conference reports

THANKS TO OUR SUMMARIZERS

Saurabh Arora
Mukarram Bin Tariq
Leah Cardaci
Marc Chiarini
Rik Farrow
Nathaniel Husted
Kevin James
Ski Kacoroski
Kimberly McGuire
Will Nowak
Shaya Potter
Chris St. Pierre
Josh Simon
Gautam Singaraju
Anu Singh
Tung Tran

LISA '07: 21st Large Installation System Administration Conference

Dallas, TX
November 11–16, 2007

KEYNOTE ADDRESS

■ *Autonomic Administration: HAL 9000 Meets Gene Roddenberry*

John Strassner, Motorola Fellow and Vice President, Autonomic Networking and Communications, Motorola Research Labs

Summarized by Rik Farrow

John Strassner gave a keynote that was, strangely, considered to contain too much math for most of the audience. John began by demonstrating his motivation for coming up with a system that can function when there are seven different groups controlling over 60 sets of services, all—theoretically, at least—striving to satisfy the same business goals. Part of the problem with this picture (see his slide 4 diagram on the LISA '07 Web site), is that it is much too complicated for mere mortals to understand how the different groups can work together. The other issue is that the data within each group is not compatible—that is, each group is a vertical stovepipe, with systems not designed or originally intended to be shared among groups.

Even the meanings of goals, such as Service Level Agreement (SLA), are different among the various groups. At the management level, an SLA specifies the point where lowered performance means a loss of income, whereas at the network administration level, the SLA specifies the percentage of bandwidth to be allotted to each customer. The end result is that there is no single policy that works across all levels, from management all the way down to specific devices such as routers.

John's view of autonomics means that system administrators will be freed from lower-level, repetitive tasks and allowed to manage systems at a higher level. The sysadmin will not be removed from management, but from knowing how to find and tweak various configurations files. The change to the use of autonomics will be gradual, with people and autonomic systems working in partnership.

John's group, working within Motorola, has already produced working tools for managing telecommunication networks. This set of tools is designed to sense changes in a system and its environment, analyze these changes to protect business goals, and plan and execute reconfiguration. As all of this occurs, the system learns by observing the effects of reconfiguration, as well as through people providing positive reinforcement of behaviors that work. So this system encompasses machine learning as well as autonomics. And this is the point where John may have lost some of his audience, as slide 47 contained

two equations (just two!), leading many people to later comment there was "too much math."

John summed up by quoting Einstein: Everything should be as simple as possible, but not simpler. Æleen Frisch then led off the Q&A by pointing out that she liked slide 40 (comparing goals from five different levels) as a concrete example. John responded that there are parallel efforts going on in his labs, and although most work gets down using CLI, all monitoring is done using SNMP—and there is no mapping between the two. He doesn't expect to see Cisco and Juniper standardize on a global lingua franca, but he said that we do need to standardize the higher-level language used to describe configuration goals (see Alva Couch's article about this elsewhere in this issue). Mark Burgess then asked how autonomics can help simplify the organizational diagram (with the seven groups) that somewhat resembles a Borg cube. John pointed out the stovepipe nature of the cube, where different groups of admins really don't talk to each other. Autonomics is about building abstractions, starting at the business details and going down to CLI.

Alva Couch pointed out that John had missed the self-protection ontology, in that sysadmins need to be able to defend themselves, that is, not be blamed for mistakes made by autonomics. John agreed, mentioning that his research system includes safety policies that prevent the autonomic system from acting before a human has reviewed the logs and potential changes. Andrew Hume asked what happens when the autonomic system has conflicting policies, as seen in the HAL 9000 killing off astronauts. John pointed out that the Policy Manager involves using many tools designed to prevent this type of conflicting policy from being created and that the learning loops are also supposed to prevent this type of thing blowing up on you. Another person wondered how new sysadmins could be taught if the autonomic system has relieved the need to perform mundane tasks. John responded that the tools they are developing will help, but that they will not solve every problem.

SECURITY VIA FIREWALLS

Summarized by Saurabh Arora (arora@kth.se)

■ *PolicyVis: Firewall Security Policy Visualization and Inspection*

Tung Tran, University of Waterloo; Ehab Al-Shaer, University of Waterloo and DePaul University; Raouf Boutaba, University of Waterloo, Canada

Tung Tran presented PolicyVis, a tool to help manage complex policies using visualization of firewall rules and policies. He started by giving background on firewall policy management and then provided motivation for doing things a better way to help manage the complexities involved. Then he gave an overview of the PolicyVis tool, which he is developing with his professor Ehab-Al-Shaer at the University of Waterloo. PolicyVis is more than just a visual aid for policy management. It uses rectangles, colors, symbols, and notations to visualize segments and rules and supersets of investigated scope. It also supports compressing and zooming. Tung then used case studies to explain PolicyVis. These case studies included scenarios for investigating firewall policy for accepted traffic by an administrator, visualizing rule anomalies, and visualizing distributed policy configuration. He finished with an overview of the complex tasks involved in managing firewall policies, its misconfiguration, and vulnerabilities.

The PolicyVis Web site is http://www.cs.uwaterloo.ca/~t3tran/policyVis.

■ *Inferring Higher Level Policies from Firewall Rules*

Alok Tongaonkar, Niranjan Inamdar, and R. Sekar, Stony Brook University

Alok Tongaonkar took the stage with interesting research on firewall management. He gave a problem statement of the usage of numerous low-level filtering rules which are configured using vendor-specific tools that either generate low-level firewall rules from a given security policy or find anomalies in the rules. Then he proposed a technique that aims to infer the high-level security policy from the low-level representation. The approach involves generation of flattened rules using packet classification automata (PCA).

■ *Assisted Firewall Policy Repair Using Examples and History*

Robert Marmorstein and Phil Kearns, College of William & Mary

Robert Marmorstein began by explaining the difficulties involved in firewall repair and explained how policies are dynamic, long, and complex. Then he mentioned error detection using a passive query tool. He stressed that there is no way to automate error correction; we can only give partial specification to the tool. His technique is to use Multiway Decision Diagrams (MDD) and perform logical queries against a decision diagram model. Using the query logic, the system administrator can detect errors in the policy and gain a deeper understanding of the behavior of the firewall. The technique is extremely efficient and can process policies with thousands of rules in just a few seconds. Although queries are a significant improvement over manual inspection of the policy for error detection, they provide only limited assistance in repairing a broken policy. He gave an example of this technique on a representative packet, illustrating that the firewall complies with or (more importantly) deviates from its expected behavior.

The project is hosted on sourceforge and researchers are invited to join it: http://itval.sourceforge.net.

INVITED TALK

■ *The Biggest Game of Clue® You Have Ever Played*

Don Scelza, Director, CDS Outdoor School, Inc.

Summarized by Nathaniel Husted (nhusted@iupui.edu)

Don started his talk by stating his objectives for the session. The session was not to teach the attendees about how to beat their kids at Clue, nor was it really about person search and search management. It was aimed more at

those who were responsible for systems and were scared to death about what to do with a big one. It was about how to handle very large-scale problems. He provided examples of these large-scale problems by mentioning some of the search incidents he was involved from 2004 to 2007. Two incidents included autistic males lost in the wilderness, another included a lost woman in stormy conditions, yet another included a lost woman with a history of strokes and brain damage, and there were multiple incidents that involved females being abducted. He also mentions an IT-specific event after the World Trade Center incident in 2001 and a hacking incident in 2004.

Don then outlined the attributes of large-scale problems and their solutions. Many of the problems are time-critical. They may also involve loss of human life or of property. Some may even be criminal in nature. The solutions to these large-scale problems will generally involve numerous people. They might also involve numerous organizations and even law enforcement. Before you can solve any of these problems, you should have a plan. Even if the problem is not covered in the planning, the sheer fact that a plan was created helps you solve the problem. Don provided an example of this with a story about Captain Alfred Haines, a pilot. In 1989 his DC10 lost its hydraulic controls. Although this loss was not covered in the plans, the plans allowed him to cross off what wasn't the problem and decide how to try and land the plane safely without hydraulics.

According to Don, the three best things to know during your planning are your history, your theory, and your –subject. In the realm of lost-person search, this involves knowing what type of events have taken place in a specific area and the characteristics those events have in common. It's also good to note whether there is a common solution when similar events have taken place. Also, look at how previous problems were solved. Finally, make sure to look up and know any theories in your field that could help find the solution. You should also know your subject. In the case of lost-person search, there is a set of behaviors lost persons are most likely to exhibit.

Don then described the theory used in the lost-person search field, entailing concepts such as Probability of Area (POA), Probability of Detection (POD), and Probability of Success (POS). The POD is the probability of the searcher detecting an object if it was in a specific area. The POA is the probability that the subject is in a specific area. The POS is equal to the POA multiplied by the POD and is the probability that if the subject is in a specific area, the subject will be detected.

Don stressed that you should know your resources when solving a problem. You should know what certifications your resources have and whether they will help or hinder the search. Resources also have a cost. Finally, you need to be aware of how to get your resources if they are not currently available and how long it will take to receive those resources. In the case of lost-person search, there are ground resources, dog resources, and aircraft resources.

Don also had advice for what to do after a large-scale problem has been solved. He suggested that you review what actions were taken during the situation as well as what went well and what went poorly. The review session should also cover what needs to be changed in the preplanning stage. The review group should also decide what data needs to be cycled into history and statistics. If the situation could have been prevented, the review group should make note of that as well. However, Don warned that these review sessions can easily turn into finger-pointing sessions, so they must be implemented carefully. An example of such a review was the Hug a Tree program. This program was developed after a boy was lost in 1981 for four days. On the fourth day his body was found two miles from the campsite. The problem was that he kept moving around. A review of this situation led to the Hug a Tree program, in which young children were taught to stand still by hugging a tree.

Don ended by urging everyone to go and enact a plan when they returned to the office by posing questions such as: What history do you need to find out? What team will you put together to help you? What do you need to do when you get to the office? What preplanning and resources need to be on hand? Finally and probably the most important, Where will you get coffee?

In the question and answer period, Don was asked how an ICS would be scaled down to an organization with few individuals. Don replied that there were times when only two people were in the command structure during an incident he was involved in. One of the benefits of a good ICS plan is that the structure can be grown as time proceeds. Don said that this sort of growth was one of the benefits of preplanning.

In response to how much IT was used in search and rescue and whether there was a contingency plan, Don replied that IT is heavily used in search and rescue. People in Ops use it to print maps and people in Plans use it for spreadsheets. He also said that many things are still done by hand and when computers malfunction or something stops working, paper forms provide the needed backup.

The next questioner asked whether Don needed volunteers to help design and implement a computerized system for search and rescue. Don answered with a resounding yes and suggested that anyone who wanted to help should contact him. His email is dscelza@cdsoutdoor.com.

The next question dealt with morale and energy issues during extended searches. Don mentioned that as time progressed during a long search, he brought in counselors to sit down and talk with the individuals helping with the search. The counselors wore brown vests and acted incog-

nito. One of the keys to maintaining morale is to keep the team briefed on the current status. The commander is the driving person who keeps everyone motivated. The key, Don said, is communication.

The final questioner asked how one might work around the fact that best practice can be a competitive advantage in the private sector. Don acknowledged that this is a problem and that corporate citizens need to figure out how to sanitize their plans so that they can provide the information to others. Don also mentioned that talking to people at conferences such as LISA is one of the best ways to share information while staying under the radar.

INVITED TALK

- ***Deploying Nagios in a Large Enterprise Environment***

  Carson Gaspar, Goldman Sachs

  Summarized by Josh Simon (jss@clock.org)

In his invited talk "Deploying Nagios in a Large Enterprise Environment," also known as "If You Strap Enough Rockets to a Brick You Can Make It Fly," Carson Gaspar discussed how a project went from skunk-works to production and how monitoring was explicitly delayed until after an incident. Their Nagios (version 1.x) installation had several initial problems:

- Performance: By default, Nagios (pre 3.x) performs active checks and can't exceed about three checks per second and did a fork()/exec() for every statistical sample. Also, the Web UI for large or complex configurations takes a long time to display (an issue fixed in 2.x).
- Configuration: Configuration files are verbose, even with templates. It's too easy to make typos in the configuration files. Keeping up with a high churn rate in monitored servers was very expensive.
- Availability: There were hardware and software failures, building power-downs, patches and upgrades, and issues of who monitors the monitoring system when it's down.
- Integration and automation: Alarms need to integrate with the existing alerting and escalation systems, and they need to be suppressed in certain situations (e.g., when a building is intentionally powered down). Provisioning needed to be automatic and integrated with the existing provisioning system.

They solved or worked around these problems by switching from active to passive checks (which gets them from 3 to 1800 possible checks per second), splitting the configuration to allow multiple instances of Nagios to run on the same server, deploying highly available Nagios servers (to reduce any single points of failure), and generating the configuration files from the canonical data sources (for example, so any new server gets automatically monitored). They also created a custom notification back end to integrate with their Netcool infrastructure and to intelligently suppress alarms (such as during known maintenance windows or during scheduled building-wide power-downs).

The monitoring system design criteria specified that it had to be lightweight, with easy to write and easy to deploy additional agents, avoid using the expensive fork()/exec() calls as much as possible, support callbacks to avoid blocking, support proxy agents to monitor other devices (such as those where the Nagios agent can't run, such as NetApps), and evaluate all thresholds locally and batch the server updates.

The clients evolved over time; some added features included multiple agent instances, agent instance-to-server mapping, auto reloading of configuration and modules on update, automatically reexecuting the Nagios agent on update, collecting statistics instead of just alarms, and performing SASL authentication among components. The servers evolved as well, with split-off instances based on administrative domain (such as production application groups versus developers), high availability, SASL authentication and authorization, and service dependencies.

The project initially involved a single project with fewer than 200 hosts but was eventually scaled up to large sections of the environment. Documentation and internal consultancy are critical for user acceptance, as is the architecture for the eventual adoption in production for the enterprise. For example, one HP DL385G1 (dual 2.6 GHz Opteron with 4 GB RAM) is running 11 instances with 27,000+ services and 6,600+ hosts, and it's using no more than 10% CPU and 500 MB RAM.

- ***Application Buffer-Cache Management for Performance: Running the World's Largest MRTG***

  David Plonka, Archit Gupta, and Dale Carder, University of Wisconsin—Madison

  Awarded Best Paper!

No summary available.

INVITED TALK

- ***Scaling Production Repairs and QA in a Live Environment (or How to Keep Up Without Breaking the World!)***

  Shane Knapp, Google Inc.

- ***Hardware Ops Release Engineering (or How I Learned to Stop Worrying and Love the Red Tape)***

  Avleen Vig, Google Inc.

  Summarized by Leah Cardaci (lcardaci@cs.iupui.edu)

Shane Knapp and Avleen Vig both related their experiences with dealing with scaling issues for Google's Hardware Operations (HWOps) group. Shane began by briefly relating his history in Google, from starting out in a tech center in 2003 to his current work in technical project management.

He then went on to describe the changes in the nature of HWOps from 1999 to the present. Until 2003, HWOps

had few machines to deal with and was able to use manual processes, physical records, and noncentralized data storage. However, the group saw growth in many areas including machines, technical information to track, and employees to coordinate. The group adopted automation for key processes such as installation, centralized their data storage, and are currently developing the next series of tools.

Shane described the current workflow of machine repairs. This high-level overview followed the process from the time the machine is assigned for repair to the time it is released. He then went on to cover how HWOps was able to scale its services to deal with the enormous increase in machines and employees. One additional challenge to this process was the fact that the changes made have to be made in a live environment, so releases had to be well planned.

The initial improvements were made by looking at the key areas that had problems or were slowing down the overall process. In addition a choice was made to develop and follow the process at a high level. This level of focus allows the individual sites to follow and develop their own process at the floor level, which is important given the diversity of the various sites involved in the overall hardware repair process. There has also been a shift from the group being a black box to the rest of the company to now using in-house technologies.

Being involved in the development of HWOps has provided several insights into how to deal with the challenge of growth and deployment in a live environment. It is important to adopt standard languages and coding styles in order to allow projects to be passed on and maintained. Although it is good to lock down the key parts of the process to allow streamlining, it is also essential to build in flexibility. Planning is crucial and the process should be as visible as possible. One of the hardest lessons learned was that sometimes you have to use the solution that is available now, even if it is not the best solution. The technologies used should be chosen carefully. For example, Python is a better choice for their purposes than Perl, because it is easier to code consistently and is more readable, allowing for easier maintenance. It is important to centralize data and use a consistent scheme so that new employees can easily understand the meaning of the data. Automate as much as possible, but workflow must be understood before automation tools can be developed. Statistical analysis can help to identify areas of the process requiring additional work.

