paul Krizak from AMD asked if the code would be freely available to the public. Andrew said no. Session chair Matthew Sacks asked how workload balancing was done, whether new VM instances were started on demand or workload was shifted to idle VMs. Chuck Yoo from Korea Univer-

sity asked whether any VM scheduler had been changed and was interested in more details on how modeling was done.

RC2—A Living Lab for Cloud Computing

Kyung Dong Ryu, Xiaolan Zhang, Glenn Ammons, Vasanth Bala, Stefan Berger, Dilma M Da Silva, Jim Doran, Frank Franco, Alexei Karve, Herb Lee, James A Lindeman, Ajay Mohindra, Bob Oesterlin, Giovanni Pacifici, Dimitrios Pendarakis, Darrell Reimer, and Mariusz Sabath, IBM T.J. Watson Research

Kyung Dong Ryu gave an overview of the IaaS cloud project at IBM Research, named RC2. He started by introducing IBM's research division and explained that IBM labs are scattered across the globe. All individual labs buy and install their own computing resources, which are often underutilized but which now can be integrated into a single playground for all IBM employees.

One of the key differences from Amazon's EC2 and Rack-space clouds is that RC2 needs to run on AIX and mainframes too. Dr. Ryu showed the cloud's architecture and briefly stopped on each of the components. He himself was mainly involved with development of Cloud Dispatcher, which handles user requests and prevents the overloading of other components. Among components described in detail were Instance Manager and Image Manager. Security Manager provides a trusted virtual domain with a way to control what traffic can come from the Internet and what communication is allowed between virtual domains. Another important issue raised was the pricing model implemented to make users release resources. It was shown how introduction of a charging policy was affecting cloud utilization.

Answering the question about availability of RC2 outside of IBM, Dr. Ryu expressed his doubt about any eventual open sourcing but pointed out that some of the components may have already been released, such as the Mirage image library. On the other hand, RC2 may be offered as an IaaS cloud to selected customers.

Invited Talks I

Panel: Legal and Privacy Issues in Cloud Computing

Richard Goldberg, Attorney at Law, Washington, DC; Bill Mooz, VMware

Summarized by Robyn Landers (rblanders@uwaterloo.ca)

This panel session was a follow-up to Richard Goldberg's invited talk earlier in the day, giving an opportunity for elaborating on the discussion along with Q&A. Session host Mario Obejas asked most of the questions to keep things going.

Goldberg began by reminding us why cloud computing is “dangerous.” You give up control. The government could demand a copy of your data. You or the cloud provider could be subjected to subpoena. The implication is that the cloud provider may not handle your data the way you would have, so there may be more there to expose. Think of it as a legal attack vector.

Bill Mooz countered with the observation that clouds could be less dangerous than ordinary outsourcing depending on what SLA you can negotiate, or even your own datacenter, depending on your own operational standards. Activities such as payroll and tax returns were among the first to go to the cloud, yet you’d think those are things people would want to keep most private, and there haven’t been disaster stories. Goldberg allowed that it’s a question of different dangers, and one needs to plan for them.

The discussion touched on service level agreements, who they favor, the extent to which you can negotiate terms and penalties, and the importance of being able to get out. Again, comparisons were drawn between cloud and regular outsourcing based on whether customers are isolated on dedicated equipment. Software licensing cost and compliance are also issues here, as traditional per-CPU licensing may not apply to a cloud scenario.

After some discussion of the merits of a mixed mode in which you keep your most precious data on-site but outsource less critical data, the conversation came back to legal issues such as HIPAA compliance, questions of jurisdiction when outsourcing to companies in other states or countries, and ownership of and access to data in the cloud.

Rik Farrow asked about the ability to ensure data destruction when using cloud services, since the mixing of data in the cloud may expose more to subpoena, for example. Mooz speculated that you probably can’t comply with a DoD contract requiring disk destruction if you’re using the cloud. Goldberg agreed; even de-duping on-site mixes data (implying the potential exposure of other data in response to court order is as bad or worse in the cloud).

Johan Hofvander asked whether current law addresses the difference between real property (back in the days of livestock and horse carts) and intellectual property (in our modern times of digital music, movies, and personal information), and the duty of care. Mooz said it’s likely spelled out in the contract, and you take it or leave it.

Session host Obejas repeated a scenario from an earlier session: imagine a multi-tenant situation in which one tenant does something bad and law enforcement agencies want to shut it down, affecting innocent tenants. Has this ever hap-

pened? Goldberg said that although he originally made up that scenario, it subsequently has indeed happened. A fraud case in Texas led to the FBI shutting down a service provider and taking everything. Mooz speculated that although cloud service wouldn’t have the same physical separation that regular outsourcing might, there could be virtual separation. Goldberg pointed out that the FBI might not understand that partitioning and might still take everything.

Farrow wondered whether that’s a good enough reason to choose the biggest cloud provider you can find. Goldberg agreed that although the FBI would shut down a small provider, they probably wouldn’t shut down Amazon.

The session concluded with the suggestion that one must analyze the risks, decide what’s important, and take reasonable protective steps.

Afterwards, Lawrence Folland and Robyn Landers, University of Waterloo, brought up the scenario of a Canadian university outsourcing email to an American cloud service, thus exposing itself to the Patriot Act. Imagine there were Iraqi or Iranian students and faculty members whose data the US government might be interested in monitoring. The speakers agreed that this is an interesting predicament. And viewed the other way around, could the American cloud provider get in trouble if, say, a Cuban national was included?

Invited Talks II

Centralized Logging in a Decentralized World

Tim Hartmann and Jim Donn, Harvard University

Summarized by Rik Farrow (rik@usenix.org)

Hartmann and Donn took turns explaining how they went from a somewhat functional logging infrastructure to one, Splunk, that collects a lot more logs and is easier to use. Hartmann explained that Harvard, like most universities, is composed of IT fiefdoms. His group was using syslog-NG, but kept their logs private. Donn’s group was initially not collecting logs, and he set up syslog-NG. Each had different but intersecting goals for their logging infrastructure, with Donn’s group focusing on centralizing logs and making searching logs simple, while Hartmann’s group wanted more privacy via role-based access. Both groups wanted the ability to trend, alert, and report via a Web interface.

They started out by buying two modest servers (Dell 2950s) equipped with more DRAM than Splunk recommended (16 GB) and one TB RAID5 array each. Hartmann said they initially just used syslog-NG to forward logs to the Splunk indexing software from different ports, and used the port separation as a way of indicating sources for search access

roles. In the next phase, they added Splunk agents to collect more data. They were both reluctant to use agents (what Splunk calls “forwarders”) because of their concern for performance and maintenance issues, but were gradually won over. Agents allowed them to collect system and network statistics from servers and to collect logs from DHCP and DNS servers that don’t use syslog for logging.

By their third phase, they had to purchase a new license from Splunk, enough to cover gathering 100 GB of log messages per day. This growth was partially the result of adding more agents to collect logs, as well as the addition of another group at Harvard. In their fourth phase, they added servers just as search front ends and added more servers for indexing and for searching. They have two of everything (as Splunk charges by logging volume, not by server)—they have a truly redundant system. They also switched to collecting logs using a Splunk agent on the syslog-NG servers, as the agent encrypts logs and transfers chunks of logs using TCP, making the system much more secure and robust. They are also learning how to write their own Splunk agents. In the future, they plan to add more security monitoring, collapsing some of their MRTG monitoring, but not RRDs, into Splunk, and getting rid of Cacti.

