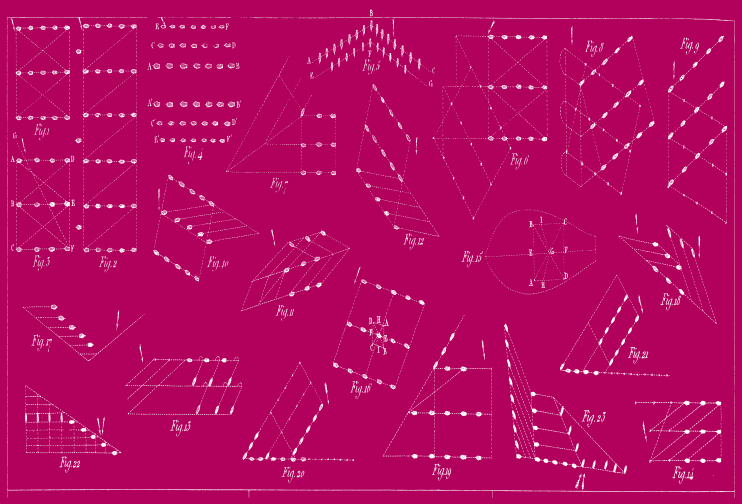
JUNE 2010 VOLUME 35 NUMBER 3

;login:

THE USENIX MAGAZINE



USENIX
The Advanced Computing Systems Association

USENIX Upcoming Events

19th USENIX Security Symposium (USENIX Security '10)

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

2010 Electronic Voting Technology Workshop/ Workshop on Trustworthy Elections (EVT/ WOTE '10)

Co-located with USENIX Security '10

AUGUST 9–10, 2010, WASHINGTON, DC, USA
http://www.usenix.org/evtwote10

3rd Workshop on Cyber Security Experimentation and Test (CSET '10)

Co-located with USENIX Security '10

AUGUST 9, 2010, WASHINGTON, DC, USA
http://www.usenix.org/cset10

4th USENIX Workshop on Offensive Technologies (WOOT '10)

Co-located with USENIX Security '10

AUGUST 9, 2010, WASHINGTON, DC, USA
http://www.usenix.org/woot10

2010 Workshop on Collaborative Methods for Security and Privacy (CollSec '10)

Co-located with USENIX Security '10 and sponsored by USENIX and Deutsche Telekom

AUGUST 10, 2010, WASHINGTON, DC, USA
http://www.usenix.org/collsec10

1st USENIX Workshop on Health Security and Privacy (HealthSec '10)

Co-located with USENIX Security '10

AUGUST 10, 2010, WASHINGTON, DC, USA
http://www.usenix.org/healthsec10

5th USENIX Workshop on Hot Topics in Security (HotSec '10)

Co-located with USENIX Security '10

AUGUST 10, 2010, WASHINGTON, DC, USA
http://www.usenix.org/hotsec10

Fifth Workshop on Security Metrics (MetriCon 5.0)

Co-located with USENIX Security '10

AUGUST 10, 2010, WASHINGTON, DC, USA
http://www.securitymetrics.org/content/Wiki.jsp?page=Metricon5.0

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

Sponsored by USENIX in cooperation with ACM SIGOPS

OCTOBER 4–6, 2010, VANCOUVER, BC, CANADA
http://www.usenix.org/osdi10

Workshops co-located with OSDI '10:

Sixth Workshop on Hot Topics in System Dependability (HotDep '10)

http://www.usenix.org/hotdep10

Networks, Systems, and Computation (NetEcon '10)

Sponsored by USENIX in cooperation with ACM SIGecom

http://www.usenix.org/netecon10

2010 Workshop on the Economics of 5th International Workshop on Systems Software Verification (SSV '10)

http://www.usenix.org/ssv10

Workshop on Managing Systems via Log Analysis and Machine Learning Techniques (SLAML '10)

http://www.usenix.org/slaml10

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

Sponsored by USENIX in cooperation with LOPSA and SNIA

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

9th USENIX Conference on File and Storage Technologies (FAST '11)

Sponsored by USENIX in cooperation with ACM SIGOPS

FEBRUARY 15–18, 2011, SAN JOSE, CA, USA
http://www.usenix.org/fast11
Submissions due: September 2, 2010

8th USENIX Symposium on Networked Systems Design and Implementation (NSDI '11)

MARCH 30–APRIL 1, 2011, BOSTON, MA, USA
http://www.usenix.org/nsdi11

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

contents

;login:
VOL. 35, #3, JUNE 2010

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

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

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

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

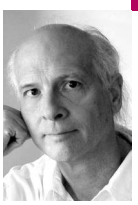
$90 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $125 per year.

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

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

RIK FARROW

musings



Rik is the Editor of *;login:*.

rik@usenix.org

IN MY FEBRUARY 2010 COLUMN [1], I took a humorous look at the future of computing. I let my imagination go wild, as my candidate for the future of system administration, Chuck, dealt with the various issues that arose during a workday. Of course, the picture really isn't as rosy as I made it out to be. For example, the organic, in-wall system might take at least 20 years before it becomes practical. And if a system with equivalent power were built using today's technology, Chuck would have been quickly roasted—that is, if the wall itself didn't melt down first.

You have likely heard of the walls of power and memory that current system and CPU designers are facing. Clock speeds are not getting much faster, as faster clock speeds mean more power dissipated as heat. At 120 watts per centimeter squared, Chuck's office wall would need to radiate megawatts of power if each centimeter produced the equivalent processing power found in today's server-class CPUs. But, on a more practical level, just dealing with cooling one 120 watt chip is difficult enough without making big changes in how servers are designed.

The wall of memory continues because SDRAM has not improved in performance nearly as fast as processors have. And with multicore chips, there are now many processors sharing the memory bandwidth from the same SDRAM. System designers can optimize potential bandwidth by having multiple memory controllers and pathways to memory, and this does help some of the time. But the problem still exists: in programs that are memory intensive and that have poor spatial locality, CPU cores will be stalled waiting for memory.