The biggest lesson learned was simply to be careful when making changes. The consequences of any change must be fully understood. Everyone affected by the changes needs to be informed that they will take place. In case something does go wrong, it is important to have a rollback plan to restore normal operation. Be thoughtful when granting rights.

Avleen Vig went on to cover his experiences working on release engineering for HWOps. Avleen has been with Google since 2005 and worked in HWOps to develop in-group tools and release engineering processes. At first, there was no release engineering in HWOps. However, with the extreme growth seen, it became necessary to adopt a formal release process.

He went on to describe the current state of release engineering at HWOps. Before a release can happen, there must be a plan for deployment, a plan for rolling it back, testing, and notes describing the changes for the end users. Each release is categorized into one of three categories: critical, important, all the rest. These categories dictate release requirements such as minimum warning time.

The timing of a release is crucial. Releasing during weekends, holidays, or other times when staffing will be light should be avoided. Notify all those affected when a release has been successfully completed as well as when something goes wrong.

A key lesson learned is that it is important not to get mired in the red tape and to allow for flexibility. For example, it is better for a crucial fix to go out on a Friday instead of the following Monday even if that goes against the practice of not deploying on the weekend.

After the talk, the group was asked whether a change control board was used for their change control review. The process had just changed to include the involvement of a formal change control board.

Avleen was asked about the burn-in hardware testing process. He replied that this involved stress testing of the hard drive, RAM, floating point unit, and other areas.

When asked about the biggest differences made in streamlining the process, the presenters replied that looking at the life of repairs for machines helped. They were able to identify machines that continually failed and replace them.

INVITED TALK

A Service-Oriented Data Grid: Beyond Storage Virtualization

Bruce Moxon, Senior Director of Strategic Technology and Grid Guru, Network Appliance, Inc.

Summarized by Will Nowak (wan@ccs.neu.edu)

The term "Storage Virtualization" is now used to describe any level of storage abstraction. Bruce Moxon helped to shepherd the audience through the fog and understand various current and future storage technologies. Bruce first took a look at conventional storage and how that works in the enterprise. Typical situations, such as overloading a single cluster node while the other nodes remain underutilized, were tackled.

By using NetApp products as a talking point, some generic solutions to common problems were illustrated. Technologies such as vFiler allow storage administrators to segregate service-specific storage into its own virtual file server instances. This abstraction enables load sharing, or easy migration in the event of an overloaded server.

Other types of virtualization, such as data virtualization, were also touched upon. Bruce gave an example of a thin client test lab at a NetApp facility in RTP. This test lab utilized blade servers and a series of NetApp filers to simulate a large client load on the filer hardware. Each blade could boot from the network, using a virtualized file system image. This allowed the total lab to use only the base file system storage cost, plus a small storage cost for client personalization. This type of virtualization provides a tremendous savings in raw storage allocation.

In the storage futures discussion, Bruce made several comparisons of Old World technology, such as the typical NFS file server, to new technologies such as the Google File System or its open source equivalent, the Hadoop File System. Bruce suggested that these distributed file systems, which take advantage of low-cost generic hardware, would continue to gain traction where they are applicable. Other interesting developments, such as storage appliance inserts, in-line encryption, and storage direction engines, were also touched upon.

The consensus of the session was to bring home the potential advantages of looking at a virtualized storage infrastructure. Abstract out your storage requirements to better serve your customers.

VIRTUALIZATION

Summarized by Shaya Potter

Stork: Package Management for Distributed VM Environments

Justin Cappos, Scott Baker, Jeremy Plichta, Duy Nyugen, Jason Hardies, Matt Borgard, Jeffry Johnston, and John H. Hartman, University of Arizona

Scott Baker presented a new approach to package management for administering large numbers of virtual machines. Because each virtual machine is an independent entity, this provides good isolation. However, it also results in an inefficient use of resources, owing to the inability to share file system state; namely, each VM has its own disk and each disk will be cached separately by the underlying physical machine, causing increased contention for both memory and disk resources.

To solve this problem, they introduce the Stork package management system, which enables secure and efficient inter-VM sharing of file system content. Stork has two characteristics. First, its package manager, similar to tools such as apt and yum, is combined with a publish-subscribe mechanism that enables VMs managed by Stork to be automatically notified of package updates. Second, it enables packages to be stored in the "stork nest" and then shared with any VM on the same host.

When a package is installed into a system with a stork nest, it is first installed in the local machine's file system as well as into the stork nest. Every file within the stork nest is marked with the NOCHANGE/Immutable bit, preventing it from being modified. The nest's version is then shared with the VM by overwriting all of the package's files, excluding files marked as configuration files, with hard links to the version of the file in the nest. As many VMs on the host can make use of the same packages, they will each use only the version that is contained within the nest, enabling efficient sharing of files in a secure manner. Stork is currently used on PlanetLab machines, and it has been shown to offer significant disk space savings for most packages. One notable exception of this is the j2re package, where a large amount of data was unpacked during the packages post-install scripts. If the files were to be repackaged in the already extracted state, this issue would be avoided. [Editor's note: There is a much more detailed article about Stork in this issue.]

Decision Support for Virtual Machine Re-Provisioning in Production Environments

Kyrre Begnum and Matthew Disney, Oslo University College, Norway; Æleen Frisch, Exponential Consulting; Ingard Mevåg, Oslo University College

Kyrre Begnum presented an approach to managing large numbers of virtual machines, involving notably on what physical machine they should be provisioned. This is a hard problem because system administrators need to optimize for physical machine redundancy to enable physical servers to be removed for maintenance, without compromising the ability to use virtual machines as well as remove bottlenecks resulting from resource conflicts.

To enable system administrators to solve this problem, they introduce three metrics to help determine where a virtual machine should be deployed. The first metric focuses on the amount of server redundancy. If we were to remove a physical machine from a clustered environment, could we redeploy the virtual machines contained within it to the other machines within the cluster? This is notably a problem with Xen, as it does not allow the host to overprovision the memory resource. To quantify this, they introduce the notation R/S to express the redundancy level of a cluster, where R is the number of servers currently in use within the cluster and S is the number of servers that can be removed from the cluster.

The last two metrics deal with resource conflicts. Many resources that a VM will use are shared, one important one being disk IO. If multiple VMs on a single physical ma-

chine make heavy use of that resource, their overall performance will suffer owing to contention in use of that resource. To determine where a virtual machine should be deployed, we need to know what conflicts exist between virtual machines in their use of shared resources. The Resource Conflict Matrix enables administrators to measure the level of conflict between virtual machines deployed on their servers. The final metric enables them to measure the value of conflict on a particular server with the focus on minimizing the level of conflict.

■ *OS Circular: Internet Client for Reference*

Kuniyasu Suzaki, Toshiki Yagi, Kengo Iijima, and Nguyen Anh Quynh, National Institute of Advanced Industrial Science and Technology, Japan

Kuniyasu Suzaki presented an approach for booting virtual machines over the Internet. The OS Circular framework enables a virtual machine to fetch a disk image over the Internet using HTTP and demand-page the disk blocks that are needed *as* they are needed. These blocks will then be cached locally so that they do not have to be constantly refetched.

To enable this demand-paging model, OS Circular divides a file system image into 256-KB compressed blocks, where each block becomes its own file, named by the SHA1 hash of its data. This enables the VM to verify that the data was fetched correctly. Each file system has a mapping file that maps block numbers to the correct SHA1 named file; a file system is mounted by making use of the mapping file and demand-paging and caching the blocks as needed. A file system can be updated by creating new SHA1 named files for the updated blocks and updating the mapping appropriately.

One problem with demand-paging a file system is that network latency can have a severe impact on the file system, especially on an initial boot of it, when no data is cached locally. To optimize latency, they leverage ext2optimizer to profile the file system and place files needed by the boot processes to be placed at the beginning of the file system. By removing fragmentation normally existing in a file system and leveraging read-ahead techniques, one can minimize the overhead from the network latency.

■ *Secure Isolation of Untrusted Legacy Applications*

Shaya Potter, Jason Nieh, and Matt Selsky, Columbia University

Shaya Potter presented an approach to contain independent services and their individual application components. Software services need to be contained because software is buggy and those bugs can result in security holes, providing an attacker with access to the system. However, services are made up of many interdependent entities, so containing those entities appropriately can be difficult.

To resolve these issues, Potter et al. introduce two abstractions, Pods and Peas. Pods provide a lightweight virtual environment that mirrors the underlying operating system environment. Processes within a Pod are isolated from the underlying system, and as such Pods are able to isolate an entire service. Because a Pod is hosted on a regular machine, it does not need many of the resources regular machines need (e.g., what's needed for booting), enabling it to contain just the resources needed for the entire service.

The second abstraction, the Pea, enables a simple access control mechanism on the resources made available to the Pod. The overriding principle is that just because a process is within the Pod does not mean it needs access to the resources the Pod makes available. Peas are notable, when compared to existing containment systems such as Janus and Systrace, for performing file system security in the correct location, namely the file system itself, and therefore they do not suffer from common "time of check, time of use" race conditions. Peas also implement a simple-to-understand configuration language that leverages the skills system administrators and users already have to perform as part of their daily tasks. Finally, access control rule creation can be difficult because the knowledge necessary to build rules is divided between the developers, who know the minimum needs of the application, and the administrator, who defines local security policy, so Shaya Potter demonstrated a rule composition mechanism that enables a developer to provide a minimal rule set that defines the minimal needs of the applications while enabling the administrator to build upon that and to define what local policy one wants to apply to the application.

INVITED TALK

■ *Who's the Boss? Autonomics and New-Fangled Security Gizmos with Minds of Their Own*

Glenn Fink, Pacific Northwest National Laboratory
Summarized by Marc Chiarini (marc.chiarini@tufts.edu)

In this talk, Glenn Fink tells us that autonomic computing (AC) is coming, albeit much more slowly than we think. He also suggests that our jobs are not in danger in the near future, though a sysadmin's duties will change significantly as the world transitions to AC technologies. To put things in perspective, Fink gives us a "personal guesstimate" of how far along we are on the four big cornerstones of autonomic computing as defined by IBM: self-configuration at 60 percent (with tools such as Cfengine, Puppet, and BCFG2 aiding this process); self-healing at 25 percent (an admittedly generous estimate, because most of the academic work on this has been in the security arena); self-optimization at 10 percent (another generous estimate, as we only know how to do this in very specific domains); and finally self-protection at 40 percent (where there has been a lot of good research into detecting and responding to attacks and general failures). Of course, these progress markers do not average out to 33 percent for the whole of

AC, because we have no clear way as yet of integrating the various systems that implement these processes.

Fink presents autonomic computing as a direction (or continuum) rather than a goal. This is to say that it will always be difficult to draw a bright line between AC and non-AC systems; we will be able to watch the changeover happening, but we won't be able to "see it happen." Like many other evolutionary processes, autonomic computing is being driven by the necessity to meet demand for services. IT infrastructure growth has been exponential in recent years. Combined with a software crisis (over budget, beyond schedule, buggy, and difficult to maintain), a hardware crisis (in which volume overtakes reliability), a tech education crisis (a lack of qualified high-tech workers), and the (relatively) prohibitive costs of IT personnel, this growth rate is unsustainable without automation or excessive outsourcing. Unless we want nearly everyone in IT to lose their jobs, we need to think hard about how to build autonomic systems.

When we start deciding what AC should look like, we quickly fall into a contest between the purist and the pragmatist. The purist believes that maintenance is the dominant long-term cost, that system populations should be as homogeneous as possible, that policy should be centrally defined, and that admins should be significantly constrained to avoid conflict with autonomic processes. By contrast, the pragmatist thinks that downtime is the dominant cost and decentralized quick fixes (by almost any means) in a highly heterogeneous environment are the way to go. Fink suggests something in the middle: Ensure that pragmatic fixes feed back into an established, inspected, and trusted library of practices that is open-sourced. All autonomic computing will be done under human supervision, with the added goals of communicating why decisions were made and how those decisions relate to other autonomic systems.

Fink spent the last half of the talk enumerating both a wish list and a fear list concerning autonomic computing. In his conversations with colleagues and IT professionals, he discovered three characteristics that will be most important: AC systems should act like a junior sysadmin, investigating and reporting with lots of little open-ended tasks; they should be able to robustly handle real-world situations with little or no supervision; and they should be able to communicate like a human, providing sufficient detail in a natural language context. Of the prodigious list of fears, the most important were probably issues of trust, process verification, and delegation. How do I know the system is doing what it should? Can I trust the system to verify existing agreements or negotiate new agreements with other systems?

In the end, Fink believes that our jobs are in danger, not from autonomics, but from outsourcing. Autonomics will be able to take care only of well-defined tasks and problems, and someone will always be needed to verify autonomic behavior and adherence to business objectives. The ways in which AC will change the profession are manifold: Computers will be trusted with more kinds of work, resulting in fewer tedious tasks; sysadmins will have more time to help users (hone those social skills now!); there will be natural dividing lines among AC specialists (as witnessed in the medical fields), and ultimately it is the specialists (e.g., DB and storage) who will be impacted more than the nuts-and-bolts system and network administrators. Finally, much more ethnographic study of both IT professionals and users will be necessary before AC is ready for prime time.

INVITED TALK

No Terabyte Left Behind

Andrew Hume, AT&T Labs—Research
Summarized by Josh Simon (jss@clock.org)

Andrew Hume discussed the disk dilemma: Space is cheap, so users want, get, and use more of it. However, this leads to all sorts of interesting problems, such as how to partition and how to back up the disk (especially when you get toward terabytes on the desktop). Traditional tools (such as dump) take 2.5 days to back up 250 GB. Making the space available from servers can be problematic (given local or networked file systems and the associated problems with network bandwidth). We've talked about these issues before, but there are still no good solutions.

Let's take a hypothetical example of recording a TiVO-like service without any programming wrappers. Recording everything all the time for both standard and high-definition programming leads to about 1.7 petabytes per year of data, even assuming no new channels get added. This is too big for the desktop, so we'll need to use space in the machine room: a 2U or 3U generic RAID unit at 2–4 TB/U costs up to $1,500/TB, and you'd need 133 of them per year. This uses 16 TB per square foot and requires 27 feet of aisle space per year with modest power and cooling. But that's a lot of money and space. We can possibly be clever by looking at the access patterns; for example, we can move the older and less-accessed shows off to tape, or keep only the first 5 minutes of the show on disk and the rest on tape, and thanks to a tape library (e.g., an LTO-4 with 800 GB/tape and 120 MB/s sustained write at 60-s access and a 2.5-PB library costs $172/TB and uses 41 TB per square foot, and expansion units are $7/TB and 79 TB/square foot) we can still provide every TV show on demand with no user-visible delays. Sounds good, right?

Wrong. It gets worse when you realize the fallibility of media. Ignoring the issues with tape (such as oxide decay, hardware becoming obsolete, and so on), we've got problems with disks.

Here's the reality about using disks, networks, and tapes: Things go bad, trust nothing, and assume everything is out

to get you. You don't always get back what you put out. Compute a checksum for the file every time you touch it, even when it's read-only. Yes, it's paranoid, but it's necessary if you really care about data integrity, especially with regard to disk and tape. Andrew is seeing a failure rate of about one uncorrectable and undetected error every 10 terabyte-years, even in untouched, static files.

As disk use grows, everyone will see this problem increasing over time. The issue of uncorrectable and undetected errors is real and needs attention. We need a way to address this problem.

PLENARY SESSION

The LHC Computing Challenge

Tony Cass, CERN

Summarized by Leah Cardaci (lcardaci@cs.iupui.edu)

Tony Cass discussed the challenges associated with working toward the debut of CERN's Large Hadron Collider (LHC), which will deploy next year. Cass began with an introduction to CERN and LHC. CERN's goal is to "push back the frontiers of knowledge" by investigating important scientific questions. (For example, one major question is why certain elements are heavier than others; one theory is the existence of the Hick's field.) This often involves the deployment of new technologies to support the research performed. CERN's goals are to unite people from different countries and cultures and help train future scientists.

Cass gave a brief overview of four LHC experiments: ATLAS, CMS, ALICE, and LHCb. Each of these experiments will produce about 40 million events per second, which will be analyzed and reduced to a few hundred good events per second. This means the four experiments will require around 15 petabytes of storage per year. The three steps of data handling are reconstruction, analysis, and simulation. CERN is responsible for reconstruction and data retention; other locations deal with analysis and simulation. Overall, enormous computing resources are required. The challenges involved in running these experiments are having sufficient computing capacity, managing the high number of machines required, tracking and distributing the data, and understanding the state of the resulting highly complex system.