Prasanth Sundaram of Wireless Generation in NYC wondered about their archival policy. Hartmann answered that they follow their university’s policy for log retention, which does not require keeping logs very long. Donn pointed out that Splunk classifies logs as Hot, Warm, Cold, and Frozen, with Frozen logs not searched by default and good candidates for archiving. Sundaram then asked about issues in searching segmented indexes. Donn answered that Splunk hides segmentation from the user and searches Hot through Cold logs by default. Sundaram wondered how they decided to set up indexers. Donn answered that they chose to index by function: for example, all Linux servers in one, Cisco equipment in another, and so on. Hartmann said that it takes some time to figure out where you want to put stuff and that they should have been more methodical when they started out.

Someone wondered about the motivation for adding access controls for viewing logs. Donn explained that each college has its own bits of IT and they wanted to find a way to keep all that log data in one place. There are some security concerns, for example, in HTTP logs; people could find out students’ ID info. Hartmann said that when they first started out splitting out indexes, they discovered that the app team was doing searches much faster, because they were constrained to certain indexes. Donn explained that all users get a default index, as well as a group of indexes.

Matt Ryanczak of ARIN asked about capacity planning. The logging volume he could figure out, but the server sizes? Hartmann said that for the servers, the Splunk Web site has some pointers, and they beefed up the guidelines when they went with that. They did talk to Splunk reps after they had made plans, and they were told their specs would work well. Donn added that they had to buy systems that would give them at least a year’s headroom.

The duo ended by showing some more slides of examples of results from querying their Splunk servers, finding a problem with a new version of NTPD, finding a pair of chatty servers (using a top 10 Net speakers search), and finally showed how six MRTG graphs of firewall activity could be collapsed down to one.

Refereed Papers

Summarized by Shawn Smith (shawnpsmith@gmail.com)

PeerMon: A Peer-to-Peer Network Monitoring System

Tia Newhall, Jānis Libeks, Ross Greenwood, and Jeff Knerr, Swarthmore College

PeerMon is a peer-to-peer performance monitoring tool. It is designed for general-purpose network systems, such as systems that are normally run in small organizations or academic departments, typically running in a single LAN system. Each machine’s resources are controlled by the local OS, and each machine primarily controls its own resources. Each node runs a PeerMon process.

Tools built on PeerMon can allow good load balancing and increase performance. For example, one of the tools developed to utilize PeerMon is called SmarterSSH. It uses PeerMon data to pick the best machines to ssh into. More information about PeerMon and the tools that can use it can be found at <http://cs.swarthmore.edu/~newhall/peermon/>.

They were asked how they bootstrapped the process. It looks like there are lots of configs. In `init.d`, they have a script to start/stop/restart PeerMon. At startup, the PeerMon daemon starts and runs as a regular user. A cron job periodically runs to check if PeerMon is still running. If not, it runs the script. Did they run into any issues with race conditions? There aren’t any issues with race conditions, since they got rid of the old method of writing data to a file and having peers access the file.

Keeping Track of 70,000+ Servers: The Akamai Query System

Jeff Cohen, Thomas Repantis, and Sean McDermott, Akamai Technologies; Scott Smith, formerly of Akamai Technologies; Joel Wein, Akamai Technologies

This paper is about the Akamai Query system, a distributed database where all machines publish data to the database and the data gets collected at several hundred points. The Akamai platform consists of over 70,000 machines that are used for providing various Web infrastructure services. Akamai needs to have the ability to monitor the network so that if a machine goes down, they can find and solve the problem. The monitoring needs to be as close to real time as possible.

A cluster is a set of machines at a single datacenter that shares a back end. Some number of machines are designated in that cluster to provide data. Every machine has one query process and some number of processes that publish into Query. Every two minutes, the Query process takes a snapshot of all the rows of database tables that have been sent to it, puts them together, and sends them to the next level of hierarchy. Cluster proxies collect data for the whole cluster and put data together to be sent to the next level. Top-level aggregators collect data for the whole network. There are also static tables: machines that are up and down may change pretty frequently, but the set of machines that are supposed to be up only changes when they install or change hardware. Static tables describe the configuration of the network. Some machines are SQL parsers, whose job is to collect the queries and compute the answers.

Some purposes of using Query are mission-critical: down machines, misconfigurations, anything else that might go wrong that they might need to fix. They also need the ability to test out new queries, but don't want to issue test queries to the same place as alert queries, because they might take down a machine. Query's uses include: an alert system; graphical monitoring; issuing incident response queries to figure out where the problem is; looking at historical trends in usage.

For alerts, you can set a priority for each alert: 20% disk space is less urgent than 3%, which is less urgent than completely out of disk space; make sure a machine has a problem for a certain length of time before deciding it's an issue; email only notifications, do more proactive monitoring, and get operators in the NOC to directly go into action. Although most alerts are Akamai alerts, some are for customer monitoring. There are a few hundred machines in the query infrastructure and the system handles tens of thousands of queries every minute, tens of gigabytes turning over completely every two minutes.

How do they address the issue of scale from such a large system when making a query without overloading? They prefetch. Are they polling the system periodically or are the requests random in nature? Some of the uses of Query will be automated, but some of them will be people sitting at a desktop. When a query is made, are they able to tell me how fresh/complete the data is? Yes, there is a table that describes how old the data is. How often do people actually check that data? Depends on the application; if it's a person sitting down and querying the number of machines, it probably hasn't changed much in the past several minutes. But if there are alerts they probably care more about the staleness.

Troubleshooting with Human-readable Automated Reasoning

Alva L. Couch, Tufts University; Mark Burgess, Oslo University College and Cfengine AS

Architecture defines connections between entities, and troubleshooting requires understanding those connections. Human-readable automated reasoning provides a way to recall connections relevant to a problem, and to make and explain new connections via a strange sort of logic. An entity is defined as something someone manages, such as a host, a service, or a class of hosts and services. A relationship is a constraint between entities: for example, a causal relationship involves keywords "determines" and "influences," and a dependence relationship involves "provides" and "requires." An example of the notation is "host01|provides|file service".

It is strange because most attempts at computer logic attempt to translate English into logic and then reason from that, whereas this method translates architectural information into simple English and then reasons from that, without translating the English into logic. The main advantage is speed. The two claims of the paper are that the logic is easy to describe and compute and that the results of inference are human-readable and understandable.

Positive aspects of the system include: uses simple sentences; is very fast; produces a very quick answer.

Negative aspects of the system include: doesn't handle complex sentences; doesn't support complex logic; produces a relatively naive answer, the "shortest explanation."

In the future, they plan to work with field testing, coding in MapReduce for at-scale calculations and applying this to other domains, such as documentation. They welcome people trying out the prototype at <http://www.cs.tufts.edu/~couch/topics/> and letting them know how it works, how it could be improved, and what it should really do.

There was only one question: Does it connect with graphviz? Yes, very easily.

Invited Talks I

10,000,000,000 Files Available Anywhere: NFS at Dreamworks

Sean Kamath and Mike Cutler, PDI/Dreamworks

Summarized by Rik Farrow (rik@usenix.org)

Kamath explained that this talk came out of questions about why PDI/Dreamworks, an award-winning animation production company, chose to use NFS for file sharing. PDI/Dreamworks has offices in Redwood City and Glendale, California, and a smaller office in India, all linked by WANs. Solutions like FTP, rcp, and rdist don't work for read/write client access, and at the time NFS use was beginning, sshftp and Webdav didn't exist. And they still don't scale. AFS is not used in serious production and doesn't have the same level of support NFS does.