There are actually two other walls that haven't been mentioned but are equally important when considering the future of system designs: cost and history. These walls are related, as we shall see.

Multicore

The current designs for overcoming the walls of memory and power require having more than a single core in each CPU. Each core can run at a lower clock rate, and this reduces the power required. And by slowing down the processors, they won't be stalled for as many cycles—but they will

still have to wait for memory. Clever cache design, having multiple threads backed by their own register sets, helps to hide memory latency by keeping the cores busy, as was done by Sun in their Niagara chips [2] and Intel with HyperThreading. But the problems of power and memory still exist.

Just designing chips with more cores is not likely to help. Intel's Nehalem Beckton (Xeon 7500) with eight cores, and AMD's Magny Cours [3] with 12 cores (but on two dies) use similar strategies to deal with the memory wall. Both have multiple paths to memory and lots of L3 cache (the Xeon has twice as much). I did some back-of-the-envelope calculations for the maximum memory bandwidth requirements for these chips, and, no surprise, memory cannot keep up. Of course, my exaggerated calculations don't match real-world applications. In the real world, memory access patterns—for example, does an application access only a small portion of memory that fits in cache or make almost random accesses across a large amount of memory?—are all over the board [4]. Faster clocks and more cores with more memory bandwidth do make a difference for high-end server applications.

But there are more problems with multicore designs today. On the power front, a helpful computer scientist pointed out to me that almost one-half of the power budget for current multicore chips goes to support, such as I/O, cache, and memory controllers. In a 120-watt chip, that leaves only 60 watts of power left for the cores themselves. In a 12-core design, that's 5 watts per core. If it were a 64-core design, there would only be about 0.9 watts per core—enough for relatively slow, in-order-execution designs like the Intel Atom, but not enough processing power for databases or High Performance Computing (HPC).

The same scientist mentioned that there are few applications today that can take advantage of more than four cores. Writing parallel applications is hard, as well as expensive, so only those applications, such as databases and specific HPC apps, are written to run in parallel. Unless and until there are good uses for more cores, having many cores will only be useful for the handful of applications that already support a lot of parallel execution.

Mixed Cores

Instead of today's multicore systems, I expect we will see more mixed cores. Sun's Niagara, Intel's Xeon, and AMD Magny Cours all have homogeneous cores—each core has the same architecture. Each core is a general-purpose processor, complete with floating point, integer, single-instruction, multiple-data (SIMD), and other capabilities. The design of core architectures is based on research into which general-purpose instructions get executed often and are worthwhile spending time and chip real estate to optimize.

Now imagine instead that code you will be running often has no floating point requirement in it at all. You could have processor cores without floating point support and focus on integer-only performance.

You might be thinking, "How silly is that?" but I did more than just think about it. I ran greps on the source code to the Barrelfish, MINIX 3, and Linux kernels, and lo and behold, kernels have very little floating point. None in Barrelfish and MINIX 3, and just support for emulating the floating point processor that was lacking in x386 chips and architecture support in the Linux kernel. It seems that kernel code, especially multikernels and microkernels, could run very well on integer-only cores.

Many userland utilities are integer-only as well. I took a look at the most recent version of Apache (2.2.15), and it turns out that several modules,

including mod_ssl, use doubles, as does the Apache Portable Runtime utility library. Perhaps with some work (and for sites that don't require SSL), you could run Apache without floating point support in hardware.

Going to the other extreme, some applications require a mix of floating point and integer operations. Scientific applications commonly require both [4]. But the GPUs used in today's graphics cards, and starting to appear as co-processors, are specialized, floating point-only pipelines. The Xeon 7500 includes a GPU not on the same die as the cores but part of the same package. Perhaps processing units from GPUs might also wind up being cores within a future multicore chip.

The Other Walls

I mentioned that besides the walls of power and memory, there are the walls of cost and history. I don't have any references for these walls—but I didn't invent them, just named them.

Drastically changing CPU architectures, for example, to support heterogeneous cores, would have an enormous cost. This cost would involve having to create new operating systems, compilers, and support libraries that would support the ability to properly use specialized cores. While an operating system may already run properly on an integer-only core, many other applications would not. With compiler and library support, applications could have integer-only threads that could run in parallel with the rest of the application, taking advantage of specialized cores. But doing this involves developing not just the new CPUs, but also the operating systems, compilers, libraries, and perhaps new languages (or language extensions) required.

Then there is history. Any change in the way CPU architects design CPUs means that the way programmers work, system administrators manage, and distributions provide packages would all have to change as well. If some organization designed a heterogeneous, manycore CPU that blew the socks off today's hottest (and I mean hot) multicore CPUs, the new CPU would be useless without a lot of software to support it. Just writing that software would require programmers who thought differently about writing applications—for example, having integer-only threads whenever possible.

Fortunately, we do have several groups working on operating systems and libraries that can support heterogeneous cores. Barrelfish [5, 6] is designed to do this, although that has not yet been tested. Both MINIX 3 and seL4 intend to support a multikernel design that might be able to work with heterogeneous cores. Groups at UC Berkeley (Par Labs), CMU, Google, and (likely) Microsoft have researched running on heterogeneous multicore CPUs. So people are thinking about this.

Going back to cost, going against history has a cost as well. In the obvious case, the cost is replacing decades of code development with something radically new and different. And then there is the cost involved because people who have spent years working with the existing paradigms would have to change as well. When you have spent your career promoting a particular way of thinking about something, accepting a big change may have a totally unacceptable cost for many people. The cost of change has held back many huge discoveries and technical improvements, whether the topic is the earth revolving around the sun, germ theory, or CPU design.

Whether having integer-only processing cores makes sense is really not the point. We have used CPUs designed for general processing tasks for some very good reasons. The first is cost, in that making a family of closely

related designs makes development and support (the compilers and libraries) cheaper. In the world of embedded systems, you can order systems-on-chip (SoC) with just the processing and I/O support features your application requires. But there will be extra cost for anything customized. Volume manufacturing drives price down quickly.