A three-tiered system is used for data handling. Tier 0 is the accelerator center, responsible for recording and processing the data from the accelerator and long-term storage. Tier 1 centers are responsible for distributing the data to researchers, as well as for analysis of the data. Tier 2 centers are involved in simulation and end-user analysis. Grid technology was adopted to provide the high amounts of computing resources needed. This involves three grid infrastructures, EGEE, OSG, and NorduGrid. The project had to meet certain levels of interoperability for submission of jobs through the system and administration of the system. Reliability will be a continuing challenge once the experiment is launched and the project has increasing reliability goals to meet.

Management of machines is provided by ELFms Vision, a custom toolkit developed by CERN and others to provide a system that would meet all of the project's needs. Quattor provides scalable installation and configuration management. Lemon provides monitoring, including looking at information outside of the individual computers, such as UPS status. LEAF, a collection of high-level workflows, automates tracking nodes' physical status as well as their configuration status. Integration with Quattor and Lemon allow for a great deal of automation in the management of nodes. This design has allowed CERN to deal with the great increase in machines added throughout the preparation, and it will continue to scale further. A huge amount of data has to be stored and distributed for this experiment, which poses another challenge. The accelerator produces an average of 700 MB per second of data, and the system will need to be able to support almost twice that amount to allow for recovery. There are three different types of access use cases: sustained transfer to a remote site, rapid transfer of data set to nodes, and long-running analysis access of data on a server. Each type has its own requirements and creates a different footprint on the data servers. No existing system met all needs, so CERN developed CASTOR, the CERN Advanced STOrage system. CASTOR is based on databases, schedules the data distribution to prevent overwhelming the system, and also schedules based on priority. Continuing challenges will be keeping the data lifetime long enough, dealing with the disparity of capacity vs. IO rates, integrating different data systems without interfering with the use of the system, and handling the export of data.

The final challenge is to manage the incredibly complex system developed to support the LHC experiments. This has been aided by the use of a user status view, which shows the current status of all the jobs for a single site, pulling the information from the (possibly many) nodes they are running on. This also involves grid monitoring and a new visualization technique to help managers focus on the critical problems in the system.

Overall, the project involves many challenges related to its size and complexity. So far, many of these challenges have been met, but the real test will begin once the system goes into full operation.

Cass was asked whether the group was ever in a situation where waiting to buy machines would be more cost-effective. He replied that they had seen those situations, but at their scale there was a greater latency because of deployment time, so that had to be take into consideration.

Another question was whether CERN was concerned about malicious attempts to corrupt the data. Cass replied that they didn't think the project was high-profile enough for their data to be a target, but they had considered that their computing resources could be a target.

MISCELLANEOUS TOPICS I

Summarized by Marc Chiarini (marc.chiarini@tufts.edu)

Policy Driven Management of Data Sets

Jim Holl, Kostadis Roussos, and Jim Voll, Network Appliance, Inc.

IT departments frequently ask Network Appliance for a unified software and hardware infrastructure that will provide them with easily managed and well-protected storage and data services. The primary reason for this request is to optimize the use of physical resources and reduce complexity, thereby reducing cost. A typical way to achieve this goal might be to use shared storage arrays that allow multiple disparate disks to be viewed and acted upon as single logical entities. Unfortunately, organizations rarely use a single vendor for their storage infrastructure, and even when they do, there exist incompatibilities among products and service tiers. There are frequently too many individual storage containers because of data growth and replication, making management very difficult and resulting in under- or over-provisioning of both storage and protection. Instead of having a unified physical storage and data management layer, customers tend to engage in two separate disciplines: storage management (e.g., how many and what kinds of disks, controllers, and LUs are needed) and data management (e.g., how resources are used, backup discipline, replication discipline, where to place files, databases). Since data management relies on storage management, large organizations often end up manually translating the former into the latter by way of the help desk. Roussos's team developed software to handle the automatic right-sizing and placement of storage resources.

The unified storage and data management software presented in the paper introduces three abstractions: a resource pool, a data set, and a policy. A resource pool is a fixed amount of physical capacity, performance, and IOPs along with well-defined sets of capabilities, such as deduplication, replication, and redundancy. It allows easier management and optimization across more storage containers. A data set is a collection of data and the replicas that use a single data management policy. Data sets abstract storage containers and locations from the data and reduce the number of objects to manage. A policy describes how a data set should be configured with regard to protection and provisioning. Policies establish clearly defined roles, with storage architects constructing them, data admins selecting which ones are used, and a conformance engine configuring storage according to the selected policy. The conformance engine performs multiple tasks, including monitoring current configurations, alerting administrators to policy violations, and reconfiguring automatically when possible.

Roussos gave a very compelling comparison between a traditional graph of storage infrastructure and a view of the same graph in terms of data sets, which greatly simplifies and clarifies things. The take-away from the presentation was that a unified data and storage management layer vastly reduces the number of entities that must be managed and the number of steps required to perform traditional tasks. Lastly, it gives admins the advantage of conformance monitoring, to continually check that everything is laid out according to plan.

ATLANTIDES: An Architecture for Alert Verification in Network Intrusion Detection Systems

Damiano Bolzoni, University of Twente, The Netherlands; Bruno Crispo, Vrije Universiteit, The Netherlands, and University of Trento, Italy; Sandro Etalle, University of Twente, The Netherlands

For those system administrators who are not quite familiar with the security aspect of our profession, IDSes (or Intrusion Detection Systems) are software systems (sometimes coupled with hardware) that are designed to detect (and sometimes take action against) malicious activities occurring on a host or in a network. ATLANTIDES deals exclusively with attacks in a network. There are two approaches to detection: signature-based approaches, which search network packets for specific predefined and well-known sequences of bytes, and anomaly-based approaches, which gather statistics about the packets "normally" seen on the network and indicate when those statistics stray significantly from the norm. Network IDSes are considered an efficient second-line defense (after firewalls) because they are virtually transparent to the monitored network and generally do a decent job. There are some significant disadvantages to both types of detection, however, that can greatly reduce the cost/benefit ratio: Signatures must be carefully selected for a particular site in order to reduce the number of false alarms that are generated; anomaly-based detection uses a threshold to raise alarms, which must also be tuned. In short, these tasks threaten to overwhelm IT security personnel.

Bolzoni's team proposes a solution that greatly reduces the management workload resulting from required detection tuning and verification of alerts. ATLANTIDES is an anomaly-based network IDS that can be combined with any traditional NIDS to efficiently improve the rate of false positive alarms. Instead of watching incoming network traffic for signatures or anomalies, the system learns over a short time (one to seven days depending on the diversity of outgoing traffic) what "normal" output traffic looks like. Whenever the incoming NIDS would normally raise an alert on suspicious activity, the ATLANTIDES correlation engine determines whether the output traffic seems suspicious as well. If so, an alert is raised that, because of this double-checking, has a high likelihood of being true. If there is a mismatch between what the input NIDS sees and what ATLANTIDES sees, the system can be configured to either discard the alarm as a false positive or, in the case of a potential false negative (incoming traffic looks OK, but outgoing does not), escalate the severity of the alarm.

To determine the efficiency and accuracy of ATLANTIDES, Bolzoni's team ran tests against both a well-known Internet traffic data set (DARPA99 multiprotocol) and a recently captured unfiltered HTTP traffic data set. In the case of the DARPA data, the tests showed a reduction of between 50% and 100% in the false positive alarm rate when compared to use of a single NIDS alone. In the HTTP traffic set, ATLANTIDES also improved the rate by more than 50%. The observed maximum output analysis rate was around 100 MB/s. The team plans to do further testing with more real-world data in the near future, but they are very excited about the results so far.

■ ***PDA: A Tool for Automated Problem Determination***

Hai Huang, Raymond Jennings III, Yaoping Ruan, Ramendra Sahoo, Sambit Sahu, and Anees Shaikh, IBM T.J. Watson Research Center

Ruan presented a system that improves the efficiency with which system administrators can analyze and respond to trouble tickets. The motivation for this research was a lack of robust, tailored, and easy to use tools for problem determination. System administrators (yes, even folks at IBM) tend to troubleshoot in an ad-hoc, time-consuming manner; they build different customized scripts for different platforms and frequently reinvent the wheel; the knowledge they gain is usually stuck in their heads and cannot be easily leveraged.

The PDA approach is threefold: attempt to characterize the nature of real-world problems by analyzing problem tickets and their resolutions; provide a common platform to standardize monitoring and diagnosis tools (also known as probes); and capture problem determination knowledge in expressible rules. The approach collects both high-level system vitals and "drill-down" problem analysis steps. The study utilized about 3.5 million trouble tickets generated over 9 months and analyzed the ticket distribution and time spent resolving tickets across a wide range of products and within the products themselves.

Ruan's team discovered several interesting statistics: 90% of the tickets resulted from trouble within 50 applications, the top two being a mail app (20%) and a VPN app (10%). Within the mail app, 70% of the tickets came from only 11% of its modules. Within the VPN app, 70% of the tickets came from only 8% of its modules. More important than this distribution of trouble tickets across applications was the amount of time it took to resolve OS problem tickets, roughly an order of magnitude longer on average. Taken in combination, application and configuration problems related to problems with a particular OS made up the majority of tickets. Thus, PDA is designed to focus on issues stemming from OS and system software misconfiguration.

PDA implements a thin probe model, in which generic checks are made on managed servers on a scheduled basis. The probes can be built via native commands, scripts, existing tools, or even specialized executables. The probes transmit standardized key/value pairs to a rules engine that checks potentially extensive yes/no decision trees for compliance, asking for more probe information when necessary. If a violation is discovered, the engine executes whatever action was specified in the rule sets, which might entail terminating future probes, sending an alert to a Web interface or email, or taking corrective action. New probes and rules can be authored via a simple Web interface that leverages existing collections of trouble tickets, probes, and rules.

In future work, Ruan's team hopes to address issues with the security of authored probes and also investigate the possibility of making probes and rule sets shareable across different platforms and sites.

INVITED TALK

■ ***Experiences with Scalable Network Operations at Akamai***

Erik Nygren, Chief Systems Architect, Akamai Technologies
Summarized by Shaya Potter (spotter@cs.columbia.edu)

Akamai deploys a large, worldwide-distributed network that provides many services, including HTTP/HTTPS, live and on-demand streaming, and app delivery. Akamai is such an integral part of the Internet that we use it every day without even realizing it.

Akamai distributes its servers all over the world, as those who use the Internet are highly distributed as well. According to Akamai's measurements, one has to be on 1000 separate networks to be close to 90% of Internet users. By distributing servers toward the edges, they gain greater performance and reliability and are able to absorb traffic peaks better, as they avoid congestion points that occur where networks peer. In fact, ISPs want Akamai, as it saves them money because the traffic never leaves their network.

To distribute content to its distributed machines, Akamai deploys its own overlay network to create a highly reliable tunnel among the machines. Today the tunnel includes 28,000 machines in 1,400 locations. As Akamai uses commodity machines and network links, it expects lots of faults, so it has to treat failures as a normal occurrence. The primary way of dealing with this is with large amounts of redundancy. Redundant machines can be easily repurposed, enabling Akamai to handle faults even within a single cluster of machines. For instance, in a single cluster of machines, a "buddy" of a machine that goes down can take over for it by simply grabbing the IP of the failed machine and handling requests that are directed to it. Geographic and network redundancy combined with multipath communication in its overlay network enable Akamai to handle faults within the network links. Finally, the company has fully redundant NOCs distributed around the world, so that no one NOC has functionality that cannot be replaced by another NOC.

To manage all these computing systems, Akamai has implemented a query system that enables efficient real-time monitoring of its systems. It uses a relational database model, in which each machine provides a set of tables that contains information about its current state. Akamai's query systems compose the information provided by the machines into a set of 1400 distinct tables, with table updating occurring in the 1–3-minute range. This enables alerts to be created via regular SQL queries and the management of a large number of machines in a more automatic manner.

INVITED TALK

Ganeti: An Open Source Multi-Node HA Cluster Based on Xen

Guido Trotter, Google

Summarized by Will Nowak (wan@ccs.neu.edu)

Guido Trotter gave an overview of Ganeti, outlining its goals and usage, provided a road map for the future, and made a valiant attempt at a live demo. Ganeti is a open source management layer that rides on top of a vanilla Xen setup, allowing management of multiple nodes in a cluster. Tasks such as provisioning, management, failover, and some disaster recovery are handled by the Ganeti software package. Ganeti's goals are formulated much like other Google technologies. The project aims to increase availability, reduce hardware cost, increase machine flexibility, and add a layer of service transparency. Ganeti was also designed to scale linearly, be hardware agnostic, be broadly targeted, and maintain small, iterative development.

Ganeti leverages Xen currently, but Guido mentioned that in the future they hope to support other virtualization technologies. The toolkit is written in Python, using LVM, DRBD, and MD for storage and Twisted with SSH for RPC. Ganeti is best supported on Debian-based systems, but porting to other Linux distributions should be trivial.

Questions were raised regarding overlap with the Linux HA project. Guido's response was that Ganeti was designed at Google internally to fit a specific need that available software could not fill and that he would be interested in seeing how the two products could better serve each other.

More information on Ganeti can be found at http://code.google.com/p/ganeti/.

MANAGING GRIDS AND CLUSTERS

Summarized by Saurabh Arora (arora@kth.se)

Usher: An Extensible Framework for Managing Clusters of Virtual Machines

Marvin McNett, Diwaker Gupta, Amin Vahdat, and Geoffrey M. Voelker, University of California, San Diego

Marvin explained the motivation of their research, which was to help system administration become more effective and allow sharing among multiple resources efficiently. Their approach is to use virtual clusters (i.e., to deploy multiple VMs on each physical machine). The tool they are developing, called Usher, simplifies VM administration. The best part about Usher is its extensible architecture. Marvin and his team have done extensive work in making Usher extensible, by providing user application APIs and VMM wrappers and using plug-ins to add new functionality to Usher. The available plug-ins as of now are LDAP, IP Manager, and DNS. Usher has been successfully deployed in the following places: the Russian Research Center at the Kurchatov Institute, UCSD CSE System, and research projects such as spamscatter and spaceshare. The Usher Web site is http://usher.ucsd.edu.

When asked whether Usher is available for all virtualization technologies, Marvin replied that it is only available for Xen, but you can easily write a wrapper for vmware, KVM, etc.

Remote Control: Distributed Application Configuration, Management, and Visualization with Plush

Jeannie Albrecht, Williams College; Ryan Braud, Darren Dao, Nikolay Topilski, Christopher Tuttle, Alex C. Snoeren, and Amin Vahdat, University of California, San Diego

Jeannie Albrecht gave an overview of building distributed applications and introduced us to the Develop-Deploy-Debug cycle of a distributed application. Then she focused on challenges involved in this cycle of locating and configuring distributed resources. She also stressed the challenges involved in recovering from failures in a distributed deployment. The goal of her research is to develop abstractions for addressing the challenges of managing distributed applications. She took the specific example of developing a distributed application, say Bytetorrent, for the presentation. She started with different phases of the application and discussed evaluation through management architecture such as PlanetLab. She explained the hurdles involved in each phase of the example application, and here she proposed a distributed application management infrastructure—Plush. She explained the Plush architecture and how it can acquire resources, configure resources, and start and monitor applications. Plush has a beautiful graphical user interface, called Nebula, that is used to describe, run, monitor, and visualize deployed applications. The Plush home page is http://plush.cs.williams.edu.

Everlab: A Production Platform for Research in Network Experimentation and Computation

Elliot Jaffe, Danny Bickson, and Scott Kirkpatrick, Hebrew University of Jerusalem, Israel

Everlab was spawned from the EU-funded research project Evergrow, which was proposed for large-scale network management. Elliot Jaffe began by giving an overview of Evergrow. During that project, they felt the need for a better management system, so they moved toward PlanetLab, which is very tightly secured and offers centralized man-

agement. But the consortium of the EU project was not very supportive in joining PlanetLab, so they came up with Everlab. He mentioned that Everlab is inviting researchers to join and use its underloaded resources (as opposed to the overloaded resources of PlanetLab). Elliot came up with a few noteworthy conclusions about research projects in general: (1) funding is only for research; (2) release, deployment, and management are not research; (3) there is a difference between a flash-in-the-pan system and a computing standard. He then asserted that sound funding should be made available for deployment and management as well.

The Everlab home page is http://www.everlab.org.

INVITED TALK

Using Throttling and Traffic Shaping to Combat Botnet Spam

Ken Simpson, Founder and CEO, MailChannels
Summarized by Leah Cardaci (lcardaci@cs.iupui.edu)

Ken Simpson gave an overview of his approach to fighting spam, which is based on the concept of attacking spam by attacking the economics of spam. He began by relating his personal work history on dealing with spam, from his beginnings with ActiveState to forming a company with other former ActiveState employees.

Simpson went on to provide a history of the spam problem, noting that his was a rough view and anyone was free to correct mistakes during the Q&A session. In 2002, spam had not been a major problem, was not a crime in most areas, and was handled using regular expression filters. In 2003, spam had made mail almost unusable, the CAN-SPAM act was created, and the spammers went underground in response. In 2004 Bill Gates announced that spam would be beaten in two years. Now spam is a fully criminal endeavor, run by organizations such as Russian gangs.