Their implementation makes extensive use of the automounter and LDAP to fill in variables used by the automounter. The same namespace is used throughout the company, but there are local-only file systems. Their two most important applications are supporting artists' desktops and the rendering farm, which impose different types of loads on their NetApp file servers.

Kamath said that they must use caching servers to prevent file servers from "falling over." They may have hundreds of jobs all accessing some group of files, leading to really hot file servers. For example, some sequence of shots may all access related sets of character data, and the only way to get sufficient NFS IOPS needed is through using caches.

Kamath described their California networks as 1200 desktops, split between RHEL and Windows, renderfarms, 100 file servers, 75% primary and 25% caching, and a 10 GigE core network, including some inter-site connectivity. *Megamind*, the most recently completed movie, required 75 TBs of storage. They have petabytes of active storage, and nearly two petabytes of nearline storage. Active storage is all SAS and Fibre Channel, with nearline composed of SATA.

Kamath went into some detail about how crowd scenes have evolved in digital animation. Earlier, crowd scenes were done by animating small groups of characters, cycling the same set of movements, and repeating these groups many times to provide the illusion of a large crowd of individuals. In *Megamind*, they selected random characters in large crowds and added motion to them, vastly increasing the number of files needed to animate crowd scenes: they went from tens of thousands

of files with cycling to millions and millions using random-based character selection.

Kamath summed up by saying that having a global namespace is key, so that users understand where to find things. They leverage the automounter and LDAP, and use local and cross-site caching to keep performance acceptable and reduce migration latency. They work with both developers and artists and keep their architecture flexible.

Doug Hughes of D. E. Shaw Research said that they had a similar problem, but not on the same scale. D. E. Shaw has lots of chemists and data, but they use wildcards with the automounter to point to servers via CNAMEs. Kamath responded that that worked really well for departments and groups, but it is hard to manage load that way. Jay Grisart of Yahoo! wondered if they would use filesystem semantics in the future. Kamath said that they will continue to use NFS, but are looking for alternatives for databases. Someone from Cray asked what configuration management software they use, and Kamath said it is currently a homegrown system. Hugh Brown of UBC wondered if desktops have gotten so powerful that they need caching. Kamath explained that desktops have gotten screaming fast (six core Nehalems, 24 GBs RAM, 1 GigE) and they can do rendering on them at night.

Invited Talks II

Data Structures from the Future: Bloom Filters, Distributed Hash Tables, and More!

Thomas A. Limoncelli, Google, Inc.

Summarized by Tim Nelson (tn@cs.wpi.edu)

A future version of Tom Limoncelli traveled back in time to remind us that we can't manage what we don't understand. In this talk, he introduced some technologies that sysadmins may see in the near future. After a quick intro, he reviewed hash functions (which produce a fixed-size summary of large pieces of data) and caches ("using a small, expensive, fast thing to make a big, cheap, slow thing faster"), then began discussing Bloom filters.

Bloom filters store a little bit of data to eliminate unnecessary work. They hash incoming keys and keep a table of which hashes have been seen before, which means that expensive look-ups for nonexistent records can often be avoided. As is usual for hash-based structures, Bloom filters do not produce false negatives, but can give false positives due to hash collisions. Since collisions degrade the benefit of the Bloom filter, it is important to have a sufficiently large hash size. Each bit added is exponentially more useful than the last, but re-sizing the table means re-hashing each exist-

ing key, which is expensive. Bloom filters are most useful when the data is sparse (hashes tend to be quite large—96, 120, or 160-bit hashes are common) and are commonly used to speed up database look-ups and routing.

Distributed hash tables are useful when data is so large that a hash table to store it may span multiple machines. Limoncelli gave several examples of what we might do with an effectively infinite hash table, such as storing copies of every DVD ever made or of the entire Web. A distributed hash table resembles a tree: there is a root host responsible for directing look-ups to the proper host. If a host is too full, it will split and create child hosts. Since the structure adjusts itself dynamically, the sysadmin does not need to tune it, provided enough hosts are available.

Key-value stores are essentially databases designed for Web applications. While a standard relational database provides ACID (atomicity, consistency, isolation, and durability), a Web application doesn't necessarily need each ACID property all the time. Instead, key-value stores provide BASE: they are Basically Available (it's the Web!), Soft-state (changes may have propagation delay), and Eventually consistent. Bigtable is Google's internal key-value store. Bigtable stores petabytes of data and supports queries beyond simple look-ups. It also allows iteration through records by lexicographic order, which helps in distributing work across multiple servers.

Matthew Barr asked about using memcache as a key-value store. Limoncelli answered that yes, memcache is a (simpler) key-value store using only RAM and is very useful if you don't need a huge amount of storage as Google does. Cory Lueninghoener asked how we should expect to see this stuff coming out. Limoncelli replied that sysadmins should expect to see it in open source packages. Much is already available and may be adopted faster than we expect.

Someone asked about key-value stores with valueless keys, and whether the fact that the key exists is useful information. Limoncelli answered yes, these are used in Bigtable queries and are helpful when sharding queries over multiple machines. Rick Bradshaw (Argonne) asked how complex Bigtable's garbage collection was and whether deleting data was possible. Limoncelli replied that as a sysadmin he doesn't think much about garbage collection, but that he does delete data and can explicitly request garbage collection if necessary.

Practice and Experience Reports

Summarized by Robyn Landers (rblanders@uwaterloo.ca)

Configuration Management for Mac OS X: It's Just UNIX, Right?

Janet Bass and David Pullman, National Institute of Standards and Technology (NIST)

David Pullman started with a quick history of configuration management in the NIST lab. This has been a fairly easy task on traditional UNIX variants with the usual tools such as Cfengine, but not so with older versions of Mac OS, which were mostly hand-maintained. The advent of Mac OS X made it seem as though it should be doable, as the title of the talk indicates. Pressure to ensure compliance with security standards was also a driver.

Pullman outlined their progress towards achieving two goals: getting settings to the OS and managing services. This started with simple steps such as one-time scripts, but these were thwarted by lack of persistence across reboots or per-user applicability, for example. This led them to the investigation of plists, the configuration files for OS X. They supplemented meager documentation available with an assortment of tools, including interesting ones such as Fernlightning's fseventer and Apple's own dscl, as well as OS X utilities such as Workgroup Manager and launchd. Such tools enable detection, examination, and modification of the plists involved in configuring a given service or application. Brief comparisons were drawn with Solaris and Linux service managers.

Some difficulties arising from the inconsistency of parameter values in plists were pointed out. Pullman suggested that perhaps the long lead time before OS X Lion's release gives us an opportunity to influence Apple in this regard. Meanwhile, the approach seems to be successful so far for NIST.

Rick Bradshaw from Argonne National Lab asked whether they started with a custom-built OS image or barebones. Pullman said they don't have enough control at purchase time to inject a custom image.

Lex Holt of the London Research Institute described their environment with about 700 Macs. They use the Casper suite from JAMF for building images, supplemented by their own scripting. Unpredictable arrival of new hardware complicates image building. Pullman's group also looked at Casper, but they prefer to "know where the knobs are" themselves, and sometimes need to act more quickly (e.g., for security issues) than image-based method might allow.

Another audience member asked what version of Cengine NIST used. They used version 2, but they're talking to Mark Burgess about some enhancements for version 3, and would be happy to share this. Robyn Landers, University of Waterloo, mentioned that they are getting started with DeployStudio and JAMF Composer for image and application management on Macs. Pullman's group had not yet looked at DeployStudio, but they are interested.