For now, I expect to see multicore CPUs with homogeneous cores. I don't expect to see cache-coherent systems with more than 16 cores (although you can already have quad-CPU systems with more than 16 cores). I say this because of the limitation on the amount of power available in single-die designs and the pressure on memory busses required by cache-coherency.

Designs like the Intel Single Chip Cloud use much simpler CPUs and dispense with cache-coherency in exchange for message passing. If you read the MINIX 3 article in this issue or read about Barrelfish in the April 2010 issue, you will see that message passing appears to be the way forward for multicore systems.

Lineup

Andrew Tanenbaum, with help from many co-authors, updates us on what is happening with MINIX 3. MINIX 3 is a modern operating system designed both as a platform for learning about OS design and as a real OS with a BSD-style license, with reliability, flexibility, and security highest on the feature list. I like what I read about the future directions of the MINIX 3 project. And where Linux (or Solaris or BSD) runs on every core in multicore CPUs, MINIX 3 not only has a much smaller code footprint but might someday require only some library code to run on each core, as Barrelfish does today (the dispatcher and CPU driver).

Jim Sangwine writes another article that looks to the near future. Sangwine's current research focus is on the use of 3D in Web applications. In his article, Sangwine looks at the leading contenders, Flash with the PaperScript library, WebGL, and O3D, using history as well as sample applications written using each library to test for performance and ease of programming.

Alva Couch takes us on a sociological safari into programming via ritual. Couch, like many of the older generation, tended to want to know everything about what they were doing—I certainly did. But today's programmers work much faster, taking advantage of Internet searches to find what works without needing to know *why* it works. Couch examines design patterns, examples, and rituals to distinguish these aids to programming, and he draws some conclusions about this new phenomenon.

Andy Seely takes us on a different journey, one into the realm of DoD sysadmin. Seely works for SAIC, and their customer needed a secure, easy-to-maintain (dead simple to maintain, really) and reliable replacement for a failed DNS server setup. Seely came up with a hardened Solaris VM that can be cloned, configured, and tested remotely, while being instantiated wherever needed just by using a VMware console.

Matt Simmons journeys back in time to LISA '09. Simmons shares his experience of attending his first LISA, and may perhaps encourage those who have yet to attend to head to San Jose for LISA '10.

Rudi van Drunen is back with a look at using small embedded systems. As an example, he focuses on Arduino, a small, inexpensive, open source platform that comes with an IDE for programming, lots of example code, and even plug-in modules (shields) for adding extensions to this simple processor.

David Blank-Edelman has written about libraries useful for creating dummy data. The article is not dumb, or just full of stuff, but demonstrates how you can create various types of test data quickly.

Peter Galvin shares his excitement about a new Sun/Oracle product, the Exadata V2. The Exadata V2 is both a database engine and a storage subsystem, but with a difference. Even the storage subsystem can carry out database operations, and the entire system can act as a transactional processing center and a data warehouse simultaneously—no mean feat. Delving into the hardware is part of what got me going about CPU futures (as did Tanenbaum's article).

Dave Josephsen writes about the use of packet capture tools for monitoring. Josephsen describes useful tools for capturing packets, including combining many sources into single archives using standard formats (flows and pcap), as well as one of my own favorite analysis tools, Argus.

Robert Ferrell covers his top five misconceptions about information security. In case you think he is just making this stuff up, he actually has special, super-secret techniques for collecting the information that becomes the ideas behind his columns. And no, he is not using Data::Generator.

Our main book reviewer, Elizabeth Zwicky, is taking a vacation this time. However, we have two excellent book reviews, one from Brandon Ching about a book on how organizations can use Second Life, and the other from Sam Stover giving us an in-depth look at a "short" (in Stover's terms) book about cloud security.

In conference reports, this issue covers FAST '10 and the SustainIT and TaPP workshops.

As regular readers of "Musings" may have noted, I fully expect the future of computing to be distributed. In the larger sense, this already exists in the forms of clouds, Web apps, Hadoop, and new frameworks such as Microsoft's Midori and .NET projects. But I also believe that the days of the homogeneous core processors are numbered, and I can only muse upon the possible futures of computing, from small embedded systems such as smart phones all the way up to high-end servers.

REFERENCES

[1] February 2010 "Musings": http://www.usenix.org/publications/login/2010-02/openpdfs/musings10-2.pdf.

[2] Richard McDougall and James Laudon, "Multi-Core Microprocessors Are Here," *;login:*, vol. 31, no. 5, Oct. 2006: http://www.usenix.org/publications/login/2006-10/pdfs/mcdougall.pdf.

[3] Nebojsa Novakovic, "Head to Head: Intel's Nehalem EX and AMD's Magny Cours," *The Inquirer*, April 6, 2010: http://www.theinquirer.net/inquirer/feature/1599367/head-head-intel-nehalem-ex-amd-magny-cours.

[4] R.C. Murphy and P.M. Kogge, "On the Memory Access Patterns of Supercomputer Applications: Benchmark Selection and Its Implications": http://www.sandia.gov/~rcmurph/doc/ToC_56_7.pdf.

[5] Barrelfish: http://www.barrelfish.org.

[6] Rik Farrow, "The Barrelfish Multikernel: An Interview with Timothy Roscoe," *;login:*, vol. 35, no. 2, April 2010: http://www.usenix.org/publications/login/2010-04/pdfs/roscoe.pdf.