Covered next was the economics of spam. Simpson suggested that the current way of handling spam, filters, will not be able to have an impact on the overall economics of spam. Although current filters are fairly accurate, the ease in increasing the volume of messages means that the spammers can always win the game of averages.

Currently, spam is being sent from compromised computers, which are organized into botnets controlled by a bot herder. The botnets are rented to the spammers by the bot herder, providing a constantly changing set of machines from which to send messages and thus overcome blacklisting. This doesn't mean that blacklisting is not useful; in fact it allows a great deal of spam to be blocked and keeps systems from being overwhelmed with traffic. Also, the use of blacklists now means that a given botnet will quickly lose its ability to spam, and new machines must be compromised constantly to keep up.

Botnet herders are only paid once the final SMTP acceptance message is received, so they will not profit if the mail is blocked by a blacklist or filtered. For this reason, spam software has an extremely short timeout compared to legitimate mail servers, which follow the three minutes recommended in the RFC.

The current state of affairs is that spam filtering is reaching the limit of its possible increase in accuracy, and identifying zombies to simply block traffic from them is very difficult. Simpson suggests a new approach designed to attack spam by removing the profit in it. This approach uses both blacklisting and whitelisting, and then throttles all suspicious traffic to see whether it will reach the very short timeout of the spam software.

Simpson went on to discuss a case study of the deployment of this system. In this case, a pharmaceutical company saw a overnight reduction from 70% of mail being spam to 20% being spam. The system is deployed in software at the edge of the network. One challenge introduced by this system is the fact that the throttling of suspect traffic requires a great increase in the number of concurrent connections the mail servers must handle. The solution was to introduce a system in front of the mail server that handles the throttling and to use real-time SMTP multiplexing to reduce the connections the server has to handle. Looking at the suspicious traffic revealed that 80% of those machines that dropped the connection were running a consumer version of Microsoft Windows.

Simpson was asked whether the spam software won't simply be adjusted to increase the timeout window once this approach was widely accepted. He replied that there is typically a long time before spammers adjust to such measures, and that this would still affect overall profitability owing to the short time before the machine is placed on a blacklist.

Someone pointed out that people did in fact care about spam in 1995. He went on to point out that spammers can react very quickly to changes in spam defense.

Another audience member suggested that the current profit margin was so high that it seems unlikely that serious damage can be done to the profitability of spam.

MISCELLANEOUS TOPICS, II

Summarized by Marc Chiarini (marc.chiarini@tufts.edu)

Master Education Programmes in Network and System Administration

Mark Burgess, Oslo University College; Karst Koymans, Universiteit van Amsterdam

In this talk, Burgess discussed the philosophical and technical difficulties of supporting a traditionally vocational subject within a strong academic framework. One of the biggest controversies involves the question of teaching what is viewed not as a discipline, but rather as a set of technical skills. How do we teach something that most people believe is only gained through experience? Who should teach it, professors or practitioners? What material

should be used? One quickly becomes mired in a plethora of questions for which the promise of a good answer does not even exist. Burgess suggests the following: that the "discipline" needs to be described in a fairly rigorous form that can be handed down for posterity. It cannot be presented as currently practiced, because it changes too fast; we need to find paths to and from other disciplines that will promote an awareness of the subject; we should try to preserve the hands-on, engineering-focused aspect of system administration; it is important to stay in touch with rapid industrial development (something for which other academic disciplines are not well known); and finally and perhaps most importantly, we need to make it well known that the formalization of system and network administration does not belittle those who have learned the subject by other means. We should not think of system administration in universities as replacing everything that people have learned the hard way, but rather as a way to document those efforts and hand down the best parts.

In the second half of the presentation, Burgess gave the audience a more complete description of the nature of the programs at each university, which, although developed separately, are remarkably similar in scope and direction. The Amsterdam University program, which is compressed into one year, speaks volumes about the ability to teach system and network administration as a core academic discipline. Oslo University College offers a two-year program divided into four semesters. The first semester is spent giving students background knowledge with courses such as networking, firewalls, info security, and system administration fundamentals. The second semester teaches students to stand on their own two feet, with a course heavy on lab work, a course on how to read research papers, and a course on ethics. The third semester attempts to make students think critically about what they've learned and adds some specialized courses. The last semester culminates in a thesis that draws on the foundation of the previous coursework.

The subject of university education is very different from self-learning or even targeted training (such as that provided at LISA). Accredited academic courses immerse a student in a common culture, not only granting knowledge about the world but also teaching the processes required for abstraction and the development of generalized theoretical frameworks out of specific empirical evidence. This common culture aids in the understanding and advancement of most subjects, and judging from the success of the two programs detailed in the paper (with three groups of students from each program having gone on to professional IT positions in various organizations), system administration is no exception.

■ ***On Designing and Deploying Internet-Scale Services***

James Hamilton, Windows Live Services Platform

In this presentation, James Hamilton gave the audience a whirlwind tour of the Microsoft Live Platform and how he and his team, driven by the past 20 years of experience, developed best practices for building "operations-friendly" services. Three key tenets empower Hamilton's practices: Expect failures—try hard to handle them gracefully; keep things simple, since complexity breeds problems; and automate everything, because automated processes are testable, fixable, and ultimately much more reliable. Another strong guiding belief is that 80% or more of operations issues (in Internet-scale services) originate in design and development, primarily of applications. As a consequence, if one wants low-cost administration, one must abandon the long-held view that there must be a firm separation among development, test, and operations.

What tasks do operations folk perform in Internet-scale service environments? Because the services change frequently, 31% of their time is spent deploying new applications and features, and 20% entails incident management for problems with known resolutions. According to Hamilton, if done right, both of these are eminently automatable. The most important reason for automating simple incident management is not the relatively small cost of personnel; rather, it is the fact that the more frequently a human being touches software or hardware, the greater the chances of breaking something.

When automated, these tasks may improve the operations-friendliness of your infrastructure by a factor of 2. But Hamilton takes things to the limit with a tenfold increase via recovery-oriented computing (ROC), better designs for applications, automatic management and provisioning, incremental release, graceful degradation, and admission control. ROC assumes that software and hardware will fail frequently and unpredictably. Applications and servers should be heavily instrumented to detect failures. Different failures are caused by two different types of bugs: Bohr bugs cause repeatable functional failures and were usually generated in development. Monitoring should report these with high urgency. Eisenbugs usually occur because of a confluence of ill-timed software and hardware events. Recovering from them involves a series of steps such as rebooting, re-imaging, and finally replacing offending machines.

On the application side, there are some best practices to follow: Develop and test in a full environment; continually perform service health checks in production; make services run on fault-isolated clusters of servers; implement and test the tools that operations will use as part of the service; and partition and version everything. The principles for auto-management and provisioning include the following: Expect to run services over geographically distributed data centers, even if you don't do it now (making your service resilient to high latency); manage "service roles" as opposed to servers (i.e., develop one image, test on that image, and install that image on identical servers); force-fail all services and components regularly, when people are around to fix them (i.e., if you don't test an execution path, expect it not to work); and most importantly, make certain that rollback is supported and tested before

deploying any applications. Rollback works if you always use an incremental process with two or more phases. Finally, when capacity planning, remember that no amount of "head room" is sufficient. Unimaginable spikes will always occur. Instead of wasting resources, find less resource-intensive modes to provide degraded services. If you're ultimately backed into a corner, Hamilton gives you permission to practice admission control (e.g., drop requests from new users).

■ ***RepuScore: Collaborative Reputation Management Framework for Email Infrastructure***

Gautam Singaraju and Brent ByungHoon Kang, University of North Carolina at Charlotte

No one who uses a computer needs to hear another story about how bad the email spam crisis has become. But did you know that NACHA (an electronic payment association) estimated 2004 losses to phishing alone at US $500 million? Email providers and researchers have been fighting spam in various ways for a long time, through content-based filtering, real-time blacklists (RBL), PGP, bandwidth throttling, sender authentication (DKIM, SenderID, SPF, etc.), certification schemes (e.g., Habeas and SenderPath), and reputation management (Gmail). All of these handle spam to different degrees and organizations tend to employ more than one technology to keep ahead of spammers. But why should organizations (especially those with a small user base) fight this menace in relative isolation? Why not band together to leverage their various chosen technologies as a powerful antidote to spam? RepuScore proposes to aid this collaboration.

RepuScore, an open-source effort, allows organizations to establish the accountability of sending organizations based on that sender's past actions. It can be deployed alongside any existing sender authentication technique and collects reputation votes (in favor of or against senders) from existing spam classification mechanisms and individual users. Each organization computes its own reputation "view" of the world and submits it to a central RepuScore authority, which in turn continually generates global sender reputations (i.e., how much can I trust this sender?). The architecture is hierarchical: Each participating domain maintains one or more RepuServers, which classify senders via filters and compute local history-weighted reputation scores for each peer. These statistics are aggregated in a single RepuCollector that averages reports from every server. A single vote on reputations is then sent to a central RepuScore authority, which implements a weighted moving average continuous algorithm.

The original RepuServer algorithm works as follows: In each interval, a current reputation is computed for every sender domain (domains from which email was received) as CurRep = (# of good emails)/(total # of emails). The reported reputation is then calculated as *alpha* * ReportedRep(previous interval) + (1 - *alpha*)*CurRep, where *alpha* is a correlation factor that essentially determines the importance placed on the past reputation of the sender. A lower *alpha* emphasizes current reputation over past reputation, and vice versa.

Singaraju remarks that the ideal behavior for a reputation management system is to have a slow increase in reputation as an organization "proves" itself, but a quick decrease in reputation if an organization starts behaving badly. To achieve this, the researchers modified the original algorithm to only increase reputation if the sender improved from one interval to the next, and to decrease reputation if the sender did worse than in the previous interval. In various tests, high values of *alpha* were able to achieve the desired behavior while remaining resilient to various attacks (e.g., Sybil attacks, which weaken reputation systems by creating a large number of pseudo-identities). RepuScore does make some assumptions—for example, that all RepuServers at your location are secure and reporting correct information and that many organizations are participating—in order to deliver an effective solution.

INVITED TALK

■ ***Homeless Vikings: BGP Prefix Hijacking and the Spam Wars***

David Josephsen, Senior Systems Engineer, DBG, Inc.
Summarized by Tung Tran (tunghack@qmail.com)

Dave said that in the history of spam wars, there are two primary categories of defense: IP-based and content-based spam filters. Dave asked the question, "Who is the biggest user of SPF?" The answer: spammers. (The audience member providing this correct answer received a free book from Dave.)

He then explained BGP prefix hijacking (prefix hijacks make the IPs of others your own) and gave an example to show how it works. Moreover, he pointed out the fundamental reason for BGP's vulnerability: BGP is designed for cooperative use. He showed us how to be a spammer: Get a T1 connection or be a shady ISP.

The Q&A was very intense, with the main discussion focusing on IP-based and content-based spam filters. The first questioner disagreed with the speaker's theory about the attack (BGP prefix hijacking). Dave admitted that the BGP attack might not be too popular. Some questioners raised the issue that content-based spam filtering is not scalable and IP-based filtering is cheaper and still works. They supported their argument by mentioning their specific issue: receiving more than 1 million messages a day. However, the speaker and some others disagreed with this idea. They said that the problem lies not with the content-based filter, but with the implementation of this method. They also asked those who support the IP-based method to publish a paper to better outline their idea.

INVITED TALK

Beyond NAC: What's Your Next Step?

Mark "Simple Nomad" Loveless, Security Architect, Vernier Networks, Inc.

Summarized by Nathaniel Husted (nhusted@iupui.edu)

Mark said that Network Access Control (NAC) provides a way of regulating and controlling access to the network. A NAC initiates this process when the machine starts up and tries to access the network. NACs are also a way of enforcing policy on the endpoints of the network. Mark specifically stated they are neither a security nor policy solution but an enhancement.

Mark also discussed how NACs should be implemented and how they should perform. He illustrated this by telling a story: A VP is at an airport, trying to make a very large business deal that is worth hundreds of thousands or even millions of dollars. The VP has visited various Web sites, thereby infecting his computer with malware, viruses, and other programs of ill repute. The question posed at the end of this story was, Do you allow him onto your network to look up some information and close the deal? Mark said that ideally you should, but make sure he only has access to the resources he needs to complete the deal. This will mitigate much of the damage he could do if he had full network access.

The NAC should also be an inline solution and it should be very fast. The latency should be under 1 millisecond and should be in the microsecond range. This includes any IDS or IPS services involved in the solution. The solution must react to events in real time. It also must be able to react to a very large number of these events. It must work as well with thousands of users as it does with one user. The IDS and IPS services the NAC implements must also be state of the art. They must be able to decompress GZIP traffic on the fly and handle the numerous protocols that are out there. The NAC should also be seamless and scalable. It should require no changes to the existing infrastructure, regardless of the size of the infrastructure.

Another problem with implementing a NAC is deciding upon ownership. Mark said that one customer actually did not purchase a NAC solution because the customer could not decide what department within the organization would have control of the NAC. Since so many different departments are involved in a NAC solution, choosing who runs it is an integral part of its deployment.

NACs are also not a static solution; they need to adapt. Attackers are constantly finding ways to bypass NAC solutions. These attackers were not botnets or external sources, but contractors and evil end users. Many users will do this just to avoid NAC policies. This can be done by spoofing various data. This data spoofing generally involves default policies applying to equipment such as printers. Also, certain MAC addresses that are allowed to manage items on the network are prime spoof targets. NACs have to adapt not only to attacks but also to new technology. They have to support all platforms and changes to those platforms. They also need to adapt to policy changes (e.g., a directive forbidding management accounts from being allowed on local computers). NACs also need to enact policy post-authentication and be able to cover a broad range of policy decisions, such as allowing IM but disallowing file transfer within IM. Mark suggested that the IPS be tied into the system and able to enforce these policy changes on the fly.

Another important factor of a NAC is its ability to cope with the ever-increasing mobile workforce. The NAC must work over wireless, dial-in, and VPN interfaces. The NAC also must deal with contractors and guests who require access at your organization. Mark suggested that all NACs should at least have the ability to access the Internet and use a single printer.

NACs are not a replacement for perimeter technology. Antivirus servers, firewalls, and other network security devices are still needed to protect against client-side attacks. You should also have technology in place to protect against alternate routes of attack.

Mark also discussed some things that NAC vendors do not tell you. The first is that after authentication, users still could be doing bad things. It is possible for the user to spoof information that checks for system compliance so that their virus-laden computer can connect to the Internet. Mark again stressed the fact that, to be effective, the NAC solution must be inline, in the core, in the perimeter, and everywhere else on the network. The system must be a mediator among all users on the network, and some NAC solutions are not. Vendors also will not tell you that NACs only control access to network resources and do not control access to applications and data independent of network resources. Mark also said that tunneling protocols can bypass virtually all vendors on the market. Also, NACs do not help if sensitive material in need of protection resides in the data that can be accessed. One example Mark gave involved a person with legitimate data access collecting data snippets and combining them to form a position for insider trading.

Mark finished his talk by discussing where NACs are heading. He sees future NACs being able to identify more than just who is accessing the network; they will also identify what data they are using and what applications they are using to access that data. Future NACs will be able to use layered profiles to limit network access. They will limit access based on user identity, application usage, and data usage. He also sees NACs providing easy correlation of events to help administrators put seemingly unconnected events together to solve a bigger security puzzle. Mark also sees NACs providing more automated reactions to events on the network than current IPS solutions. In general, he sees NACs becoming more automated as time progresses.

There were few questions asked during this session but the answers provided were detailed. Mark was asked his opinion of signature-based IDS and IPS systems and how viable he thought they were in the immediate future. Mark said that he wasn't a big fan. He suggested that a company have at least one commercial solution but should back it up with Snort. He also suggested that a company should use a combination of both anomaly- and signature-based systems to cover the full range of scenarios.

The second question concerned agents and the effectiveness of NACs against encrypted traffic such as SSH. Mark explained that an agent would ideally be in programmed Java to allow for maximum cross-platform usage. In general the NAC will be unable to read SSH traffic, but in some regards it can be predictable. He said that some studies have shown that the first thing an administrator does when logging into a machine is type `su`. This has a clearly defined length and could be detected even if the traffic is encrypted. There is nothing more that can be analyzed from the encrypted traffic beyond correlation.

The final question was about the validity of Gumjack (systems on a USB device) in NAC situations. Mark said that it could scale well but you would have to buy a Gumjack device for every employee's computer, so the logistics and economics could be daunting.