Anycast as a Load Balancing Feature

Fernanda Weiden and Peter Frost, Google Switzerland GmbH

Fernanda Weiden began the talk by explaining the motivation for using anycast for failover among load-balanced services: availability, automatic failover, scalability. Amusingly, management buy-in occurred after connectivity to a sales building went out. The combination of anycast and load balancing brought benefits such as simpler routing configuration and elimination of manual intervention for failover. Elevation to desirable service standards was achieved by distributing servers and a load balancer to each site, along with centralized management.

Peter Frost described the implementation. An open source software stack runs on Linux. The Linux-HA Heartbeat mechanism manages NICs and management software (helping avoid the "equal cost multi-path" problem), while `ldirectord` manages services on the back-end servers. The Linux IPVS kernel module helps with load balancing, and Quagga network-routing software was a key piece for managing service availability, ensuring that peering is always in place while secondaries are kept inactive until needed.

Now that the authors have completed this project, their methodology has proven to be readily extensible to other services, thanks in large part to Quagga's features.

An audience member from Cisco asked how long it takes for the routers to reconverge in case of an outage. If the outage arises from a clean shutdown, it takes less than one second, but in the case of a dirty shutdown (e.g., power failure), it takes 30 seconds. Only one side of the load-balanced pair is active at a time, in order to avoid sharing connection tables.

Rick Bradshaw of Argonne National Lab asked about logging for service monitoring. Indeed, it is logged, and the authors contributed an enhancement to `ldirectord` code on this. They added the ability for `ldirectord` to capture the underlying information about events and health check failures rather than merely the fact that such events occurred. This is reported in standard `syslog` format.

David Nolan of Ariba has been making much use of anycast and wondered about monitoring to verify proper route

announcements and propagation. NIST has a small monitoring server at each site, in-band (from the user's point of view), to discover hostnames of the servers. If they don't match with what's expected, it issues an alert.

David Lang of Intuit asked whether the authors considered ClusterIP, given their active/passive arrangement. No, they never needed to consider active/active. They had some concern about security, given the unauthenticated connection between load-balanced servers sharing their data.

Session host Aileen Frisch asked the speakers what's next for them, since their work on this project is finished. Thanks to the extensibility of their system, their colleagues have been successfully adding services such as LDAP, HTTP proxy, and logging into their load-balanced methodology.

Nolan asked about BGP filtering: was there difficulty getting cooperation from network administrators regarding advertising routes? What about the risk of taking over other legitimate IP address spaces not belonging to them? Fortunately, the network people at NIST were friendly and cooperative, and their strict management of routing maps helps ensure safety.

Nolan said that his organization has Windows working very well as back-end anycast servers after a difficult initial setup and wondered about Windows at NIST. All NIST's anycasting is on the load-balancer level, not on the back end. There was no need to worry about difficulty with Windows for configuration of the network stack.

The final question touched on deployment. Did they ship preconfigured servers to their other sites and ask those sites to just trust them and plug them in? No; at first they used a commercial appliance, but the ordinary Linux servers they switched to later were easily configured remotely.

iSCSI SANs Don't Have to Suck

Derek J. Balling, Answers.com

iSCSI SANs typically "suck" because SCSI is very sensitive to latency and Ethernet often has bursts of poor performance, leading to latency. Derek Balling presented his site's experience with servers, iSCSI SAN devices, and the network topology that connects them. The initial approach had various drawbacks that they were able to overcome by careful redesign and reimplementation. Balling showed connection diagrams illustrating before and after connectivity, and transition steps. This helped make the situation more understandable. He also gave some practical advice that should help one carry out any non-trivial project more successfully.

Every server has two network interfaces for ordinary data and two more for the SAN. If a link fails, its redundant link

is activated and the network spanning tree protocol (STP) reconverges to use it. The advantage is that moved to every device has multiple paths with automatic failover to what it needs. Ironically, the concomitant disadvantage is that the time required for STP to converge causes the dreaded latency that iSCSI cannot tolerate well. This was especially noticeable when adding new nodes to the network: virtual machines relying on the SAN would die while waiting for STP to reconverge.

They dodged this by using uplink failure detection in the network equipment to avoid triggering STP. However, the problem repeated whenever rebooting switches, and when adding new switches that don't support uplink failure detection and thus required STP to be enabled. A redesign was thus in order.

The redesign called for a flat network connection topology, with a separate network (rather than merely a separate VLAN) for the SAN. The network administrators were leery of this, so Balling's group had to show how it would work acceptably. Another challenge was the desire to carry out the reorganization live, since the environment was already in production. Thanks to thorough planning and rigorous adherence to the plan during execution, they succeeded in making the transition smoothly. Perhaps unfortunately, it went so well that their management now expects this level of success all the time!

Balling concluded his presentation by emphasizing the rigor of the planning. Everything was drawn out on a whiteboard, and the team verified connection paths at every step in the process. Nothing was rushed; time was given for reconsideration and peer review. Execution was equally rigorous. In the heat of the moment when one might forget why the order of certain steps matters or be tempted to try an apparent shortcut, it's essential to stick precisely to the steps laid out in the plan. System administrators, unlike network administrators, might be less accustomed to such rigor.

In response to Balling's talk, David Nolan of Ariba observed that system administrators and network administrators don't always realize what they can do for each other. How can that deficit be overcome? Balling suggested that simply socializing with one another might help.

Invited Talks I

Operations at Twitter: Scaling Beyond 100 Million Users

John Adams, Twitter

Summarized by Shawn Smith (shawnpsmith@gmail.com)

What's happened at Twitter in the last year: a lot of work on specialized services; made Apache work much more efficiently; moved to Unicorn; changed the handling of Rails requests; and added a lot more servers and load balancers.

Everything in new Twitter is over AJAX; you won't see submit tags in the source. They used logs, metrics, and science to find the weakest points in applications, took corrective action using repeatable processes, and moved on.

Adams said, "We graph everything." They used Mathematica curve fitting to see when they would hit the unsigned integer boundary. He then said, "The sooner you start using configuration management, the less work you'll do in the future, and the fewer mistakes you'll make." About two months into Twitter, they started using configuration management with Puppet. A fair number of outages occurred in the first year caused by human errors. Now something is wrong if someone is logging into a machine to make a change.

They also use Loony, which connects to machine db and ties into LDAP, allowing them to do things en masse across the entire system. With it you are able to invoke mass change. They only run Loony if something has gone extremely wrong, or to find every machine in a cluster that is a mail server that happens to be running Red Hat.

They use Murder, as in a murder of crows. They use BitTorrent for deployment and can deploy to thousands of machines in anywhere from 30 to 60 seconds. They have moved away from syslog, as "syslog doesn't work very well at the loads that we're working at." They use Scribe for HTML logs, and Google analytics on the error page. They modified headers in Ganglia so that everyone knows exactly when the last deploy went out. For every feature you want to deploy at Twitter, it has to be wrapped up in a darkmode or decider flag with values from 0 to 10,000, 10k representing 100% deploy. Rails runs the front end of Twitter, and the back end is Scala Peep, which allows you to dump core on a memcache process. They like Thrift because it's simple and cross-language. Gizzard allows them to shard across hundreds of hosts and thousands of tables, distributing data for single and multiple users. Adams recommends mounting with atime disabled, as "mounting a database with atime enabled is a death sentence."