ANDREW TANENBAUM, RAJA APPUSWAMY, HERBERT BOS, LORENZO CAVALLARO, CRISTIANO GIUFFRIDA, TOMÁŠ HRUBÝ, JORRIT HERDER, ERIK VAN DER KOUWE, AND DAVID VAN MOOLENBROEK

MINIX 3: status report and current research

Despite the fact that Andrew Tanenbaum has been producing open source code for 30 years as a professor at the Vrije Universiteit, somehow he found the time to (co)author 18 books and 150 papers and become a Fellow of the ACM and of the IEEE. He was awarded the USENIX Flame Award in 2008. He believes that computers should be like TV sets: you plug them in and they work perfectly for the next 10 years.

ast@cs.vu.nl

Raja Appuswamy is a PhD student at Vrije Universiteit. His research interests include file and storage systems, and operating system reliability. He received his BE from Anna Univeristy, India, and his MS from the University of Florida, Gainesville.

rappusw@few.vu.nl

Herbert Bos obtained his MSc from the University of Twente in the Netherlands and his PhD from the Cambridge University Computer Laboratory (UK). He is currently an associate professor at the Vrije Universiteit in Amsterdam, with a keen research interest in operating systems, high-speed networks, and security.

herbertb@cs.vu.nl

Lorenzo is a post-doctorate researcher at Vrije Universiteit Amsterdam, where he joined Prof. Tanenbaum and his team working on systems dependability and security. Lorenzo's passion for systems security was further inspired by work at UC Santa Barbara and Stony Brook University. Lorenzo received an MSc and a PhD in computer science from the University of Milan, Italy.

l.cavallaro@few.vu.nl

Cristiano Giuffrida is a PhD student at Vrije Universiteit, Amsterdam. His research interests include self-healing systems and secure and reliable operating systems. He received his BE and ME from University of Rome "Tor Vergata," Italy.

c.giuffrida@few.vu.nl

Jorrit Herder holds an MSc degree in computer science (cum laude) from the Vrije Universiteit in Amsterdam and will get his PhD there in Sept. 2010. His research focused on operating system reliability and security, and he was closely involved in the design and implementation of MINIX 3. He is now at Google in Sydney.

jnherder@gmail.com

Tomáš Hrubý has master's degrees from both the Charles University in Prague and the Vrije Universiteit. After graduating, he decided to go down under and spent some time at the University of Otago in New Zealand and NICTA in Australia. He is currently a PhD student at the Vrije Universiteit, working on how to match multiserver operating systems to multicore chips.

thruby@few.vu.nl

Erik van der Kouwe got his master's degree at the Vrije Universiteit and is now a PhD student in computer science there. He works on virtualization and legacy driver support.

vdkouwe@cs.vu.nl

David van Moolenbroek has an MSc degree in computer science from the Vrije Universiteit in Amsterdam, and is currently working as a PhD student there. His research interests include file and storage systems and operating system reliability.

dcvmoole@few.vu.nl

MOST PEOPLE WANT THEIR COMPUTER to be like their TV set: you buy it, plug it in, and it works perfectly for the next 10 years. Suffice it to say that current computers—and especially their operating systems—are not even close. We will consider the job done when the average user has never experienced a system crash in his or her lifetime and no computer has a RESET button. In the MINIX project, we are trying

to get closer to that goal by improving the reliability, availability, and security of operating systems.

What started in 1987 as MINIX 1, a tool to teach students about operating systems, has become MINIX 3, a more mature operating system whose internal structure promotes high availability while preserving the well-established POSIX interface to application programs and users. Although the name has been kept, the two systems are very different, just as Windows 3 and Windows 7 are both called Windows but are also very different. In this article, we will briefly describe the architecture of MINIX 3 and what it is like now—as an update to the February 2007 *;login:* article [1]—and the work currently in progress to develop it further.

The impetus for much of this work was a grant to one of us (Tanenbaum) from the Netherlands Royal Academy of Arts and Sciences for 1 million euros to develop a highly reliable operating system, followed four years later by a 2.5 million euro grant from the European Research Council to continue this work. This funding has primarily supported PhD students, postdocs, and a couple of programmers to work on the project, which has led to a series of releases, of which 3.1.7 is the latest one.

The MINIX 3 vision has been guided by a number of core principles:

- **Separation of concerns:** Split the OS into components that are well isolated from each other.
- **Least authority:** Grant each component only the powers it needs to do its job and no more.
- **Fault tolerance:** Admit that bugs exist and plan to recover from them while continuing to run.
- **Dynamic update:** Plan on staying up all the time, even in the face of major software updates.
- **Standards compliance:** Be POSIX-compliant on the outside but don't fear change on the inside.

We believe we have made a good start toward producing a general-purpose, POSIX-compliant operating system with excellent fault tolerance. As hardware speeds have shot up over the past two or three decades, we do not believe that most users really care about squeezing the last drop of performance out of the hardware. For example, in MINIX 3, a build of the entire operating system (about 120 compilations and a dozen links) takes under 10 seconds on a modern PC. Good enough.

If you get the MINIX 3.1.7 CD-ROM from www.minix3.org and install it, to the user it looks like other UNIX systems, albeit with fewer ported application programs (so far), since that has not been our focus. But on the inside it is completely different. It is a multiserver operating system based on a small microkernel that handles interrupts, low-level process management, and IPC, and not much more. The bulk of the operating system runs as a collection of user-mode processes. The key concept here is "multiserver operating system"—the design of the system as a collection of user-mode drivers and servers with independent failure modes. While the term "microkernel" gets a lot of attention—and microkernels are widely used in cell phones, avionics, automotive, and other embedded systems where reliability is crucial [2]—it is the multiserver aspect of the system that concerns us here.

The microkernel runs in kernel mode, but nearly all the other OS components run in user mode. The lowest layer of user-mode processes consists of the I/O device drivers, each driver completely isolated in a separate process protected by the MMU and communicating with the kernel via a simple API and with other processes by message passing. The next layer up consists of servers, including the virtual file server, the MINIX file server, the process manager, the virtual memory manager, and the reincarnation server. Above this layer are the normal user processes, such as X11, shells, and application programs (see Figure 1).