INVITED TALK

The Economic Meltdown of Moore's Law and the Green Data Center

Kenneth G. Brill, Executive Director, Uptime Institute

Summarized by Kimberly McGuire (klmcguir@iupui.edu)

According to Moore's Law, the number of transistors on a piece of silicon will double every 24 months. In fact, the number of transistors on a piece of silicon has been doubling every 18 months, faster than originally predicted by Moore's Law. However, this increasing rate of computational performance is greater than the rate of power efficiency improvement. The result of multiplying increasing computational performance with energy efficiency improvement is that more and more electricity is being consumed at the plug. In 2005, Dr. Koomey of Stanford and the Uptime Institute estimates that servers used 1.2% of the electricity generated in the United States; this figure is up from 0.6% in 2000.

This lag in power efficiency will drive a site's Total Cost of Ownership (TCO) up and reduce economic productivity. Because of these increasing power needs, square feet costs of a data center are irrelevant; costs will be driven by the power consumption of the IT equipment. The increasing demand and cost of electricity has big tech companies moving to areas where they can get power at less than $0.03 a kilowatt-hour or to areas of the country that have surplus power.

Mr. Brill suggested four metrics to determine whether your data center is "green": (1) IT strategy optimization; (2) hardware asset utilization; (3) energy-efficient hardware deployment; (4) site infrastructure overhead minimization. Turn off machines that aren't doing work, virtualize, and use what you have efficiently. As spindle speed doubles, power consumption goes up by a factor of 8. Does everything need to be on the fastest disk? Buy energy-efficient hardware. It's available but currently does cost slightly more, but you'll likely save in incremental cost. Remember that less than half of what comes out of the plug goes to computation. The remainder is overhead.

A green data center for a large global enterprise can make an estimated $100 million in profit or competitive advantage over 10 years. Scaled-down savings are available for smaller centers.

Business units want the latest and greatest equipment for their money. However, IT's current economic chargeback systems typically fail to take the true cost of ownership into consideration. Part of those new, faster computers is the cost of electricity, which as a single site cost element all by itself will soon exceed the cost of the server over three years. Unfortunately, this is only one of several major site infrastructure cost components that need to be billed back to users.

Until the IT cost chargeback system is fixed to determine true costs, users will be motivated to make suboptimal decisions. It is the responsibility of an IT manager to explain those true costs and try to convince business units that they don't have to sacrifice much in performance to see a substantial reduction in the three-year TCO for a piece of equipment. Ideally, chargeback to those business units is key to containing TCO for a site.

Another way a company can reduce site costs is by using the Information Technology Infrastructure Library (ITIL). ITIL was originally developed by IBM for the United Kingdom. ITIL uses checklists and detailed descriptions of processes and practices that can be tailored to fit any IT organization. Mr. Brill pointed out that a site's IT equipment needs to be included in the configuration database when building or moving to a new data center.

CONFIGURATION MANAGEMENT

Summarized by Kevin James (kevljame@cs.iupui.edu)

Moobi: A Thin Server Management System Using BitTorrent

Chris McEniry, Sony Computer Entertainment America

Chris presented a solution used by Sony Computer Entertainment America (SCEA) to update and deploy their 2000+ game servers. Moobi is an image distribution system based on PXE, DHCP, TFTP, and BitTorrent that he developed after years of searching for ways to deploy his

many server instances and minimize downtime. He began by describing the motivation for Moobi.

SCEA's online gaming servers grew from 350 in 2004 to more than 2000 in 2007, while tasking two administrators at most to maintain them. Initially he turned to Cfengine but found that he was unable to express what he needed using this powerful tool. Chris is quick to say that the problem was not with Cfengine; they simply couldn't accurately express what they needed in their classes. In one instance, after updating the ntp configuration on the game servers, several cascading events caused the servers to crash. Realizing several deficiencies in their process, Chris was inspired by the Linux Terminal Server Project (LTSP). LTSP works by intercepting the normal boot order of the kernel. He thought, "Why can't we do other environment setup here?" The decision was to load an image during this step.

Unfortunately, the booting of the servers became a bottleneck. He decided to leverage the spare capacity of the servers and available network bandwidth and run the BitTorrent client during the initial kernel boot. By sending different segments of the image to different nodes within a subnet, not only are the servers able to load an image faster but slow-to-start servers are able to quickly catch up. After running a controlled experiment, Chris reports that Moobi running with only three boot servers was able to update an impressive 600 nodes in approximately an hour, with 95% of the machines finished in the first 15 minutes. The last 5% were slowed by PXE boot failures.

In the future, Chris plans to port Moobi to more OS distributions and provide better hardware detection on boot. Another improvement would be to develop a method for integrating Moobi into existing configuration management tools.

- ***PoDIM: A Language for High-Level Configuration Management***

  Thomas Delaet and Wouter Joosen, Katholieke Universiteit Leuven, Belgium

  Awarded Best Paper!

Thomas Delaet presented PoDIM: an object-oriented language aimed at creating high-level specifications for configuration management tools. Instead of defining what processes are necessary to complete the configuration of a host or how to enforce the configuration, PoDIM focuses on defining the relationships between different entities within the host and network, delegating the details to various tool-specific translators. He cites Paul Anderson's paper "Towards a High-Level Machine Configuration System" given at the 8th annual LISA Conference [*Proceedings of the 8th Large Installation Systems Administration (LISA) Conference* (Berkeley, CA: USENIX, 1994), pp. 19–26] as a reference.

In PoDIM, "all 'things' are objects." It leverages existing research in language creation and software engineering in the "Rule Language," used to define your site policy. Delaet lists static typing, multiple inheritance, and contractual programming constraints (preconditions, post-conditions, and class invariants) as advantages of their approach. Another advantage is the use of object references when attributes refer to other objects versus actual object copies. This is used to define dependencies between different object classes. One creates a site policy by creating several rules and constraints that define the composition of each object as well as how their attributes may be modified. This is accomplished by using an SQL-like syntax.

After the site policy is defined, it is fed to the PoDIM compiler, resulting in several complete object descriptions, much like a normal compilation step. In response to an audience question, Delaet explained that failures during this step do not necessarily cause the entire compilation process to fail. Only objects that depend on a particular rule definition, either directly or through reference, fail during compilation. These are used by a configuration-tool-specific templating engine, which generates the files necessary for that tool. The code that results is then supplied to the configuration system. This allows for easy integration into current configuration frameworks. The current reference implementation for the templating engine generates Cfengine code.

In the future, Delaet plans to introduce greater modularization into the translation process by separating the rule logic from the object definitions. Other improvements include simplifying the integration of PoDIM into higher-level tools and GUIs, creation of templating engines for other tools (LCFG, Bcfg2, Puppet), as well as a method for translating the native configurations of such tools into PoDIM. Finally, he stressed the need for a communication mechanism to facilitate the resolution of cross-machine dependencies.

- ***Network Patterns in Cfengine and Scalable Data Aggregation***

  Mark Burgess and Matt Disney, Oslo University College; Rolf Stadler, KTH Royal Institute of Technology, Stockholm

Matt Disney presented the results of work toward introducing decentralization into Cfengine. Recognizing that centralized management strategies will eventually fail on some level, they took cues from network management patterns to develop decentralization schemes in Cfengine's monitoring. Drawing on graph and tree traversal algorithms, they developed a logical overlay network for each scheme, independent of the actual physical layout of the network. Each scheme is characterized by an expansion phase, during which nodes are queried, and a contraction phase, during which the responses of each node are aggregated. One highlighted application of this approach is in the field of autonomics, during which such feedback from nodes is necessary in the reconfiguration process.

To study the behavior of these schemes, three different experiments were developed, and statistics were collected over 50 runs. They first tested a scheme called Echo, in which a query is pushed to nodes arranged in a tree overlay and then the responses are collected; this is repeated and compared to the performance over parallel star and serial star overlays. The results from this experiment showed that although the parallel star overlay performed the fastest and the serial star overlay required the lowest workload, the echo overlay provided a nice trade-off between the two, running in half the time of the serial star while generating one-fifth the workload of the parallel star.

The next experiment, GAP chain, arranged the nodes into a chained overlay, but did not use an expansion phase. Instead, responses were simply aggregated from the nodes. After test runs similar to the Echo experiment, the results showed that the GAP chain performed even better than they had originally expected. Further investigation of this showed that adding a sleeping factor to the nodes increased performance, even allowing for a single-cycle update after some adjustment.

The third experiment used the same methodology as the second, but with a tree overlay. The results showed the performance of this overlay to be quite stable, but similar to the chain; greater performance can be achieved by adjusting the sleeping factor attached to the nodes.

Although the results of their experiments are quite promising, Disney does report some limitations and errors they encountered. Virtual machines were used to simulate the test network; therefore the time on each node could have become skewed. Also, the sample sizes were small, only 20 nodes; he believes that more representative results could be obtained with larger sample sizes. In conclusion, they plan to implement more pattern overlays in Cfengine. To facilitate this, the group plans to explore enhancements to pattern specification in Cfengine.

INVITED TALK

Hardening Your Systems Against Litigation

Alexander Muentz, Esq.

Summarized by Kimberly McGuire (klmcguir@iupui.edu)

First and foremost, Mr. Muentz does not work for Microsoft. The information contained in this summary or in his presentation is not legal advice and is for informational purposes only. This area of the law is in flux and what may be law today may not be tomorrow.

Civil litigation is an IT risk for which preparations must be made in the event a lawsuit is filed against your company. Civil litigation poses a security risk as it allows outsiders to view and handle sensitive data and could potentially lead to financial losses for you and/or your organization. There are myriad reasons a person or persons could file a civil suit against a company.

A civil lawsuit starts with a complaint that lists all legally supported claims. The next step in the process is discovery. During the discovery process each side produces all responsive information related to the lawsuit. Additionally, during discovery each side gets to interview, under oath, selected individuals from the other side and each side can subpoena information from third parties with relevant information. Discovery is based on good faith; if either side fails to produce relevant information purposely or accidentally, they can face fines, data recovery fees, dismissal of claim or defense, dismissal of lawsuit, or loss of suit. Finally there is a settlement, trial, or arbitration to determine the outcome of the lawsuit.

Litigation is so expensive primarily because of the discovery process. Once a civil lawsuit is filed a litigation hold is put into place requiring you to preserve all responsive data and documents. Data and documents include but are not limited to email, digital documents, voicemail, backup tapes, system logs, and slack space on disk drives. Then the data and documents are collected and a discovery conference is held. During this conference each side discusses the sources and people they have and sets a schedule and format. After the discovery conference all the data is reviewed at least twice—in some cases, three times. The first review is usually done by a junior attorney; the second and third reviews are done by more experienced lawyers. At between $90 and $150 an hour for each lawyer it is easy to see how quickly the expenses can grow.

The discovery process is also the biggest security and privacy risk for a company and its employees. It is a privacy risk for employees because of the grey area between personal lives and business. It is no longer uncommon to find people who work from home on a regular basis or use personal email for business. It is an IT security risk because of the broad sweep of the process. The law firm takes everything and anything that may be related to the civil suit. The law firm may have inadequate security or may contract out some of the work to third-party vendors, leaving sensitive data in insecure hands.

What can you do to prepare yourself? Do an ESI (Electronically Stored Information) audit. Identify all key systems and determine their contents. Use policies to define retention of ESI, how users can remotely access systems, the decommissioning of systems, and use of personal email for work, and follow those policies. Finally, implement a collection plan for end-user PCs and file servers. Preparation is key.

There are also some steps you can take if you find yourself already in the middle of litigation. First and most importantly, cooperate with your lawyers. Enforce the litigation hold and request additional storage capacity to handle the

additional data. Attend the discovery conference, assist in working out a technical plan, and be prepared to correct any bad technical information the other side may be trying to pass off as legitimate. If required, help select third-party vendors to ensure that data is reviewed in a secure location. If you are deposed, explain exactly what you did and why you did it.

Alexander Muentz is based in Philadelphia, PA, and is licensed to practice in the state courts of New Jersey and Pennsylvania. The slides from his presentation are up at http://www.usenix.org/events/lisa07/tech/muentz_talk.pdf, and a related article appeared in *;login:*, Oct. 2007.

SPECIAL LUNCH & LEARN

Should the Root Prompt Require a Road Test?

Alva L. Couch, Associate Professor of Computer Science, Tufts University

Summarized by Kevin James (kevljame@cs.iupui.edu)

At this year's Lunch & Learn session, Professor Alva Couch led a discussion on an issue that has great importance for the future of the system administration profession: What makes a good system administrator and how do we measure this? Alva began by asking, "Is there a mysterious compound 'W' that makes system administrators functional and ensures success"? Often certification is thought of as being the proper way to become a good system administrator, but Alva believes that we have this relationship backward: certifications serve experienced system administrators far more than new ones. He takes the stance that certification tests cannot measure what makes a good system administrator, "Quality X," but instead measure an individual's knowledge of a specific product or brand. Yes, we should attempt to determine functional system administrators by certifying them, but there are some things that tests cannot measure.

He proffered driving as a metaphor for our current problem, as "we are drivers of the technological revolution." We certify a driver's knowledge using a written test and skills using a road test; our current testing frameworks accomplish the first well enough, but there isn't an equivalent for the road test in system administration. Again, Alva believes we are asking the wrong question. A better question would be, "What is the difference between new drivers (and sysadmins) and more experienced ones?" Accidents: New drivers are associated with higher accident rates, and rates seem to decrease with the experience of the driver. Alva credits this to an increase in situational awareness and judgment, which allows drivers, system administrators, and even pilots to "understand the broader effects of your actions." These make up our Quality X, that which makes sysadmins good. This quality is only attained through causing accidents and learning from them.

Having identified Quality X, Alva continued breaking down its role in system administration. Situational awareness entails determining not only what could be wrong and what could have caused it but also the side-effects of your solution both on your systems and on your consumers. The extent of one's situational awareness can be considered the "maturity level of a system administrator"; when solving problems, whether our focus is limited to the local system or extends to the whole enterprise and beyond to your lifecycle planning depends on the amount of experience you have attained. Mentoring becomes indispensable; the inexperienced can learn from their mistakes in an environment where someone can guide them on a path to greater awareness.

He concluded that we will never be able to measure maturity as an administrator and therefore should give up on knowledge-based tests to achieve this. Instead, we should focus on increasing the availability of mentoring and other methods for expanding experience. Handing the discussion over to the audience, Alva left us with a few thoughts. In the technological revolution, we are drivers. We want "professional" status and "respect." Please drive safely.

WORK-IN-PROGRESS REPORTS (WIPS)

Summarized by Gautam Singaraju (gautam@singaraju.com)

Fettle, a Program for Populating and Dressing Racks

Andrew Hume, AT&T Labs—Research

Maintaining multiple numbers of servers and racks poses a significant administration problem when placement at different locations is required. Andrew Hume developed a program that allows users to place the servers on racks based on the number of Ethernet connections, power supplies, switches, etc. The tool creates a 3D presentation to show how the servers can be placed. The tool, Fettle, should be available soon on sourceforge.net under an open source license.

Excellent Performance with Aging Hardware

Alberto D'Ambrosio, National Institute of Nuclear Physics, Turin, Italy

Citing the Brooklyn Bridge as an example, Alberto D'Ambrosio suggested that system administrators now have to monitor and support systems developed by others. At his organization, two machines were used as mail servers. These could handle the load for the first few years, simply fixing any problems that started showing up. However, as spam started increasing, the servers began to reject emails, and processing and storage increased tenfold. The cluster had become less reactive owing to SCSI starvation. Performance increased once they relocated to a Bayesian database. They started recycling their old servers to provide additional performance benefits.

■ *What's New in Amanda: The Open Source Backup Platform*

Dustin J. Mitchell, Storage Software Engineer, Zmanda, Inc.

Amanda supports a device API that allows pluggable storage backends such as tape, disk (vtape), RAIT (Redundant Array of Independent Tapes), WAN, and optical media. Application API integrates Amanda with tar, dump/restore, different databases, Windows, AFS, and NDMP. The Amanda transfer architecture (XFA) has a client-server model, in which the client passes the messages to the supervisor, which is present on the server over the network. The client compresses the data received from the application, which is then encrypted by the server and sent to the taper. Perl in the code allows new contributors to join the system while providing low-level processing and using the high-level libraries. Dustin Mitchell invited new developers and contributors to join in development of Amanda.

■ *Analysis and Verification of XACML Policies*

Saurabh Arora, Pablo Giambiagi, and Olav Bandmann, Security, Policy and Trust Laboratory, Swedish Institute of Computer Sciences, Sweden

XACML provides access to a rich language for expressing security policies and makes it possible to integrate many different authorization models into the same framework. Policy management tools need to be enhanced in order to help system administrators design sound policies, support policy change management including policy optimization, facilitate cooperation among administrators, support GUI functionality, and support properties that are not directly expressible in standard policy languages (e.g., Separation of Duties or Chinese wall). In the upcoming XACML 3.0, one of the most important additions is a mechanism for delegation of security administration. It provides a rich language for expressing security policies in which policies can be issued by authorized issuers. The new specification can be used to implement decentralized administration in applications and a mechanism for delegating security administration.