What steps did they take to go from unscalable to where you are now? They looked at data metrics; get metrics on everything. Are Scribe Hadoop patches available? Yes, on github. Do they have good management support to allow them to work on projects like these? The DevOps movement is about getting cultural changes in place for the better and, yes, management has been supportive. What have they done that is an acceptable failure? How many Web servers can they afford

to lose? If they lost $n\%$ of our servers, they wouldn't want to alert. They need very fast timeouts. Can they comment on Scala? Scala is a Java-like language that runs inside of the JVM, so the scaling constraints are known; there are a number of things inside Scala that are designed for concurrency that our developers wanted to take advantage of. Twitter has a fondness for oddball functional languages; it has some Scala and some Haskell floating around.

Invited Talks II

Er, What? Requirements, Specifications, and Reality: Distilling Truth from Friction

Cat Okita

Summarized by Scott Murphy (scott.murphy@arrow-eye.com)

This was a humorous yet informative overview of why we need requirements and specifications. Cat opened with the rather blunt question, “Why do we bother with these things?” After all, this is all paperwork and paperwork sucks and this is what nontechnical people do and then inflict on us. This set the context for the talk, moving from theory to practice to reality.

Beginning with theory, Cat displayed a neat little slide showing the “Quick’n’Dirty” overview:

Goals—Why are you doing this?

Requirements—What means that you've met the goals?

Specifications—How do you meet the requirements?

Implementation—Perform the task as per the specifications.

Review—Does the work match the requirements/solve the problem/meet the goal?

Completion—You are done.

Cat continued with a more detailed discussion of the above points, starting with goals. The point of defining the goal is to determine why you are doing something, what you are trying to do and/or what problem(s) you are trying to solve. What makes a good goal? Several examples were given, showing that this can be a very nebulous item. Requirements were next: what meets the goal(s), what is success, what are the limitations, who is involved, and how can you tell that you are done? More examples and discussion followed, again showing that this can be a slippery area. Next up were specifications, getting a little more to the part techies prefer. This part defines how we are to meet the requirements and should be detailed, specific, and prescriptive. Implementation follows specifications and covers the “getting it done”—do stuff, build, test, deploy. Then it's review time. Did we hit our goal,

did we meet the requirements, did we follow the specifications? If we can answer that in the affirmative, then we have arrived at completion. The visual is a slide showing a sandy beach and a pair of lounge chairs.

Cat then talked about best practices for requirements and specifications, boiling the whole concept down to getting the most bang for your buck. In order to influence goals, requirements, and specifications, you need to participate in the initial planning meetings where these items are discussed. By participating, you get to influence the project. A project goes much more smoothly if everyone is on the same page, so an air of cooperation and communication is necessary. Document everything, preferably a living document that captures choices, decisions, and reasons why things do not meet initial requirements. It is important to have a common set of definitions and understanding of the goal. Using overloaded jargon or unclear requirements will result in nobody knowing what to do or how to do it. The goal should be appropriate (slide of a fish on a bicycle) and it should be kept short. When defining goals, ask people for input, don't tell them, as that removes a potential information vector. Once all of this has taken place, you need to agree on the goals. This will keep surprises to a minimum—ideally, to none. In summary, a goal should be one or two short sentences that anybody can understand.

Once the goals have been identified, some housekeeping is in order. You need to clarify the goals, define your audience, specify conditions for success, and set limits to the scope. Clarifying the goals will bring focus to the project, turning the goals into something useful and describing what will meet the goals. Defining the audience will identify who actually cares (or should), who needs to be involved and who the project is for. Specifying the conditions for success serves a double purpose: first, to let you know that you have met the goal, and, second, to provide conditions for completion, as both are not necessarily the same. Limits are also a very important item to specify. We want to keep a rein on scope creep, people involved, external items, money, and time. Projects tend to expand without bounds if you do not have limits specified up front. A goal should also be realistic, relevant, measurable, and unambiguous. This brings us to “How do we meet goals?” In order to meet goals, we need to meet the requirements, stay within the project limits, and have details to measure. An example of specific requirements would be “Use Apache 2.x with Tomcat to serve dynamic Web content” vs. “Use a Web server.” Requirements should be appropriate, such as describing a Web platform to be a standard readily available system vs. a Cray. They need to be sane. Turning a jury-rigged proof-of-concept into your production platform is only asking for trouble (slide of the original Google setup—scary).

Where do requirements come from? The answer is the project initiators, interested parties, and potential customers. Cat illustrated with an example—“Build a Death Star” followed up with the requirement that it be made from pumpkins. Some discussion occurred at this point, but in the end, only pumpkins were available. A follow-up slide showed a jack-o’-lantern carved as a Death Star replica. This was identified as having a missing component, as it would not be visible at night, leading to the requirement for lights. It was then determined that there is no room for lights inside, so it must be hollowed out prior to installing lights. This leads to the specification to use a spoon to hollow it out. The spoon is identified as the wrong tool. A sharpened spoon is specified next and the pumpkin is successfully hollowed out, the lights are installed and we end up with a Death Star made from a pumpkin that can be illuminated. The goal is met.

At this point, Cat introduced reality to the mix. In most projects, you end up with some choices to make—you can build, buy, or borrow to meet goals. You will probably use a combination of all of them and more. In the real world, communication is very important to the success of a project. If you are not communicating, you lose sight of the goals. Scope creep can intrude, resulting in goals not being met, cost overruns, the wrong people getting involved, etc. Documentation is your friend here. If you can’t say “No,” document the new requirement. Cat mentioned a “Ask Mom/Ask Dad” concept that can happen during a project—ask a group how things are going and, not liking the answer from one person, ask the next person. This can be fought with a single point of contact. Politics comes into play as well. I’m ignoring you is a political game—you didn’t format your request properly, I don’t have the resources to handle that right now, etc. Sometimes this can be handled with a discussion, sometimes by kicking it upstairs. Projects also suffer from a level of confusion. If fuzzy language is utilized, clarification is necessary. Words mean different things to different people. Context is important and so is culture. Cat referred to this as craft knowledge and craft-specific language. People involved can be out of touch with reality—10ms round trip time between San Francisco and New York as a requirement. Physics may make this difficult. We get hit with solutions looking for a problem and we can experience consternation brought on by bad assumptions, missing limits, and adding more people to a late project. You can also be hit with “death from above,” where things will take on a “new direction,” and “It will be completed by next Tuesday.” All you can do is get clarification, modify requirements, ignore some requirements, and document everything.

Cat presented a couple of additional examples of projects to illustrate the points presented above. We should also be

aware of odd requirements in RFPs (requests for proposals). The idea of the RFP is to solicit proposals that meet established criteria. Specifications as to Blue M&Ms or RFC1149/RFC2549 are occasionally added to ensure that the proposal meets established criteria and that the RFP has been read and understood, sort of a checksum for details.

Someone asked how to define goals and requirements for a project. How do you do it? Cat suggested starting with the Why (goal) and someone who cares about the project, the person who will be driving it. This person will have to come up with a couple of things to at least get people talking about the project. Who do I think I need to involve? This is usually straightforward. You take the people and start the discussion (even if it’s wrong), and you get, “We can’t do this,” signifying a limit; “It must,” identifying a requirement; “It should,” identifying a nice to have. Now we have a requirements list, so go back and forth between people to clarify requirements. Ask them about their part rather than tell them what they have to do. The person who cares has been documenting, right? Once this is finished, it’s time to horse trade (budget). What do we give up? How do we balance out resources? You stop when you get to the point where you are quibbling over vi vs. Emacs; the requirement is that we have a text editor.

Another person commented that the people who seem to have the easiest time learning to do this are ex-military, possibly because this is like an operational briefing: Why are we doing this? Why are we going to this place? What are we going to do there? What are we allowed to do when we are there? What equipment do we have? What is our exit plan if things go bad? How do we declare we have had a successful mission?