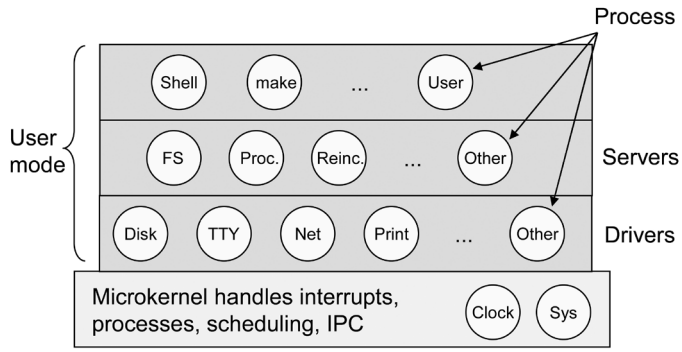


FIGURE 1: THE STRUCTURE OF MINIX 3

The reincarnation server is the most unusual part of the design. Its job is to monitor the other servers and drivers, and when it detects a problem, it replaces the faulty component (driver or server) on the fly with a clean version taken from the disk (or, in the case of the disk driver, from RAM). Since most errors are transient, even after a driver crashes (e.g., due to a segmentation fault after dereferencing a bad pointer), the system can, in many cases, continue without user processes even knowing part of the system has been replaced. We ran a fault-injection test in which 2.4 million faults were intentionally injected into drivers, and although we got thousands of driver crashes, the system continued to run correctly in all trials [3]. It didn't crash even once.

Current Status of MINIX 3

MINIX 3 is not standing still. The MINIX 3 Web site has been visited 1.7 million times and the CD-ROM image has been downloaded over 300,000 times. We have been selected to participate in the Google Summer of Code in 2008, 2009, and 2010. There is a wiki, a twitter feed, an RSS feed, and an active Google newsgroup.

Since the 2007 paper in *;login:*, there have been numerous improvements to MINIX 3, large and small. Here is a brief summary of where the system stands now.

- POSIX-compliant operating system with virtual memory and TCP/IP networking
- User interface is typically X11, although a simple GUI (EDE) is also available
- Various device drivers (e.g., Gigabit Ethernet, OSS audio framework)
- Virtual file system with support for various file systems (e.g., MFS, ISO, HGFS)
- Three C compilers (ACK, gcc, LLVM), as well as C++, Perl, Python, PHP, and more
- Various shells (bash, pdksh, sh)
- Choice of BSD, GNU, or V7 utilities (awk, grep, ls, make, sed, and all the others)
- Many packages (e.g., Apache, Emacs, Ghostscript, mplayer, PostgreSQL, QEMU, vi)
- Software RAID-like layer that protects integrity even from faulty disk drivers
- Numerous correctness, conformance, code coverage, and performance test suites

In addition, other changes to the system are planned for the near future, including:

- Porting of the DDEkit [4], which will give us many new Linux device drivers
- Asynchronous messaging (which means a faulty client cannot hang a server)
- Kernel threads

In short, while the system is not nearly as complete as Linux or FreeBSD, neither is it a toy kernel. It is a full-blown UNIX system but with a completely different and highly modular, reliable structure internally. This also makes it a good research vehicle for testing out new OS ideas easily.

Current Research

We have various ongoing research areas, all focused on the goal of producing a highly reliable, modular system according to the principles given above on modern hardware. We are also committed to producing a usable prototype that clearly demonstrates that you can build a real system using our ideas. Here is a brief rundown of five of the projects.

LIVE UPDATE

Due to its modular structure, we would like to be able to update large parts of the system on the fly, without a reboot. We believe it will be possible to replace, for example, the main file system module with a later version while the system is running, without a reboot, and without affecting running processes. While Ksplice [5] can make small patches to Linux on the fly, it cannot update to a whole new version without a reboot (and thus downtime).

Our starting point for the live update is that the writer of the component knows that it will be updated some day and takes that into account. In particular, the old and new components actively cooperate to make the update process go smoothly. Nearly all other work on live update assumes that the update comes in as a bolt out of the blue and has to be done instantly, no matter how complicated the state the old component is in. We believe this is the wrong approach and that by delaying the update for a few seconds we can often make it much easier and more reliable.

Live-updating MINIX 3 is much easier than live-updating monolithic kernels, because each component runs as a separate process. To update a component, the reincarnation server sends the component an *update* message. The component then finishes its current work and queues, but does not start, any new requests that come in while it is finishing up. It then carefully saves its state in the data store so the new component can find it later. After the new version has been started, it goes to the data store to fetch the saved state, reformats and converts it as necessary, and starts processing the queued requests. The other components should not even notice this upgrade—certainly, not those running user programs.

It is entirely possible that some data structures have been reorganized in the new version. For example, information previously stored as a list may now be in a hash table, so it is the job of the new version to first convert the stored data to the new format before running. In this way, the system may be able to run for months or years, surviving many major upgrades, without ever needing a reboot.

CRASH RECOVERY OF STATEFUL SERVERS

The current system can handle recovery from crashes of stateless drivers and servers but cannot transparently recover components that have a lot of internal state, which is lost in a crash. We need to make sure that the internal state can be recovered after a crash. Checkpointing is not a good approach for components such as the file system, which have a huge amount of state that changes thousands of times every second. Instead, we are experimenting with techniques to replicate and checksum the state internal to each process in real time, as it changes. To do this we need to change the compiler to insert code to do this, which is why we switched to LLVM, which has hooks for this (which ACK and gcc do not).

The idea is that after the crash of a stateful component, a scavenger process comes in and inspects the core image of the now-dead component. Using the replicated data and checksums, it can determine which items are valid and which are not and so recover the good ones. For example, if we can tell which entries in the inode table are corrupted (if any) and which are correct, we can do a much better job of recovering files that are fully recoverable. To some extent, journaling can also achieve this goal, but at greater expense.