The authors developed a Policy Analysis Subsystem (PAS) that translates policies and queries to propositional logic. External data such as XACML policies, attributes, and relations can be fetched by PAS to compose queries. Saurabh discussed SAT solver as a tool for (a) solving the Boolean satisfiability problem and (b) analysis of counter-model examples. PAS can iterate queries and/or adapt queries based on results of previous queries, as needed, to express higher-level queries.

■ *Grid Services at Yahoo! Comes to LISA*

Marco Nocosia

Marco Nocosia introduced Hadoop, a distributed file system and map-reduction programming platform designed to scale out to satisfy the requirements of a full-scale Web content system. Typical SAN and NAS do not support enough storage or IO bandwidth. Hadoop combines the storage and computation power of any set of computers and is written in Java; the user's program can be written in the user's preferred language. Hadoop has been developed as an open source project and interacts with HDFS. The information can be seen in a browser and Hadoop allows the clusters to run both DFS and M-R.

■ *MachDB*

Nathan Hubbard

Every organization rolls their own machine DB, which is usually independent of the organization's. MachDB is a scalable, open source implementation of the most needed of system administrator tools to maintain the machine database. MachDB began as a quickly growing startup, but it is scalable for enterprise environments. There are several design goals for MachDB: Information gathering should be architecture- and OS-agnostic, the back end should be LAMP, the XML spec should be the API, and it should be scalable to 10,000+ hosts, have human and machine readable interfaces, have an easy-to-use Web front end, and offer a history on everything and templates for easy UI modifications. The code is now in its alpha phase and will be released in a few weeks at the project Web site: http://www.machdb.org.

■ *mfork*

Maarten Thibaut

Maarten Thibaut needed to synchronize huge amounts of data among multiple servers. Serial rsync does not serve this purpose because it requires too much time. Maarten suggests using parallel rsync, which uses a fork mechanism called mfork, a simple command that allows parallelization. The command mfork forks rsync and copies data. Make was not used, as it depends on gnu and takes additional time. Parallel rsync also allows users to save the results without any issues of parallelization.

■ *Migrating to Internet Protocol Version 6 (PDF)*

Dennis Underwood and Jon Lavender, University of North Carolina at Charlotte

Migration to IPV6 is a long-term necessity, but experience with the protocol is limited and usage and implementation policies need to be established. However, migration from IPv4 to IPv6 affects the entire network and middleware will need to be replaced with end-to-end administration policy. Although the new protocol eases network administration, associated technologies keep developing rapidly. Dennis Underwood and Jon Lavender suggest that policy should be valued first and different alternatives should be devised for short-term and long-term migration strategies. Ignoring IPv6 is not an option; their alternate option of immediate migration has advantages and disadvantages. Because the models are still developing and vendors may not leverage all IPv6 capabilities, immediate adoption is simply not possible. The authors suggest the development of long-term policy strategies to gain eventual full IPv6 connectivity.

■ *User Satisfaction and Technology Acceptance Among System Administrators*

Nicole F. Velasquez, University of Arizona

Nicole Velasquez studies user satisfaction and technology acceptance among system administrators. Human-Computer Interaction (HCI) among system administrators is completely different from that of the usual users. Presently, more money is being spent on human development cost. Nicole uses verification and login information to determine information quality. The system quality that is being used should be reliable, flexible, and integrable. Nicole proposes scalability, credibility, and situational awareness.

■ *Frequency Domain Analysis and Visualization of Web Server Performance*

Marc Chiarini and Alva Couch, Tufts University

Frequency domain analysis and visualization of the Web server provide an external model of behavior of the Web server. It allows one to check how a Web server responds to simple requirements. With the help of frequency analysis, the server measures the effects of increasing load and checks for abnormal behavior. Input classes are important for performing frequency domain analysis because they take into account the different types of inputs coming into the system by checking the caching mechanisms, dynamic pages, and database server performance. The authors demonstrated their frequency domain analysis using graphs and discussed how the frequency domain analysis helps in plotting expected domain behavior and the performance of the servers.

■ *How Do You Protect the Data of 1500 Very Nontechnical People?*

Ski Kacoroski

Dealing with a nontechnological user base usually implies that people cannot back up their critical information. The solution presented here is to use backup.sourceforge.net. This enabled machines at a K–12 school to be fully backed up. The helpdesk has a Web site that can be used and backups can be made using different techniques. The system is optimized to back up the users' data at night.

INVITED TALK

■ *Prince Caspian on Location: Commodity Hardware and Gaffer Tape*

Trey Darley, Technical Consultant

Summarized by Kimberly McGuire (klmcguir@iupui.edu)

Working on the set of a big-budget, major motion picture sounds like a great job. Traveling across Europe to exotic locations, Trey Darling did this for the new Narnia movie, *Prince Caspian*, for approximately six months. Trey worked with a team of four for Walt Disney Productions and Walden Media. He started working at the Barrandov studio in Prague, Czechoslovakia. He immediately found himself working as a reactive system administrator. Because of the way contracts were set up, cast and crew brought their own equipment, including their own computers. This myriad of computers required expertise to support OS 9, Vista, and everything in between.

Working in a building constructed during World War II presented its own problems. Thick walls made it difficult to run cable. Requests for connectivity would come in on a Friday to be completed by Monday. The original network was to be for 50 to 60 users, but by the time Trey left the project the network was handling 500 to 600 users. The Linksys switches, originally selected for a network of 50 to 60, had to be replaced every two weeks. Management determined that it was less expensive to buy and replace the cheaper switches than to spend more money on switches that could adequately handle the traffic.

Next on the location list was Bledne Skaly, Poland, a small town in the middle of nowhere. The base camp consisted of trailers and tents, both of which changed configuration daily. Trey started in a pop-up tent, with a battery and a bench. Everything in the tent, including equipment, had to be packed up during thunderstorms, which hit the area regularly. The IT department eventually got its own trailer. There was no Internet access in Bledne Skaly, so they had to use a wireless modem. The base camp and "the set" were divided by a stretch of woods. To get Internet access to the set from the base camp required a series of wireless access points to be constructed through the woods. Trey and his team used garden hose as conduit to run cable through the muddy fields that surrounded the base camp.

Good-quality services were difficult to find. An example Trey gave was when one of the stars of the movie required a data recovery service on a dropped laptop. The team found a company to do the data recovery, but the hard drive was seized by the police during a raid on the company. After "talking" to the right people the drive was recovered.

Avid Unity systems were used for redigitizing and cutting the movie. Each Avid system stored 25–30 terabytes of data. There were no backup plans in place for these systems and instead of getting good-quality power converters, each $20,000 system used a $20 converter.

Trey was asked whether he would do this kind of work again. While he never said yes or no to the question, he did say that if he did it again he would press production hard to get a better feel for the scope.

INVITED TALK

■ *The Security Butterfly Effect*

Cat Okita, Earthworks

Summarized by Nathaniel Husted (nhusted@iupui.edu)

Cat's talk involved quite a lot of audience interaction. In fact, in a response to an audience member, she said that if

you have ethical problems with raising your hand, please feel free to raise any other body part. There was also a member of the audience who provided sound effects for the technical session. Although these portions of the session cannot be recreated in the summary, the educational information can. It is recommended that the readers view her PowerPoint slides at http://www.usenix.org/events/lisa07/tech/okita_talk.pdf while reading this summary.

Cat introduced her talk by posing the question, "Does the flap of a butterfly's wings in Brazil set off a tornado in Texas?" This was a question asked by a paper in the 1960s. The author of the paper did one set of calculations, then manually entered his calculations a second time and received a completely different answer. The Butterfly Effect is when small variations in the initial condition of a system produce large variations in the long-term behavior of the system.

Cat then defined the characteristics of the Butterfly Effect, or rather what it is not, since it is a subtle effect. It is not the domino effect, nor is it linear or cascading. It does not involve one clearly identifiable thing leading to a problem that spans out.

She then asked whether any of us know the initial conditions of our systems. The answer to this question is no. As we become more specialized and modularization increases, and as our software becomes more complex, this situation will get worse. It is from this lack of knowledge that we make assumptions about our systems.

Cat split the assumptions we make into three categories: environmental assumptions, behavioral assumptions, and blind spots. Environmental assumptions are those in which we think we know everything about our environment. As an example, she gave "Everyone knows it doesn't snow in the deserts." In fact, sometimes it does. Behavioral assumptions are those based on how people and systems behave. Blind spots are situations when we think a system always works in a specific way.

The rest of Cat's talk was structured as nine specific stories created from these three assumptions. The stories were split into three sections: the cast of characters involved, the problem, and the assumptions.

The first story was about THERAC-25, a medical linear accelerator produced by Atomic Energy of Canada Ltd. (AECL). It was inspired by the THERAC-6 and the THERAC-20. These older models were created by AECL as well as CGR, a French company. Before the THERAC-25 was built, there was a fallout between AECL and CGR. AECL decided to reuse the software from the THERAC-6 and THERAC-20, but not implement any of the hardware controls that the THERAC-6 and THERAC-20 used. The THERAC-25 would only use software controls, but the THERAC-25 would either underdose or overdose the patients. It would also produce strange errors and sometimes just not work. AECL explained that this was not the company's fault; it must be user error. After the deaths of some patients and radioactive overdoses of others, AECL finally agreed to add hardware controls. The assumptions made in this situation included the idea that a hardware company could write software and that the users were not providing accurate information. The first was a blind spot; the second was an environmental assumption.

The second story, "Samy Is My Hero," involved the MySpace social networking site and a boy named Samy who was interested in looking at pictures of the fairer sex. To look at these pictures he needed a large number of "friends." To reach this goal he put a little snippet of code in his profile that automatically added any viewers of his profile to his friends list. This code would then add a copy of itself to the visitor's profile as well. MySpace was not amused. In this situation MySpace assumed that users would not and could not put malicious code in their own profiles. This was both a behavioral assumption and a blind spot.

In the third story, "How to Fit an Elephant Into a Teacup," a secure data center was experiencing impressive growth and had a set of "world-class" access controls to secure the building. The operation of these world-class access controls was based on the weight of the subject. The problem was that the access controls thought one of the data center's talented employees was more than one person. This happened because the weight cutoff for the access controls was 400 pounds. This also meant that multiple people with a cumulative weight of less than 400 pounds could get in. To solve this problem, the staff members propped open the fire door of the "secure" data center. There were multiple behavioral assumptions in this situation. The first was that everyone weighs the same. The second was that multiple small people would not enter the facility at once. The third assumption was that people would never go through the fire exit unless it was necessary. The final assumption was that the physical security staff would not open the doors for someone who shouldn't be there.

The fourth story, "A Scandal in Bohemia," dealt with paranoid privacy enthusiasts and a "security researcher" named Dan Egerstad. The privacy enthusiasts were using Tor as a way to keep their browsing habits secret. Tor is a program that redirects Internet traffic to make it hard to identify where the traffic originated. It also makes it hard to block the traffic. The fine print, however, states that it does not protect any information in the traffic and provides no guarantees. Dan Egerstad posted details of sensitive email accounts he received from sniffing the traffic at the edge of the Tor network. The paranoid privacy enthusiasts were not amused. One of the assumptions the paranoid privacy enthusiasts had was that Tor would hide sensitive information. This was a blind spot. The Tor group also made an assumption that people read fine print. That was a behavioral assumption.

The final story, "Monkey Business," involved, well, monkeys. This story begins with the crashing of a VAX main-

frame. A Digital Field Service Engineer (DFSE) was called in to fix the problem. The system administrators didn't call the researchers, because nothing looked unusual about this VAX. Problems arose when the DFSE ran a diagnostic on the system. It turns out that everyone neglected to heed a sign saying "Do not disable read-only mode" on the drive controller. After the diagnostics everyone found out there were monkeys attached to the VAX and monkeys do not operate in write mode. Some monkeys were recovered, but others experienced "fatal errors." The biggest assumption in this story was that the system administrators knew exactly what the machine was being used for. This was a big blind spot.

After Cat told her nine stories, five retold here, she discussed what she saw in the future of computing. In summary, she found the Butterfly Effect to become more prominent. Cat warned that as we continue to become more specialized people will know less and less about things outside their area of specialization. We're also experiencing a "rise of the machines." It has become more common to outsource what we know to machines and hope they properly take care of all the blind spots and other assumptions.

In response to a question about applying the wisdom from this talk to smaller systems, Cat affirmed that this Butterfly Effect analysis is most certainly applicable to smaller systems. She said that many times things will work fine with one or two programs, but when three or more are introduced, weird things can happen. In response to another question she advised that we should always expect the unexpected and we can glean wisdom from the superstition, myth, and lore we pass around as system administrators. Finally, she said that there is value in differing opinions.

CLOSING SESSION

Cookin' at the Keyboard

David N. Blank-Edelman, Northeastern University CCIS; Lee Damon, University of Washington

Summarized by Josh Simon (jss@clock.org)

The conference's closing session began with the usual close-of-conference announcements, then we segued into "Cookin' at the Keyboard" with David Blank-Edelman and Lee Damon. While neither has any formal culinary experience, both like to cook. Lee demonstrated by preparing (from chopping the vegetables through serving it to David), on-stage while David spoke, a tofu and vegetable stir-fry and ice cream with homemade hot chocolate sauce. Unfortunately, for liability reasons, David and Lee were unable to share the stir-fry or the chocolate sauce. David spoke about how system administrators could learn from restaurant cooking procedures. First, as an appetizer, David spoke about why cooking is hard: You're not just applying heat to some food, you're managing the conditions with lots of variables, such as the quality of the ingredients, the temperature and humidity of the air, and the level of heat involved.

As the first course, David discussed recipes. Based on discussions with cookbook authors and chefs, he talked about how writing recipes is hard. You never make the same food twice, you can't go into explicit detail at every step (including such things as the suppliers of the food and manufacturers of the stove and pans and so forth) without scaring your audience, and most people don't use common sense in terms of recovery (if a recipe says "cook 10 minutes" they'll cook it for 10 minutes even if it's obviously done after 5). Solutions to these problems are to treat recipes as general guidelines and to never expect someone to duplicate a recipe but instead to approximate it. You also find that cookbooks specify common units and time ranges and provide visual and textual clues to let the reader (cook) make judgments. As you get more experience, you can come up with simpler recipes with fewer ingredients to achieve the same or better flavors. In other words, the better you get, the simpler it gets. So learn to simplify: It takes experience. Learn where to cut corners, when to ask questions, when to question every ingredient, and how to compromise when necessary.

As the second course, David had talked to several chefs about working in a world-class kitchen. Starting with a definition of restaurant terms and continuing through a comparison of trade (such as a burger-flipper or help desk worker) to craft (cook or system administrator) to art (chef or ubergeek), he went through the skills you need. The chefs agreed that to be a good cook you'd need a sense of urgency, the ability to take direction and clean up as you go, precision, a thorough knowledge of the subject matter, initiative, focus, and dedication. You also need to be part of a team, be willing to jump in and help when needed, and be able to receive new information and produce with it. These skills, with minor changes from food-industry to technological terms, describe much about system administration. In the case of cooking, preparation (mise en place) is included in the process. You need to be prepared physically and mentally; you need to know where everything is and have everything at hand, ready when you are, and be as fast and as efficient as possible. As you get more experience you're able to work better under pressure, to help others, and to show your creativity.

Finally, for dessert, David provided an overview of what we as system administrators can take away from the talk. We need to write better recipes and recipe interpreters, such as configuration management tools. We need to develop our skills and moves better. We need to prepare, work clean, and focus on the task. Finally, like a line cook becoming a chef, we need to chase perfection: Take teaching opportunities but not necessarily every learning opportunity, communicate with your team, document what you do, learn more things, ask for help when you need it, and be able to roll back and start over when you have to.

Fighting Spam: The State of the Art

Chris St. Pierre, Nebraska Wesleyan University

Summarized by Chris St. Pierre (stpierre@NebrWesleyan.edu)

Although spam is far from a solved problem, most attendees at the Spam Fighting workshop appeared to consider it about 95% or more solved. The remaining 5% still poses a concern, though, as does the fact that spammers have a growing army of programmers and computers dedicated to eliminating or obsoleting the hard-earned gains that have gotten us to this point.

The issue is complicated by the fact that nowadays spam is rarely an issue that is isolated to incoming mail. Many of the attendees had concerns about outgoing spam, whether sent or forwarded by their clients. Soft topics in spam fighting, including access and acceptable use policies, amount of functionality granted to the user, and more, were also discussed.