Steven Levine, of Red Hat, said that he is not a system administrator but a tech writer. In discussing how you know if you have met your goals, people responded with interesting things about how the goals shift. In his work it’s more an issue of compromising the goals. Every day he makes one compromise. It’s not that goals shift, they get compromised. Does this apply to system administration as well? I would think this would be more black and white. Cat replied that this applies to absolutely everybody. Levine asked, “How do you keep from feeling that each compromise ‘eats at your soul’? How can you sleep well at night?” Cat said that one of the things that always bothers her about that is when it’s not clear that you have been compromising. When people look at it and say we didn’t end up quite where we wanted to and stuff went somewhere. We don’t know why and we don’t care why vs. we made a clear decision about this. We’ve said unfortunately the trade-offs are all here. We are going to have to make a trade-off. Let’s go back and say, “You know those goals we had or those requirements we had? We have to change them because we have these limits that we have run

into.” I may not have been as clear as I meant to be that a lot of this does end up being an iterative process, so you go through your requirements and say, “Hang on a second. With these requirements, there is no way that we can match that goal.” Say I have a budget of \$100,000, can I build a death star? I can meet some of the requirements. Can I build it in outer space? Probably not. I’m not going to argue that it’s not frustrating,

John Detke from PDI Dreamworks said, “As sysadmins, we do a lot of research as we are not really sure what we are going to do so we develop specifications which change as we discover limitations. At what point do you go back and change the specs or the goals? Are there guidelines for how to do that without going crazy?” Cat asked, “How spectacular is your failure? If we can’t do this at all, you probably want to go back and ask if the goal is realistic. If this happens at the requirement/specification phase, I like to say this is a requirement I absolutely have to have. If we can’t do this, we stop the project until we can figure out how to do it. Being able to say here are my blocking/stopping points makes it easier to identify where we have to stop and consider what we have to do. Typically, it becomes a judgment call as to major or minor problem.”

Someone asked what her favorite tool was for capturing/manipulating all this stuff. Cat said that she’s a Mac user and likes Omni-Outliner. In a corporate environment, you may be required to use Word, MS Project, Visio, etc. She’s even used vi to outline, so it’s whatever you are comfortable with. Pick your poison.

Invited Talk

Using Influence to Understand Complex Systems

Adam J. Oliner, Stanford University

No report is available for this talk.

Invited Talk 1

Scalable, Good, Cheap: Get Your Infrastructure Started Right

Avleen Vig, Patrick Carlisle, and Marc Cluet, woome.com

Summarized by Misha Zynovyev (zynovyev@stud.uni-heidelberg.de)

This talk focused on important issues IT start-ups face in infrastructure design in their first 6 to 12 months. Avleen Vig of Etsy started by emphasizing the importance of setting up a workflow process. He argued that a long ticket queue which feels like eternity to process is better than to forget even a single thing. He also explained how much it pays off to put every aspect of operations code into a version control system and automate as much as possible. The importance of hiring

the right people and establishing active communication within the team was also mentioned.

The first rule of making your infrastructure scalable, according to Vig, is to separate all systems from each other and layer them. He declared himself a proponent of the DevOps movement and advocated fast and furious release management. MTTR (mean time to recovery) was compared to MTBF (mean time between failures) as time to assemble a Jeep vs. time to assemble a Rolls-Royce. Infrastructure needs to have the shortest possible MTTR. Data-mining of all logs is very important, as well as graphing all dynamic monitoring information. Besides monitoring everything which has broken at least once, one has to monitor all customer-facing services. Vig also said from experience how crucial it was to monitor the fastest database queries in addition to slow ones. A very high frequency of fast queries can have a stronger impact on performance than occasional slow database queries. It was advised not to rely too much on the caching layer in the database architecture. Cache must be disposable and rebuildable. The back-end database has to be ready to withstand the load if the cache is gone. The Northeast Blackout of 2003 was given as an example of a cascading failure.

Marc Cluet of WooMe took the floor to talk about database scaling. One of the first questions sysadmins face is whether to use relational or NoSQL databases. At WooMe both types are used. The stress was put on how dangerous it is to let databases grow organically. After warning about handling many-to-many tables, the speaker admitted that mistakes in database design are inevitable, but one has to be prepared for them. Adding many indexes is not a path to salvation, since one pays in memory for being fast. Although not all data can be partitioned, partitioning becomes necessary with database growth. At WooMe, data is partitioned by date. Disk I/O is, of course, the thing to be avoided. Hardware load balancers are too expensive for start-ups and there are plenty of software solutions for load balancing. But none of them will give some of the advantages of reverse proxies that are particularly useful for more static data. At WooMe they use Nginx.

At the end Marc Cluet explained the benefits of dividing Web clusters, how it adds more flexibility to maintenance, and how problems can be contained to a fraction of resources. He then proceeded to stress the importance of automation, adoption of configuration management tools, and version control systems, just as Avleen Vig had done before him. At WooMe they use Puppet and Mercurial. The talk was finished by mentioning clouds, which are used by WooMe for backups and potentially could be used for further scaling.

Jay Faulkner from Rackspace asked about the size to which WooMe and Etsy have scaled. WooMe's infrastructure scaled from 10 to 100 servers in two years; Etsy has 6 million active users. Doug Hughes of D. E. Shaw Research commented on the SQL vs. NoSQL debate, and that according to experts it is more appropriate to compare transactional and nontransactional databases, leaving the SQL language out of it. For Avleen Vig it matters what tools are the best for the job and how to support what the business requires. Duncan Hutty of Carnegie Mellon University asked how to distinguish premature from timely optimization, since Avleen Vig pointed out at the beginning of the talk that technical debt is not necessarily that bad and can be more appropriate than premature optimization. Vig answered that one has to estimate how long one's work will be in place, if it is going to disappear shortly. If it stays for a longer time it can be beneficial to spend a bit more time for optimization.

Reliability at Massive Scale: Lessons Learned at Facebook

Robert Johnson, Director of Engineering, Facebook, Inc.; Sanjeev Kumar, Engineering Manager, Facebook, Inc.

Summarized by Matthew Sacks (matthew@matthewsacks.com)

On September 23, 2010, the Facebook Web site was down for about 2.5 hours due to an error introduced in an automated feedback mechanism to keep caches and the Facebook databases in sync. Facebook's Robert Johnson and Sanjeev Kumar presented the lessons learned about designing reliable systems at a massive scale. Facebook currently serves about the same amount of Web traffic as Google. Johnson and Kumar decided to present their findings to the technical community at LISA as a learning experience, which was quite commendable for a company of this stature. It turns out that a feedback loop designed to prevent errors between the cache and the database was triggered; however, invalid data was in the database, so the data integrity logic went into an infinite loop, making it impossible to recover on its own. Ultimately, the site had to be taken down in order to correct the data corruption problem.

Most public technical presentations focus on what was done right, rather than lessons learned from what was done wrong. By reviewing what happened, a lot of progress can be made to firm up these systems and ensure that these problems do not happen again. Johnson said, "We focus on learning when things go wrong, not on blaming people." At Facebook, Johnson explained, when the blame is taken away, the engineering team is much more engaged on what can be improved so that these problems do not happen in the future.

Closing Session

Look! Up in the Sky! It's a Bird! It's a Plane! It's a Sysadmin!