SUPPORT FOR MULTICORE CHIPS

We are working on supporting multicore chips in a scalable way. Intel has already demonstrated an 80-core CPU, and larger ones may be on the way. We do not believe the current approach of treating these chips as multiprocessors is the right way to go, since future chips may not be cache-coherent or may waste too much time fighting for cache lines and software locks. Instead, we intend to treat each core as a separate computer and not share memory among them. In effect, our approach is to treat a multicore chip more or less like a rack full of independent PCs connected by Ethernet, only smaller. But we are not sure yet quite how this will work out, so we may allow some restricted sharing. We do believe (along with the designers of Barrelfish [6]) that not sharing kernel data structures across cores will scale better to the large chips expected in the future.

While some people are looking at how to parallelize applications, fewer are looking at how to parallelize the operating system. In the case of the multiserver MINIX 3, in which the processor has many cores and the operating system is made up of many processes, it seems a natural fit to put each process on its own dedicated core and not have it compete for resources with any other processes. These resources include L1 and L2 cache lines, TLB entries, entries in the CPU's branch prediction table, and so on, depending on hardware constraints. In other words, when work comes in, the component is all set up and ready to go, with no process-switching time. If cores are essentially free and the multiserver/multicore design speeds up the OS by, say, 20% or 50% or 100%, even if it uses, say, 3, 5, or 10 cores to do so, that is a gain that you would not otherwise have. Having a lot of cores sitting around idle is worth nothing (except slightly lower energy costs).

NEW FILE SYSTEM

Pretty much all current file systems are recognizably derived from the 1965 MULTICS file system [7], but a lot has changed in 45 years. Modern disk drives exhibit partial failures, such as not writing a data block or writing it to a different location from the intended one, but may still report back success. New classes of devices such as SSDs and OSD (Object-based Stor-

age Devices) are being introduced. These devices differ from disk drives in several ways, with different price/performance/reliability trade-offs. Volume managers and other tools have broken the "one file system per disk" bond but have also complicated storage administration significantly.

When RAID was introduced, it was made to look like "just another disk" to be backward compatible with existing systems. Placing RAID at the bottom of the storage stack has caused several reliability, heterogeneity, and flexibility problems. In the presence of partial failures, block-level RAID algorithms may propagate corruption, leading to unrecoverable data loss. Block-level volume management is incompatible with new device access granularities (such as byte-oriented flash interfaces). Even a simple task such as adding a new device to an existing installation involves a series of complicated, error-prone steps.

In order to solve these problems, we are working on a clean-slate design for a new storage stack. Similar to the network stack, the new storage stack has layers with well-defined functionalities, namely:

- The naming layer (handles name and directory handling)
- The cache layer (handles data caching)
- The logical layer (provides RAID and volume management)
- The physical layer (provides device-specific layout schemes)

The interface between these layers is a standardized file interface. Since the new stack breaks compatibility with the traditional stack, we run it under the virtual file system, so the OS can mount both a disk partition containing a legacy file system and a partition containing the new one. Thus old and new programs can run at the same time.

The new storage stack solves all the aforementioned problems. By performing checksumming in the physical layer, all requests undergo verification, thus providing end-to-end data integrity. By having device-specific layout schemes isolated to the physical layer, RAID and volume management algorithms can be used across different types of devices. By presenting a device management model similar to ZFS's storage pools, the new stack automates several aspects of device administration. In addition, since the logical layer is file-aware, RAID algorithms can be provided on a per-file basis. For instance, a user could have crucial files automatically replicated on his own PC, on a departmental or master home PC, and on a remote cloud, but not have compiler temporary files even written to the disk.

Virtualization

The original work on virtual machine monitors, which goes back over 35 years [8], focused on producing multiple copies of the underlying IBM 360 hardware. Modern virtualization work is based on hypervisors to which guest operating systems can make numerous calls to access services and get work done. In effect, they are more like microkernels than virtual machine monitors. We believe the boundary between microkernels and virtual machines is far from settled and are exploring the space of what exactly the hypervisor should do.

One of the ideas we are looking at is having a dual kernel. In addition to the regular microkernel to handle interrupts, service requests from drivers and servers, and pass messages, there is a component running in kernel mode to handle VM exits. However, the two parts—microkernel and hypervisor—run in different address spaces (but both in kernel mode) to avoid interfering with one another. Taking this idea to its logical conclusion, we may have one microkernel and as many hypervisors as there are virtual machines, all

protected from one another. This arrangement will, hopefully, give us the best of both worlds and allow us to explore the advantages and trade-offs of putting functionality in different places. In addition, we will look at moving as much of the hypervisor functionality to user mode as possible.

We also have some novel ideas on ways to employ this technology to reuse some legacy software, such as device drivers.

Conclusion

MINIX 3 is an ongoing research and development project that seeks to produce a highly dependable open source operating system with a flexible and modular structure. While the grants pay for the PhD students and postdocs and a small number of programmers, we are dependent, like most open source projects, on volunteers for much of the work. If you would like to help out, please go to www.minix3.org to look at the wish list on the wiki, and read the Google MINIX 3 newsgroup. But even if you don't have time to volunteer, go get the CD-ROM image and give it a try. You'll be pleasantly surprised.

ACKNOWLEDGMENTS

We would like to thank Ben Gras, Philip Homburg, Kees van Reeuwijk, Arun Thomas, Thomas Veerman, and Dirk Vogt for their great work in programming various parts of the system. Without their efforts, we would have a paper design but not an actual working system that people can use. They also gave feedback on this paper.

REFERENCES

[1] J. N. Herder, H. Bos, B. Gras, P. Homburg, and A. S. Tanenbaum, "Roadmap to a Failure-Resilient Operating System," *;login:*, vol. 32, no. 1, Feb. 2007, pp. 14–20.

[2] G. Heiser, "Secure Embedded Systems Need Microkernels," *;login:,* vol. 30, no. 6, Dec. 2005, pp. 9–13.