Currently one of the biggest guns in the spammer's arsenal is the botnet, and we detected botnets and ended spam from them. Tools such as p0f, which passively detects what operating system a given connection is from, are gaining traction since they allow mail server operators to score or reject messages that come from non-server OSes. Detecting botnets on one's own network, conversely, is getting harder, since botnet authors have started using "hide-and-seek" bots that deliberately void the high-traffic fingerprint that usually allows easy identification.

We also discussed the degree of customization granted to the end user. Although most attendees thought that allowing significant user-level customization of spam filters was ideal in theory, it is often not feasible, especially for high-volume sites, and the difference in effectiveness may not be as large as one might think.

Sender Policy Framework (SPF) and Domain Keys (DKIM) were discussed and discounted as reasonable antispam measures, although they might be useful in combination with other tools. Many spammers have embraced SPF and DKIM, which has reduced their reliability, and slow uptake by major corporations has reduced it further. Even their usefulness as antiforgery devices was debated, since they break mail forwarding as it is normally done.

Greylisting, the hot new technology from the past two years, is slowly but surely waning in effectiveness. One site reported that the effectiveness of greylisting has dropped from 21% of incoming mail dropped to only 12% in the past year.

As greylisting wanes, sender verification waxes, but it is beset by major technical difficulties. The high overhead makes it difficult to justify at high-traffic sites, even with aggressive caching. More important, spammers can use sites that perform sender verification as proxies to verify their own lists of addresses or, more nefariously, as proxies to run a Denial of Service attack against a common mail server. It was generally agreed that the potential damage one could do by enabling sender verification was not worth the benefit, which was itself high-cost.

Another high-cost antispam solution is tarpitting, which is also growing in popularity to fill the void left by greylisting. Tarpitting has roughly two forms: slowing the connection or pausing it. Either way is quite expensive, since it requires an open connection to be left open. There are several promising options, though, for reducing the cost of tarpitting. Some attendees suggested creating a purpose-built tarpit device that would pause connections, consuming resources only on that machine, and then pass the connection off to a real MTA when tarpitting was complete. Others felt it was reasonable to only tarpit after some suspicious activity was detected, perhaps as an alternative to rejecting mail that's only marginally suspicious.

Looking to the future, reputation services, including the recently launched KarmaSphere, promise to be the next must-have technology for fighting spam. At the moment, the field lacks much-needed standards, but several draft RFCs describing the SIQ standard for reputation services aim to plug this hole. Centralizing reputation services should provide a significant boon to mail administrators who have heretofore been forced to make binary spam-or-not decisions on multiple blacklists, whitelists, etc., individually.

Interestingly, we found that although spam was still a major, growing problem, email viruses had virtually disappeared. Virus writers have mostly relocated to the malware sector, where the means of transmission is generally the Web, not email. One site reported detecting fewer than 20 incoming viruses in the past eight months.

In the final segment of the workshop, we turned to one attendee's very specific issue of performance in a very high-capacity environment. Many of the spam technologies we had discussed were resource-intensive, and he needed to limit resource consumption in his 10-million-mailbox environment.

The common approaches of rejecting as many messages as possible and splitting inbound and outbound mail had already been tried, but these were insufficient. Other suggested performance tweaks were to put mail processing entirely in memory or to use machines capable of high numbers of concurrent threads.

From there, we discussed using an automated firewall management tool, such as FUT or fail2ban, to block connections from repeat spammers and create a very nimble, site-specific blacklist of sorts. This would also save the overhead of accepting connections from known spammers.

Other attendees recommended using a set of high-cost MXes as a sort of honeypot; those machines would get very slow, but this would mostly inconvenience spammers. The traffic they took away from the real MXes would allow legitimate mail more resources. As an illustration, the mail to one site's primary MX was only 86% spam, whereas the higher-cost backup MX received 98% spam.

The last solution suggested was to use an ever-growing cluster of cheap appliances. With the attractive price-point of many appliances, it could be feasible to throw enormous amounts of hardware at the problem of spam filtering in a very large environment and, by using appliances, make it essentially someone else's problem. Unfortunately, the workshop attendees couldn't agree on any appliances that had worked well across the board. Every appliance or commercial service that had worked well for one attendee had invariably worked dismally for another attendee, demonstrating the extreme variability of spam by site.

■ *MicroLISA*

Robert Au

Summarized by Ski Kacoroski (kacoroski@gmail.com)

The first MicroLisa workshop was held on Sunday, November 11. This workshop is aimed at the unique problems of sites with a limited number of staff, which means that each admin has a unique skill set, cross-training is very limited, and there is no time to specialize in storage, clustering, or other technologies. The goal of the workshop was to identify the unique problems of small sites, develop best practices, and determine how to get more of the larger community to address these issues.

Sites represented included a secondary school district, a few colleges, a computational R&D center, a small ISP, and a few startups. Some sites were standalone, whereas others were part of a larger organization from which they could get some support.

Our first discussion was on emergency and vacation coverage issues. How do you provide service when you are gone at a conference or on vacation? How can you balance your private life and the demands of work? One idea is to create the illusion of a big help desk by setting up an auto-reply message. Other ideas were to push back on management to set more realistic service levels, work to make systems more reliable to avoid pages, use service contracts to push the problems onto an external source, and rotate the pager among other IT staff to screen out noncritical issues. In terms of vacation coverage, options ranged from no coverage, people checking their phones once a day, or hiring an on-call consultant.

Our second discussion covered tools we used to monitor our systems. Nagios was the most common tool used with pages being sent to cell phones. Other tools were Intermapper, OpenNMS, and home-grown scripts. People seemed to be pretty happy with the way their monitoring tools were working. Many noted that scripts sending email were also primary monitoring tools and that it wasn't so much what was in the emails, just that they received the emails (e.g., I expect three emails an hour from this machine). In other words, we learned patterns in our incoming email. A few people felt that tools such as Zenoss and Splunk had too much overhead when compared to Nagios, SEC, or emails sent from cron jobs. Small sites need very simple low-maintenance solutions.

We had a long discussion about configuration management. How do we manage configurations? How do we determine if the overhead of a configuration management tool is worth it? A few sites used Cfengine, but most sites could not justify the overhead of setting up a configuration management tool, because they were either very heterogeneous or had very rapid changes. A mix of home-grown scripts and ad hoc solutions was the most common. It was also noted that configuration management tools puts one more layer between the admin and the machine and requires additional training, which makes it difficult to implement at small sites. Decisions on when to implement a configuration management tool at a small site were based on (1) whether it would save time and money by allowing lower-skilled staff or customers to make changes rather than the system admin and (2) whether it would help in documenting the systems.

The next discussion covered how to get help, especially expert help in areas that we just do not have time to learn in depth. The key is to get some vendors you can trust and obtain the knowledge from the vendor so you can support the system. None of us had good answers to this problem. Most folks did not like to outsource entire services because they were on the hook when something went wrong with the outsourcer and they had been burnt in the past.

Funding and working with management were discussed next. How do we convince management that equipment refreshes are a good thing? Because of funding sources, some people have big budgets for capital but little budgets for labor. In this situation people can use redundant systems as a replacement for staff. Normal maintenance and equipment refreshes are typically hard to sell to management, although some groups who are part of a larger organization are able to get this done via a central administration policy. If you are at a small site, then the best bet seems to be to determine management's pain points and figure out ways to minimize the pain in return for getting some funding for critical items.

This led into a discussion on communication with the organization. We all agreed that it is important to get out with your users and see what works and what doesn't work for them. This also helps with planning for future projects. In addition, you might be able to have the users bring some pressure on management for funding critical projects.

Once you have funding, then you have to figure out what to purchase. How do you pick a vendor? How do you test that the equipment meets your needs? What can be done to assure the vendor performs as expected? The key here is to create relationships with a few trusted vendors. For larger systems, spend the time to do testing. For smaller systems, get recommendations from people you trust and just implement them.

We touched briefly on regulatory issues and, yes, they are affecting small sites. Who needs to understand regulations? Who cares or has the liability or time? One site refused to keep critical data, believing adequate protection would be impossible for it to maintain, but many sites do not have this choice. Another idea was to outsource critical data storage (e.g., use Paypal for credit card transactions). In addition to the regulations, special privacy concerns might be important (e.g., for the rich, those in witness protection programs, etc.).

The last discussion before lunch centered on asset tracking and whether we could use that data for other purposes such as IP databases. We felt that asset tracking databases were not accurate enough and had lots of exceptions, making them not particularly useful in the system admin context. What we need is an automated way to have the equipment update the asset management database. One attendee has the beginning of a lightweight system that does this. Tools used for asset tracking were spreadsheets, wikis, and an asset tracking module on a helpdesk system.

After lunch the first topic was storage. What kind do we use? What are our criteria for picking storage? DAS is the most common at small sites because of low cost and the low level of skill needed for its operation, but it does not provide management tools. What we would like is a tool that will scan a network and map out the storage, shares, and mounts (especially when a person is just starting at a job). SATA disks were deemed to be as good as SCSI for almost all applications, but there were some concerns about costs (both training and maintenance) and the idea of putting all services onto a single storage device (NAS or SAN).

The storage discussion led nicely into a discussion about backups and disaster recovery. Most folks still use tape, although some are looking at remote disk arrays as their storage grows into the 20-TB range, because tapes are too labor-intensive. People who are part of larger organizations often make use of the resources of their parent organization.

The next discussion concerned how to train a new person. If you have the time, a good way is to have the new person do an audit of all machines and services. If not, have them shadow you for a few days or weeks and then start giving them small projects to work on. This led to how we learn and the resources we use to solve our problems. The most common learning process was to poke at a system (typically in production) and hope it doesn't break, as we do not have resources for spare test systems.

Occasionally people would have time and an extra machine to work on, but often that is not the case. For getting help people used Google, mail lists, IRC, Webcasts, LISA, and the LOPSA tool page. The biggest problem we all had was how to find the good information in the deluge that we face all day (e.g., which of the 17 million books on Exchange is the good one?). A dream would be a clipping service that would have some intelligence to determine which articles should be passed on to the subscribers. Perhaps we could use blog aggregators with tags to determine good content.

The discussion then led into time management. The biggest problem for all of us was getting large enough blocks of time on a regular basis for project work, because of the daily fires we have to put out. Keeping a log of what you do helps; so does a good ticket tracking system (with RT and OTRS being the most common).

How we document and plan was next on the agenda. Wikis are the most common documentation tool. The key is to document at the correct level, which means not a step-by-step procedure, but instead a summary that assumes the reader has a basic level of competence in the subject. The biggest issue with wikis was being able to easily control access to its pages. Planning is often very ad hoc, although some sites have a budget cycle (e.g., four years for a school district) or can use trend graphs to predict what new equipment will be needed.

The last discussion centered on the unique characteristics of small shops and whether there is a good way to define them. Small shops tend to have more budget limitations, which leads them to use more open source software. Small staff size leads to many other key issues, such as time management, knowledge depth, lack of specialists, and the need to rely on consultants. Many of the tools discussed at LISA are difficult to implement at small shops because of the time necessary to learn, implement, and maintain them even though they would save time once in place. We definitely need to figure out more multipliers to make better use of our skills. Ideas were to use students, interns, and consultants.

We wrapped up with the notion of creating a mail list dedicated to discussion of small site issues and to explore the idea of a MicroLisa column in a magazine or on a Web site.

- ***Configuration: From Managing Nodes to Managing Architecture***

  Mark Burgess, Oslo University College; Sanjai Narain, Telcordia Technologies, Inc.

  Summarized by Anu Singh (anusingh@cs.sunysb.edu) and Mukarram Bin Tariq (mmtariq@gmail.com)

Sanjai Narain highlighted the important role configuration plays in keeping networked systems up and running, along with the general lack of formalized tools for automating configuration to assure end-to-end communication requirements. He thanked the participants for attending the

workshop and sharing their views, work, and knowledge on this subject.

Sanjai also reminded the participants of the upcoming deadline for the special issue of the *Journal on Selected Areas in Communications* (*JSAC*) on configuration; the deadline is March 1, 2008.

- ***Verification and Adaptation of Network Security Policies***

  Ehab Al-Shaer, DePaul University

Ehab talked about management of security policy configuration, a complex issue because of the many rules required—which are written in 5-tuple forms and have complex semantics—the presence of distributed devices, and distributed policy interactions. This means that often not all configurations can be checked ahead of time, leading to potential security breaches. Ehab provided an overview of existing set-theoretic formalizations of intrafirewall conflicts in distributed environments and provable sets of constraints that are sufficient. He pointed to some limitations of this approach. Ehab introduced an approach based on BDDs (Binary Decision Diagrams) and contended that BDDs are a more effective way to formalize the rules; he showed how conjunctions and disjunctions of rules are expressed using BDDs. He also showed how IPSec policies can be represented using BDDs and how one can verify policy conflicts for paths and compositions. Ehab presented how to use BDDs to model routing configurations. The developed tool is available for download from Ehab's Web site.

Question (Mark Burgess): Have you considered using ontology or description logic?

Answer: We tried a little bit, but BDD is a mature area and we can leverage on its maturity.

Question: How useful would a description logic be instead of BDD for this work?

Answer: It could be more expressive, but the vast literature and research already done on BDD means that we can leverage on the success of BDD as a proven tool.

- ***WISE: Predicting Service Response Times in "What-If" Deployment Scenarios***

  Mukarram Bin Tariq, Georgia Tech

Mukarram presented a tool for evaluating the effect of response time distribution for hypothetical deployment scenarios for content distribution networks. The deployment scenarios include deployment of new data centers, changing DNS mapping for subsets of clients, and having new peering with a new ISP. For networks such as content distribution networks, no accurate model for response time exists, and it is generally hard to develop such models, owing to the scale and complexity of the system. Mukarram showed how machine learning can be used to model response time as a function of variables that can be easily observed in existing deployments; further, he presented how interactions among the observed variables can be captured in the form of a causal Bayesian network and be subsequently used to evaluate "what-if" scenarios. The What-If Scenario Evaluator (WISE) includes a high-level, declarative specification language called WISE-SL, which the network designers can use to express in a very succinct manner the scenarios they wish to evaluate. Mukarram also presented results on accuracy and effectiveness of WISE predictions based on dataset and events observed in Google's Web search service delivery network.

Question: How is the WISE approach better than doing, say, NS simulations?

Answer: Generally, it is difficult to make accurate simulations for the kinds of large, complex systems that we are talking about here. Trace-driven simulations can help somewhat in terms of input to the system, but still we need to model the system accurately, which is hard. The WISE-based approach is good in the sense that it does not require explicit modeling of the system.

Question: To what network scenarios can WISE be applied?

Answer: The cases where there are no hidden variables that can affect a scenario can be easily evaluated with this approach.

- ***MulVAL: A Logic-Based, Data-Driven Enterprise Security Analyzer***

  Xinming (Simon) Ou, Kansas State University

MulVAL presents a logic-based approach for security analysis in multihost networks. The formal definitions of security risks from OVAL and the National Vulnerability Database (NVD) serve as input (base data) to the MulVAL tool. The interactions within a network are formulated as rules in Datalog. Rules are completely independent of the network setting and are generic enough to be applied to any network. The tool can detect multistage attacks in multihost networks. The cause of an attack is represented through an attack graph.

Question: How is analysis done over multiple stages in the network?

Answer: The interaction rules are written in Datalog and multistage attacks are formulated in Datalog using recursion.

Question: How fast is MulVAL analysis?

Answer: All the interaction rules for Linux are written using approximately 20 Datalog rules. The time it takes to perform the analysis is quadratic in terms of the machines in the network. For large networks the analysis takes a few seconds and generation of attack graphs takes about 1–2 hours.

- ***Maestro: A New Architecture for Realizing and Managing Network Controls***

  T.S. Eugene Ng, Rice University

There is a lack of interface and abstraction for coordination among protocols. Eugene proposed an OS that over-

sees the whole network. The idea is to insert safety-checking applications in the network to look for network misbehaviors, such as routing loops and security breaches. The proposed OS for a network runs the logic of routing, and routers only perform forwarding. Further, a "BIOS" is required for the network. Maestro is an operating platform that provides an API to the applications and an interface for network state exchange (BIOS). Routers essentially work as "sensors" that measure the state of the network; the state is presented as Network Views and gets passed to the applications. Applications are stateless functions that can form an Application DAG (a sequence of applications). The Application DAG is triggered by some prespecified triggers.

Question: How does Maestro contrast to InfiniBand?

Answer: We have not specifically contrasted with their approach.

Question (Burgess): Is it tailored somehow to BGP?

Answer: An OS controls one administrative network.

Question: What about security implications as well as delegations?

Answer: We are looking into that.

- ***Request and Response Monitoring of Online Services***

  Juhan Lee, Microsoft

The goal of this project is to monitor request and response streams for massive online services. Typical monitoring systems do not scale to the requirements of large, complex systems such as MSN online services, where there are thousands of servers in the network.