David N. Blank-Edelman, Northeastern University CCIS

Summarized by Rudi van Drunen (rudi-usenix@xlexit.com)

David Blank-Edelman compared the modern-day sysadmin with the superheroes who live in comic books. In this very humorous presentation, David started off with a discussion of the superpowers the comic heroes have and how they map to the tool set of the modern-day system administrator. Also, the day-to-day life of a superhero was compared to the day-to-day life of the sysadmin, including the way to nurture the superpowers by means of mentoring. Important things were discussed such as how to use one's superpowers and super tools to do good, with strict ethics, very much as we sysadmins do.

The presentation was filled with snippets from comic books, movies, and soundbites, and concluded with some hard scientific evidence. This presentation was best experienced, and watching the video on the LISA '10 Web site is encouraged.

Workshop Reports

Workshop 1: Government and Military System Administration

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, Department of Commerce, Nebraska Army National Guard, Raytheon, the Norwegian government, Science Applications International Corporation, and the USENIX Board. This was the third year the GOV/MIL workshop has been held at LISA.

The GOV/MIL workshop creates a forum to discuss common challenges, problems, solutions, and information unique to the government sector, where participants may be able to gain and share insight into the broad range of system administration requirements that arise from a government perspective. The GOV/MIL workshop is an opportunity for diverse government, military, and international organizations to come together in a unique forum; it's not common to have highly technical staff from .mil, .gov, .com, and non-US agencies at the same table to candidly discuss everything from large data sets to organizational complexity to staffing and educational challenges. All expected to find similarities and hoped to be exposed to new ideas, and for the third year no one went away disappointed.

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 safe reminder to state what the appropriate level of discussion is for any new situation. 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 configuration management, technical challenges in classified environments, impact of the Sun/Oracle merger, cloud computing, and disaster recovery. Organizational and policy hot topics centered on technology considerations for foreign travel, rapidly changing information assurance policies, VIP users, and unfunded mandates from external agencies.

All attendees presented what types of personnel their respective sites or companies are seeking to hire, including discussions of what types of education and training are currently desired. Several had positions to fill, and almost all of them required security clearances. Hiring information and career Web sites were shared.

Our final effort was to respond to a challenge from the USENIX Board. Alva Couch said that USENIX is highly motivated to reach out to the GOV/MIL community but that they have found themselves unable to find the right way in. The GOV/MIL workshop conducted a round-robin brainstorm session and produced a list of ten recommendations for Alva to take back to the Board for consideration.

The final topic of discussion was to determine if there would be sufficient interest in this workshop to repeat it at LISA 2011. It was agreed that it was a valuable experience for all attendees and that all would support a follow-on workshop. The LISA GOV/MIL wiki is at <http://gov-mil.sonador.com/>. Please contact Andy Seely at govmil@sonador.com for more information about the growing USENIX GOV/MIL community of practice and to help shape the agenda for GOV/MIL 2011.

Workshop 6: Advanced Topics

Summarized by Josh Simon (jss@clock.org)

Tuesday's sessions began with the Advanced Topics Workshop; once again, Adam Moskowitz was our host, moderator, and referee. We started with our usual administrative announcements and an overview of the moderation software for the new folks. Then we went around the room and introduced ourselves. In representation, businesses (including

consultants) outnumbered universities by about 4 to 1 (up from 2 to 1); over the course of the day, the room included seven LISA program chairs (past, present, and future, up from six last year) and seven past or present members of the USENIX, SAGE, or LOPSA Boards (down from nine last year).

Like last year, our first topic was cloud computing. The consensus seemed to be that there's still no single definition for the topic. Most of the technical people present perceived "cloud" to mean "virtualization" (of servers and services), but for nontechnical or management it seems to mean "somewhere else," as in "not my problem." Regardless of the definition, there are some areas that cloud computing is good for and some it isn't. For example, despite pressure to put everything in the cloud, one company used latency requirements for NFS across the Internet to identify that something couldn't work as a cloud service. They can then escalate up the management stack to re-architect their applications to get away from the "it's always been done that way" mindset.

Some environments are using "cloud" as an excuse to not identify requirements. However, even with environment-specific cloud services, providing self-service access (as in, "I need a machine with this kind of configuration") and not having to wait weeks or months for the IT organization to fulfill that is a big win. IT organizations are often viewed as onerous (or obstructionist), so going to the cloud allows the customers to get around those obstructions. One member noted that the concept of cloud as virtualized servers and services isn't new—look at Amazon and Google for examples—and yet research is saying "it's all new." In academia, the cloud is "good for funding." (Even virtualization isn't new; this was done on mainframes ages ago.)

That segued to a discussion about how to implement this. We need to consider the security aspect: what's the impact of sending your stuff somewhere else, what are the security models and controls, is old data wiped when you build new machines, is the data encrypted across the Net, and so on. There's also the management assumption that services can be moved to the cloud with no expense, no new hardware, no new software, no downtime, and no problems. One tongue-in-cheek suggestion was to relabel and rename your hardware as cloud001, cloud002, and so on. Management needs to be reminded that "something for nothing" isn't true, since you need to pay for infrastructure, bandwidth, staffing, and so on. "Cloud" may save budget on one line item but may increase it on others.

After our morning break, we resumed with a quick poll on smartphone use. Among the 31 people in the room, the break-

down was Android 11, Blackberry 2, dumb 5, iPhone 8, Palm 3, Symbian 1, no phone 1.

Next we did a lightning round of favorite new-to-you tools this past year. The answers this year ranged from hardware (Android, hammers, iPad, and Kindle) to software (certain Firefox add-ons, Ganetti, Hudson, Papers, Puppet, R, Splunk, and WordPress) to file systems (HadoopFS, SANs, sshfs, and ZFS on FreeBSD), to services (EC2), as well as techniques (saving command history from everywhere).

Our next major discussion topic was careers in general: jobs, interviewing, and hiring. One hiring manager noted that they had a lot of trouble finding qualified people for a high-performance computing sysadmin position. Many agreed it's common to get unqualified applicants and to get few women and minorities. Even with qualified applicants (such as senior people for a senior position), it's problematic finding the right fit. Another hiring manager noted they're seeing more qualified applicants now, which is an improvement from 3 to 4 years ago.

This led to a discussion of gender balance in the field, and sexism in general. The "you need a tougher skin" feedback seems common out in the world, but one participant noted that saying that would be grounds for termination at his employer. Another person hires undergrads at his university to train them as sysadmins, but in nine years has had only two female applicants. Part of the problem is the (American) cultural bias that tends to keep women out of science and technology because "girls don't do that."

One question is whether the problem is finding people or recruiting people who later turn out to be a poor fit. The discussion on interviewing had a couple of interesting tips. If a candidate botches an interview, closing the interview instead of continuing is a courtesy. Not everyone treats "assertive behavior" as indicative of "passion," so watching your communication style is important. Over-assertiveness can be addressed by interpersonal training, and supervisor training to be able to pull someone back is a good idea.

We segued into the fact that senior people need to have an option other than "become a bad manager" for promotions. Most of us in the room have either been or are managers. Several of us see the problem as being that the technical track has a finite limit and a ceiling; one company has a "senior architect" position that's the technical equivalent of VP. Some think the two-track, technical or management, model is a fallacy; you tend to deal with more politics as you get more senior, regardless of whether you're technical or management.

Next we discussed automation and DevOps. There's a lot of automation in some environments, both of sysadmin tasks and network tasks, but it's all focused on servers or systems, not on services. Many places have some degree of automation for system builds (desktops if not also servers) and many have some degree of automation for monitoring, with escalations if alerts aren't acknowledged in a timely manner. There's a lot of automated configuration management in general; a quick poll showed that 22 of 30 of us think we've made progress with configuration management in the past five years. At Sunday's Configuration Management Workshop, we seemed to have the technical piece mostly solved but now we're fighting the political value. Many people work in siloed environments which makes automating service creation across teams (such as systems, networks, and databases) difficult.