[3] J. N. Herder, H. Bos, B. Gras, P. Homburg, and A. S. Tanenbaum, "Fault Isolation for Device Drivers," *Proceedings of the 39th International Conference on Dependable Systems and Networks*, 2009, pp. 33–42.

[4] http://wiki.tudos.org/DDE/DDEKit.

[5] J. Arnold and M. F. Kaashoek, "Ksplice: Automatic Rebootless Kernel Updates," *Proceedings of the 2009 ACM SIGOPS EuroSys Conference on Computer Systems*, 2009, pp. 187–198.

[6] A. Baumann, P. Barham, P.-E. Dagand, T. Harris, R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania, "The Multikernel: A New OS Architecture for Scalable Multicore Systems," *Proceedings of the ACM SIGOPS 22nd Symposium on OS Principles*, 2009, pp. 29–43.

[7] R. C. Daley. and P. G. Neumann, "A General-Purpose File System for Secondary Storage," *Proceedings of the AFIPS Fall Joint Computer Conference*, 1965, pp. 213–229.

[8] R. P. Parmelee, T. I. Peterson, C. C. Tillman, and D. J. Hatfield, "Virtual Storage and Virtual Machine Concepts," *IBM Systems Journal,* vol. 11, 1972, pp. 99–130.

JIM SANGWINE

3D Web contenders

Jim Sangwine started his career as a lecturer on the 3D animation degree course at Portsmouth University in England in 2002, but by 2004 he had developed an interest in programming and the Web that drew him away from animation and into full-time development. In 2007 Jim left the UK and joined Competa IT in the Netherlands as a platform-agnostic Web application developer working primarily with PHP, C#, and ActionScript. With the support of his colleagues at Competa and funding from NLNET, Jim is soon to begin work on a JavaScript physics library for use with WebGL.

jim@competa.com

3D INTERNET IS FINALLY MOVING OUT of the realm of film and fantasy. We find ourselves at a critical crossroads in the development of this new dimension to our online world. Although the Web is fundamentally standards-based, the reality for developers is that the main browsers adopt and implement standards inconsistently, to varying degrees, and at varying speeds. The situation is exacerbated by vendors of plug-ins vying for market share, and there are already a number of contenders for the 3D content crown. I have looked at what I believe are the three strongest and have run some benchmarks to compare their performance.

The Web Is a Battleground

The defining aspect of the Web development landscape has almost always been (and will, in all probability, always be) the browser wars. While a desktop application developer might have to target a particular platform based on his or her audience, with the possible option of making ports later on, the conceptual platform for Web application developers, and the target audience for our clients, is not educational establishments running UNIX or gamers running Windows 7, but, rather, "the Internet." Our OS is the browser, and it is usually our job to make sure that our application works for as close to 100% of our potential users as possible. This problem is alleviated to some extent by the existence of standards and common languages that all browsers should support (HTML, CSS, JavaScript), but despite (or perhaps because of) the efforts to standardize, the nature of the browser wars means that developers have to put more work, not less, into accessibility and legacy support each year.

The root of the problem is that the update cycle for Web standards and specifications is often very short. The first publicly available description of HTML was seen online in late 1991 and described only 20 elements, of which 13 still existed in the HTML4 specifications published in December 1997. Since 1991 there have been, including XHTML, seven official published updates to the specifications, and HTML5, including XHTML5, is currently in working draft. The first W3C recommendation for CSS was published in December

1996, CSS level 2 was published as a recommendation less than 18 months later, in May 1998, and the first public draft of CSS level 3 was released in April 2000. CSS level 3 is still in development. The standardization model for JavaScript, ECMAScript, has had three published versions since November 1996, but JavaScript is an interpreted language and each of the four major layout engines that are used in the majority of modern browsers offers wildly differing levels of support for the various specifications (see Figure 1) [1].

ECMAScript version support

	Trident	Gecko	WebKit	Presto
Name of ECMAScript Engine	JScript	Spidermonkey/TraceMonkey	JavaScriptCore/SquirrelFish Extreme	Linear B/Futhark/Carakan
ECMAScript Edition 3	Yes	0.6	Yes	1.0
JavaScript 1.5 extensions	No	0.6	Yes	1.0
JavaScript 1.6 extensions (excluding E4X)	No	1.8	Partial	Partial
JavaScript 1.7 extensions	No	1.8.1	No	Partial
JavaScript 1.8 extensions	No	1.9	Partial	No
JavaScript 1.8.1 extensions	No	1.9.1	No	No
JScript .NET extensions	No	No	No	No
ActionScript extensions	No	No	No	No
E4X	No	1.8	No	No

FIGURE 1: COMPARISON OF LAYOUT ENGINES (ECMASCRIPT)

This enormous and rapid development in the definition of Web content has led to an explosion of browser versions. There are five families of browsers with significant share in the market, according to the statistics maintained by w3schools, in descending order of usage as of February 2010: Firefox (46.2%), Internet Explorer (34.9%), Chrome (12.3%), Safari (3.7%), and Opera (2.2%). If you look into the browser versions these stats include, you will find that we are talking about three versions of Firefox, three versions of Internet Explorer, three versions of Chrome, two versions of Safari, and two versions of Opera, for a staggering thirteen distinct variants currently in use [2].

The Problem of Choice

> No study is possible on the battlefield.
> —Ferdinand Foch

It is in this rapid introduction of new browser versions coupled with an apparent reluctance to upgrade on the part of the user base that we find the real barrier to the adoption and support of new standards by commercial developers. The oldest significantly used version of Firefox (3.0–5.6% usage) was released almost two years ago, but much worse is Internet Explorer 6, which was released in August 2001 and yet still represents 8.9% of our potential users. Eight and a half years is almost half the lifespan of HTML and the browser-based Web as we know it today, yet developers the world over are still spending time, effort, and money on supporting this ancient relic. Worse than that, they are often forced to eschew new techniques to maintain legacy support, at the same time restricting the progress of the Internet and prolonging the lifespan of IE 6.