In the presented scheme, an application overlay network is used for request-response exchange. A token-based approach is used for scheduling requests. A token is generated when a new request arrives. The generated token is passed along the servers serving the request and application-specific logs are generated. The logs of the requests and their responses are enormous. It is difficult to store such large logs. Conditional logging is used to reduce the storage requirements. Correlated sampling is used to lower the sampling rate for request-response. Lee showed a couple of examples of scenarios where their technique was able to use the logs to point out problems in the services, in particular a case where information was being served in an incorrect language at an international portal of MSN services.

Question: Is correlated sampling done among domains or across the entire network?

Answer: Sampling is done across the entire network and all the components. It allows us to pinpoint what's taking the most time. In the MSN publishing platform (portal), multiple requests (asynchronous requests) are sampled in a single session. The MSN publishing platform is incrementally deployable.

Question: What configuration errors are detected?

Answer: Generally, misconfigurations that can lead to unexpected application behavior can be detected.

- ***Panel on Security Configuration***

  Moderator: Steve Bellovin, Columbia University; attendees as panelists

Configuration is important for firewalls, depends on what services are being run and on whether the service set is contingent on the versions or patch levels, and has implications for authorized parties changing configuration and how one manages the authorization list.

There are various security scenarios to be considered. (1) Appliance (firewalls, filtering routers, etc): What should the configuration be in a complex topology? Typical corporate networks have several entry points into the networks. How do we reconcile different policy needs? (2) Infrastructure: Do the infrastructure nodes have proper security configuration? How do you know whether some element's configuration is wrong? (3) Servers: What services are they running? What versions? How do you monitor changes, new nodes, etc.? (4) Personal machines: How do you balance personal needs versus corporate needs? How do you enforce or prevent upgrades? How do you change the configuration of a laptop or home computer? How do you balance the need for central control with what cannot be enforced?

Mark Burgess asked whether control is the same as security, especially if it is not enforceable. Steve and other participants noted that exerting control is a way to enforce security; it is necessary but not sufficient. Andrew Hume equated it with ensuring border security and immigration issues. Mark remarked that we should think of security in civil society, where if most of us agree to abide by a law, then it makes it easier to enforce it.

Steve directed the participants to focus back on configuration issues of security. Assuming a certain policy of controlling, how do you enforce it? One of the participants remarked that there are various constraints, which may not be overspecified, and certain things are left free. How do you compose such constraints? Steve asked whether there is sufficient homogeneity in configuration and devices to allow such compatibility.

The ensuing discussion again drifted toward the inconvenience that security poses and causes people to "disable security." The problem arises because of mismatch of values between the system administrators and the users of the network. One of the participants remarked that usually there is a gap between what a human administrator wishes to achieve and what gets translated into configuration.

Ehab observed that a necessary component that is usually not explicitly evaluated in configuration management is risk assessment; if configuration management is integrated with such assessment, it will lead to consistent and sensible configurations.

Steve Bellovin asked the participants to consider the question of how to introduce new technologies and how to deal with new applications in terms of configuration. Paul Anderson affirmed the need to state goals and criteria, for the criteria, not the specific images of OSes, are what we wish to enforce. And we do not want to specify everything else. Ehab remarked that, generally speaking, creating a program from a specification is not solvable; are we going that route? Andrew commented that because of this problem, we are stuck with configurations that work, or a certain subset of use cases that have been tested.

Mark believes that it should not be so hard to manage the variances if we are not afraid of sophistication, specifically, dealing in probabilities. We tend to operate in paranoid mode, where we want to address issues that are highly improbable, which leads to complex security configurations that are difficult to work with. Ehab also asked whether there have been any incidents of remote malicious configuration change.

Steve Bellovin shepherded the audience back to the topic, refocusing on two issues: (1) keeping track of only the authorized users and what changes they make (i.e., managing the list of users and what they are allowed to do); (2) what to do when someone breaks in through a hack.

Sanjai added that composability is important for abstraction. A declarative approach and the inference engine will allow us to verify whether two rules are in conflict. Mark mentioned IETF BDIM WG activity, which is looking at how to convert high-level goals into low-level policy.

- ***Automata-Driven Techniques for Managing Firewall Configuration***

  Alok Tongaonkar, Stony Brook University

Alok talked about optimizing the performance of firewall-rules analysis using automata-based techniques. He talked about syntax- and semantics-driven analysis techniques for firewall configuration analysis. Rule interaction makes the analysis difficult. Alok mentioned that the BDD-based techniques proposed by Ehab (the first session) are semantics-driven.

Alok told how a finite state automata (FSA) is built for packet classification. The enormous state space of a packet is divided into finite regions of the automata. Packet space is divided into regions based on their match with the firewall rules. Priorities of rules can govern the classification (i.e., application to a packet). He also discussed shadowing of rules. The rules are analyzed, and intersections and shadowing among them are found. FSA size explosion is possible because of duplication of rules that may occur from ranges and "less than" and "greater than" occurrences in the rules. Their algorithm minimizes the duplication of rules. The FSA is built incrementally using candidate (probable) and match (matched) sets, resulting in a compact automata.

Sanjai: Why do this kind of analysis? Why should we not just build right configurations to begin with?

Answer: "Evolution" of rules may inadvertently result in conflicts and misconfiguration.

Steve: Should we have a better abstraction than priority-based mechanisms?

Answer: We are trying to convert priority-based rules into nonpriority-based rules; however, this results in an explosion of the number of rules.

Steve: Is a human factor involved here and does the explosion make it more or less comprehensible?

Answer: It is not clear at this stage.

Sanjai: Are these rule lookups $O(n)$ and are there really that many rules?

Answer: Yes; for priority-based rules it has to be.

Steve affirmed that there are indeed many rules for security configuration and referenced a study by Arbor Networks.

- ***Vulnerability Analysis via Deductive Spreadsheets (DSS)***

  Anu Singh, Stony Brook University

Anu explained the desired properties of a security policy analysis tool. She explained the Role Based Access Control (RBAC) model as background. The current prototype implementation of DSS, called XcelLog, is built as an add-in to MS Excel. The formula language of DSS supports sets and tuples. Recursive relations can be represented by using DSS. XcelLog uses XSB (Prolog) as the underlying evaluation engine. The features of DSS include highlighting of explanations for results and incremental evaluation. Anu gave a demo of the DSS using the RBAC example. She also showed how to do vulnerability analysis for a multihost network in DSS.

Question: If there is more than one way of getting to a condition (attack), will DSS be able to highlight?

Answer: The tool can find multiple ways of getting to a condition (attack), but it may not be able to distinguish among them.

Question: What are the relations between DSS cells and prolog predicates?

Answer: DSS expressions represent logical conditions using cell references. Anu explained with a demo.

- ***An Analysis of the VLAN Design of an Operational Campus Network***

  Yu-Wei Sung, Purdue University

Yu-Wei talked about generation of task-driven network-wide abstractions to configure enterprise network design. He emphasized the need for abstraction of a network design by elevating the observations to abstraction that would simplify the design. It is important to understand

the intent of the operator, he added. VLAN configuration is error-prone and time-consuming, which is why abstraction of the VLAN is useful. VLANs can be abstracted by logical grouping of hosts. The key components in a VLAN configuration are the access port and trunk ports not directly connected to hosts serving as carriers for other VLANs. Abstractions model the network as a topology of hosts and switches, and a router placement strategy determines where to place the routers.

Question: How does the firewall interact with VLAN?

Answer: This hasn't been considered yet.

Question: What data size is used?

Answer: A single network with 1300 switches.

Question: Can the tool generate configuration consistent with the abstract model?

Answer: It can generate useful recommendations that can help the operator in network design.

■ ***Fixing Configuration Errors via Unsatisfiability Analysis***

Sanjai Narain, Telcordia Technologies, Inc.; Daniel Jackson, MIT; Sharad Malik, Princeton University

Security cannot be divorced from functionality. Since there are usually so many interweaving requirements, we can put all the various requirements in a melting pot and generate a configuration from that. The specific problem addressed in the talk was fixing configuration errors via unsatisfiability analysis. Sanjai explained, through an example, how security and reliability are interrelated, where if a separate IPSec tunnel is not established for the backup router, the communication would break down even if the backup router took over.

Sanjai discussed how to specify security, routing, and reliability in a single unified framework and how to do it efficiently. He presented a requirement solver that takes as input a specification in first-order logic and the configuration variables database and produces the configurations. ALLOY, a first-order language, is used for specifying the constraints, which are solved using SAT solver. Once the requirements are, at a high level, expressed as FOL constraints, the Un-SAT-core (unsatisfiability) finds the subset of the sets that are unsatisfiable. A counterexample can be obtained from the result of the constraint solver. The next step is to find the configuration variables that violate the constraint and then relax the constraints and re-solve it.

The issue of scalability arises if the constraints are specified in an obvious way. The aim is to scale this. The requirements are preprocessed and constraints are partially evaluated using the constraint SAT solver. This makes it practical to apply to large network configurations. The system uses a deductive spreadsheet to specify constraints on cell values and display results.

Question: How helpful is this analysis for network configuration management?

Answer: Host-level configurations can be modeled and analyzed using this framework.

Question: What about constraints crossing the layers of the protocol stack (application layer versus the network layer)?

Answer: The solver can capture all dependencies, including cross-layer dependencies.

Question: Does the solver give the best answer or just an answer?

Answer: Presently it gives an answer. If the notion of "best" can be formalized as a constraint, then it can give the best answer.

■ ***Panel on Autonomic Configuration***

Moderator: John Strassner, CTO Motorola; attendees as panelists

John Strassner gave a brief introduction to autonomic networking, which he described as the process of offloading simpler automatable things to automated processes. He contended that operators are losing money because the OSS (Operations Support System) is too complicated to address business and customer needs. He said that we want to build something that takes care of autonomic functions, so that the human in charge has less to do. We want the system to "learn" how to do everyday things. He gave an overview of the system being built by his group. The system uses machine-learning techniques. [Editor's note: Strassner's talk was similar to his keynote.]

Andrew: Autonomics are applicable to simple and very well-defined tasks, such as breathing or heart pumping.

John: If there is a set of transformations that can take the service and business goals, along with the environments, and give out CLI commands and configuration, then this is autonomics. Our research is about merging ontology with CLI-level stuff. It is like a "multigraph." At the higher level of the graph there is a different interpretation of "errors" than at the lower layer; the following steps are involved in such a system: IOS → Model-based translation → ACML → Analyze data and event and determine actual state → If not desirable → Reconfigure → Repeat.

Andrew: Telephony built a similar network to map errors to the customer, but this mapping has to be dynamic because the relationships are fluid. We live at a 99.99% uptime level, but that does not have too much bearing on how the operator is doing at a business level.

Sanjai Narain: The emphasis in this work seems to be on performance; aren't things like security configuration just as important? And in that case, how do you see the system learning its state and changing the configuration?

John: Learning that kind of semantics and subsequent configurations can be hard but, given higher-level goals, it can be achieved.

■ *Advanced Topics Workshop*

Summarized by Josh Simon (jss@clock.org)

The Advanced Topics Workshop was once again hosted, moderated, and refereed by Adam Moskowitz. We started with cable management 101 in separating the large bundle of CAT-5 cable into strands for us to connect our laptops to the local network switch, as there are enough of us that we overload the wireless access point. We followed that with introductions around the room. For a variety of reasons, several of the Usual Suspects weren't at this year's workshop. Despite this, in representation, businesses (including consultants) outnumbered universities by about four to one; over the course of the day, the room included six LISA program chairs (four past, present, and future; up from three last year) and 11 past or present members of the USENIX, SAGE, or LOPSA Boards (up from five last year).

Setting up involved untangling the bundled CAT-5 cables, connecting them to attendees' laptops and the local switch, getting the moderation software up and running, setting the correct time zone on the server, and so on.

Our first topic was on management versus technology. About half of the room were either full- or part-time managers, and we discussed some of the problems we have interacting with our management. Some of the concerns were knee-jerk, too-shiny managers, cultural differences when your manager is in another country, and managers who used to be in sales positions. Some folks discussed their specific situations and asked for advice in solving them. One common suggestion was to communicate differently; remember that managers (especially those on the financial side who approve capital budgets) tend to speak business-speak and not "techie." They don't care about the new gee-whiz neato-peachy-keen technology but, rather, in how this new thing will solve their problems and provide a decent return on investment.

A side discussion took place on cultural issues that differ from the North American standard most of us are used to, and how that can affect communication styles as well as resumes.

After the morning break, we discussed career concerns. Most of the people in the room had 15 or more years of experience, and many of us had more than 20 years of experience. Assuming that retirement isn't an option (for whatever reason, be it financial or boredom), what's the right thing to do if you wind up looking for work? One person discussed how he neglected to ask questions of the company during the interview process; after accepting the offer and working for some length of time, he realized he was a bad fit for the position. One suggestion for avoiding this in the future was to ask better questions before accepting any offer; another suggestion was to consider a contract-to-permanent position, since it gives both parties an out without the company having to let a senior person go. One topic that fell out of this is whether there's a technical growth path at your company or whether "senior" implies a management position. Another topic was the technology refresh rate for individuals and whether staying generalists or becoming specialists was the better course of action. (Consensus seemed to be for the former.) Those who are retiring in the fairly near future have to make peace with what's good enough and remember that "good enough" isn't necessarily the same as settling. Do what needs to be done and find enjoyment in that. Whatever you're doing, remember to work, to play, and to live, not just to exist. Do things to keep yourself interested and awake at your job; don't just settle into a rut.

We next discussed patterns as an abstraction layer in system administration. There's apparently some controversy in patterns; some think they're a good way to abstract problems and provide a common shorthand; others think they're not worth the electrons and doubt they're applicable to system administration.

After our lunch break, we discussed things we should have done differently. One example was the whole IPv6 rollout. A lot of places don't see any need to deploy it and wonder whether ignoring it now will cause problems later. CFOs don't see the benefit from or ROI in another technology refresh. Widespread adoption of something new requires there be some kind of benefit on the business level, and right now businesses don't tend to see a need for IPv6. Right now, there is very little IPv6-only content out there; if services or content were made IPv6-only, that could drive folks to convert, assuming that their equipment is IPv6-capable (e.g., not all SOHO equipment is).

We next had a brief discussion on the different uses of DNS and search engines. Both are treated as a way to find resources: DNS is a way of finding IP address-to-name mappings and search engines are a way of finding some specific document or site.

We next went around the room to discuss our favorite new-to-us tools from the past year. Common examples were AFS and ZFS, certain KVM cards, load balancers, ntop, Puppet, Ruby and jRuby, spam management tools, svk, tcptrace, the iPhone, Time Machine in Mac OS 10.5, virtualization, and zones in Solaris. Several people chose Puppet for their configuration management, mainly because it was faster to get it up and running and it had a sufficiently low learning curve for installation and configuration.

Our next discussion was on virtualization. At least one participant has done nothing with it and wondered if it was worthwhile; the consensus seemed to be that there are some areas where it's not a benefit, such as in a high-per-

formance computing environment. Someone plugged this year's refereed paper, "Decision Support for Virtual Machine Re-Provisioning in Production Environments," by Kyrre Begnum, Matthew Disney, Æleen Frisch, and Ingard Mevåg, for performance statistics. Some are only using virtualization in nonproduction environments.

We next talked about delegating identity management. The example environment is a department within a university that uses automated identity management to manage authentication and authorization controls for students, faculty, staff, alumni, and guests (e.g., investigators on research grants from other universities). The central IT organization can provide some of the information they need, but not all of it. The question of how they can cascade information management systems was addressed. The short answer is that processes need to be put into place to incorporate the data to allow for both provisioning and revocation and to make sure essential safeguards are in place such that a failure upstream (e.g., HR's database failing miserably) doesn't accidentally cause a disaster downstream (e.g., the deletion of all staff accounts).

The next major discussion concerned distributed or geographically disparate personnel. We talked about what server infrastructure needed to be used in remote data centers for part-time workers; the answer generally depends on how many people and how much network traffic there is. Items to consider are VOIP or POTS phones, home directory access over the WAN, docking stations, printers, reserved offices or cubes ("hoteling"), and possibly a conference room, depending on the size of the office and the data center. We also talked about tools for communication and collaboration; many use some form of instant messenger client (such as internal IRC channels or logged IM conversations), email, trouble ticketing systems, and wikis. If you are using any of these technologies, remember to include them in (as a critical part of, where need be) your disaster recovery planning.

Our final discussion was a quick around-the-room on the cool tool for next year. Ruby, Solaris 10, and virtualization were the most commonly mentioned, with configuration management tools, Perl 6, VOIP, wiki deployment, and ZFS rounding out the list.