One participant noted that many sysadmins have a sense of ownership of their own home-grown tool, which can work against adopting open-source tools. With the move towards common tools—at the Configuration Management Workshop, 70% of people had deployed tools that weren't home-grown—you can start generalizing and have more open source than customization. But capacity planning is hard with the sprawling environment; you need to have rules to automate when to look for more servers. It was also pointed out that automation can mean not just "build server" but also "deploy and configure database and application."

We have seen DevOps skyrocket over the past couple of years; finally sysadmin is getting some recognition from developers that these problems are in fact problems. We may be able to steal their tools to help manage it. As sysadmins we need to lose our personal relationships with our servers. We should be writing tools that are glue not the tools themselves. Moving towards a self-service model (as in the cloud discussion above) is an improvement.

Sysadmins often write software but aren't developers; the software may not be portable or may solve a symptom but not the cause, and so on. Also, many good sysadmins can't write a large solution. There's been a long-standing stand-off between sysadmins and application developers. It's coming to the point where the application developers aren't getting their requirements met by the sysadmins, so the sysadmins need to come up with a better way for managing the application space. The existence of DevOps recognizes how the industry has changed. It used to be that developers wrote shrink-wrapped code that sysadmins would install later. Now we're working together.

One person noted that DevOps is almost ITIL-light. We're seeing ITIL all over; it's mostly sensible, though sometimes

it's process for the sake of process. That segues into a big problem of automation—people don't know what they actually do (as a sysadmin, as purchasing, as hardware deployment, software deployment, and sometimes even the end user); arguably that's a social problem, but it needs to be solved. Beyond that, DevOps is another way of fancy configuration management.

It was noted that DevOps is as well-defined as “cloud.” Several people distinguish between system administration (“provide a platform”) and application administration (“the layer on that platform is working”). We ended with a sanity check; most of us think, in the general case, that a hypothetical tool could exist that could be complete without requiring wetware intervention.

After our lunch break, we had a discussion on file systems and storage. The discussion included a reminder that RAID5 isn't good enough for terabyte-sized disks, since there's a statistical probability that two disks will fail, and the probability of the second disk failing before the first one's finished rebuilding approaches unity. RAID5 is therefore appropriate only in cases of mirrored servers or smaller disks that rebuild quickly, not for large file systems. We also noted that Drop-Box (among others) is winding up on the machines of Important People (such as vice presidents and deans) without the IT staff knowing; it's ubiquitous, sharing is trivial, and so on. It's good for collaboration across departments or universities, but making the users aware of the risks is about all we can do. Consensus is that it's good for casual sharing; several recommended preemptive policies to ensure that users understand the risks. In writing those policies, consider communications from the source to the target and all places between them, and consider the aspects of discovery (in the legal sense), and whether the data has regulatory requirements for storage and transmission (such as financials, health, student records). Depending on your environment, much of the risk analysis and policy creation may need to be driven by another organization (risk management, compliance, legal, or security), not IT.

Our next discussion was a lightning round about what surprises happened at work this year. Answers included coworkers at a new job being intelligent, knowledgeable, and understanding of best practices; how much the work environment, not the technical aspects, matter; IPv6 deployment and the lack of adoption (only six people use IPv6 at all and only three of them have it near production); moving from Solaris to Linux because the latter is more stable; moving from sysadmin into development; new office uses evaporative cooling and it works; Oracle buying Sun and the death of OpenSolaris; organizational changes; project cancellations;

and virtualization allowing security to push services into the DMZ faster than expected.

After the afternoon break, we resumed with a discussion on security. Most think the state of the art in security hasn't changed in the past year. There have been no major incidents, but the release of Firesheep, the Firefox extension to sniff cookies and sidejack connections, is likely to change that. (This ignores the “Why are you using Facebook during the workday or in my classroom” question.) Cross-site scripting is still a problem. Only one person is using NoScript, and only a few people are using some kind of proxies (e.g., SOCKS). Most people use Facebook, but nobody present uses Facebook Applications; however, the workshop attendees are self-selected security-savvy people. We also noted that parents of young kids have other security problems, and some people don't want to remember One More Password.

Our next topic was on the profession of system administration. We have some well-known voices in the industry represented at the ATW and we asked what they think about the profession. The threats to sysadmins tend to fall into three categories: health, since we've got mostly sedentary jobs and many of us are out of shape; the industry, where there's enough of a knowledge deficit that the government has to step in; and the profession, as sysadmins don't seem to have a lot of credibility. Sysadmins don't have a PR department or someone from whom the *New York Times* can get a quote. Outsourcing was identified as a problem, since they tend to have an overreliance on recipes, playbooks, and scripted responses; this is the best way to head towards mediocrity. It removes critical thinking from the picture and leads to “cargo cult” computing at the institutional level. Junior administrators aren't moving up to the next level. Sysadmin as a profession is past the profitable cool initial phase and into a commodity job: it's not new and exciting; and being bored is one of the key aspects. Furthermore, it's not just about the technology, but also about the people (soft) skills: communication and collaboration are tricky and messy but still essential.

It was noted that as a profession we've tried to move away from the sysadmin-as-hero model. Our services are taken for granted and we're only noticed when things go wrong. This seems to be something of a compliment: train engineers used to be badasses because they were what sat between passengers and death, and computing around the year 2000 was like that. That's no longer true; where are the engineers now? (“Rebooting the train” was one wag's response.) Some believe that as individuals we have more power now, but he believes the reason is because what we do can affect so much more of the business than it used to: IT is more fundamental to the business. Siloing is a characteristic of big organizations.

To get very big you have to shove people into pigeonholes. Others believe that, in part because of siloing and regulatory requirements, we have less power as individuals, since the power is distributed across multiple groups and never the twain shall meet.

Technology is constantly changing, so the challenges we face today are different from those we faced five years ago. As a result we recommend hiring for critical thinking skills. Sysadmins used to be the gatekeepers to technology, but so much is self-service at the end users', that's no longer true. We provide a service that our users consume.

We ended the workshop with a quick poll about what's new on our plates in the coming year. Answers included automating production server builds; dealing with the latest buzzwords; diagnosing cloud bits; handling new corporate overlords; improving both people and project management skills; insourcing previously outsourced services such as email, networking, printing, and telecommunications; managing attrition (a 35% retirement rate in one group alone); moving away from local accounts and allowing the central organization to manage them; outsourcing level-1 help desks; simplifying and unifying the environment; training coworkers; and writing software to manage tens to thousands of applications.

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 reports of sessions at USENIX conferences.

Access to *login*: online from October 1997 to this month:

www.usenix.org/publications/login/

Access to videos from USENIX events in the first six months after the event:

www.usenix.org/publications/multimedia/

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 University*
margo@usenix.org

SECRETARY

Alva Couch, *Tufts University*
alva@usenix.org

TREASURER

Brian Noble, *University of Michigan*
noble@usenix.org

DIRECTORS

John Arrasjid, *VMware*
johna@usenix.org

David Blank-Edelman,
Northeastern University
dnb@usenix.org

Matt Blaze, *University of Pennsylvania*
matt@usenix.org

Niels Provos, *Google*
niels@usenix.org

EXECUTIVE DIRECTOR

Ellie Young
ellie@usenix.org