The Browser-Agnostic Alternative

> We can love an honest rogue, but what is more offensive than a false saint?
> —Jessamyn West

Perhaps the most obvious solution to the problem of finding a way to deliver rich, technologically advanced content to users without hitting problems with browser compatibility was always going to be via the use of plug-ins. November 1994 saw the specification of VRML (originally Virtual Reality Modeling Language), a markup standard for the definition of 3D objects, the description of surface properties (shininess, transparency, color, etc.) and the ability to invoke URLs in response to clicks on scene elements. Browser integration was by way of third-party plug-ins, and, unfortunately, was fairly inconsistent. This fragmented support for the standard across the various browsers was likely a big factor in VRML's limited adoption by developers.

In 1996 the first version of Flash was released. Initially called FutureSplash Animator, this application was bought and rebranded as Macromedia Flash in the same year and acquired and rebranded as Adobe Flash in 2007. ActionScript, the object-oriented scripting language of Flash, is based on the ECMAScript standards and so is syntactically very similar to JavaScript. Basic scripting was introduced in version 2 of Flash with a few simple timeline navigation commands (called Actions) that could be embedded in frames of movies, and version 5 implemented the first iteration of ActionScript. Flash deserves special mention here because it has achieved far greater penetration than any other plug-in or even any one browser family. Adobe claims that as of December 2009, 98.9% of users in what they call "Mature Markets" (US, Canada, UK, Germany, France, Japan, Australia, New Zealand) are running at least Flash Player 9 and 94.7% are running the latest version (Flash Player 10). Their "Emerging Market" figures (China, South Korea, Russia, India, Taiwan) are almost as impressive, claiming 98% for at least version 9 and 92.7% for version 10 [3].

The fact is that the Flash Player plug-in is maintained by one organization, and so content is rendered extremely consistently regardless of which browser is used to view it, which makes it a very appealing alternative to other plug-ins and even to JavaScript and AJAX approaches to rich content authoring. Flash was never intended as a 3D platform (although Adobe had started adding rudimentary support for 3D effects with version 10), but there are now a number of extremely powerful ActionScript 3 libraries available, including, most notably, and considered by many to be the current de facto standard for 3D online, the open source Papervision3D.

Papervision3D has already been used for a large number of projects, including some very high-profile integrations with the ARToolkit—for example, the GE Smart Grid Augmented Reality app [4]—and has proven itself as a workable solution.

There are many who feel that the existence of plug-ins, and especially Flash, is detracting from the efforts of (some) vendors to move towards the ideal of an open Web through the introduction of, and adherence to, comprehensive specifications for the native support of rich content such as video and 3D directly in browsers.

Another frequent criticism of Flash and other plug-ins is that they live independently of the page. They are not subject to the browser back and forward buttons, their state cannot be bookmarked in the browser, they don't obey changes to the text size browser settings, they don't appear in page prints, and they do not expose their content to search engines. It is often argued that these are usability issues, and that the use of discrete applications that operate outside of the page breaks the whole concept of the Web.

In addition, plug-ins can represent a security risk, since they do not fit into the security model of browsers. The popularity, bugginess, and implied access to the Internet of browser plug-ins makes them ideal targets.

The Rewards of Rivalry

> Although personally I am quite content with existing explosives, I feel we must not stand in the path of improvement.
>
> —Sir Winston Churchill, August 30, 1941

It can be (and frequently has been) argued that conflict and competition are great catalysts for progress. Just as a world war can stimulate breakthroughs in science and technology, the browser wars have also acted to push forward development. JavaScript has become faster and more powerful, CSS is supported fairly well by the majority of modern browsers, and measures are being taken to improve security across the board. The frustrations faced by developers in trying to keep up with the hacks on top of hacks that are required to maintain consistent rendering for their users are, in the opinion of some, trivial compared to those that would arise from a world without browser competition.

This recently started to hold true for developments in 3D when both Mozilla and Google began work on 3D solutions. Mozilla partnered with the Khronos Group (the organization behind OpenGL) to produce a component called Canvas 3D that exposes the OpenGL ES 2.0 APIs to JavaScript, and Google is working on their O3D plug-in which accepts JavaScript calls through their own proprietary API and uses either OpenGL or Direct3D for rendering. Both projects were intended to be considered for adoption as the basis of an open Web standard for 3D, but it is the efforts of Mozilla and Khronos that have been taken forward for development into a product dubbed WebGL. The WebGL Working Group's members include Apple, Mozilla, Opera, and Google, although Google is still continuing work on O3D.

There is, understandably, a lot of discussion about the relative benefits (and disadvantages) of O3D and WebGL. O3D is currently the more mature technology, and many are electing to sit on the fence awaiting a stable production release of WebGL.

Google's Gregg Tavares posted a message on the o3d-discuss group, stating that they (Google) "have every interest in seeing both WebGL and O3D succeed," but he also voiced an opinion that JavaScript is just too slow to match the potential of O3D [5]. This is mainly because WebGL relies entirely on JavaScript to manipulate the scene, including transforms, culling, sorting, and animation, with only the actual OpenGL calls hardware-accelerated.

Gregg also mentioned that OpenGL ES 2.0 (the API exposed via WebGL) is not supported by "lots of common hardware."

In the same thread Henry Bridge (also from Google) reiterated that they are working on both projects (the two are on the same team). He also said that it makes sense to "standardize GL for JS" but justified continued development of O3D by saying that the gap in performance makes the two technologies appropriate for different situations.

The Facts

Prompted by the comments from Google regarding the potential performance problems inherent in the WebGL model, I decided to try to quickly test the real-world practicality of the three systems.

I started by doing a direct comparison between the latest raft of JavaScript engines and ActionScript in FlashPlayer 10. I ported five of the routines from Apple's SunSpider [6] JavaScript benchmarking application to ActionScript and ran them on a 2.53GHz Intel Core 2 Duo MacBook Pro with 4GB of DDR3 RAM. I experienced a 10+% variance in test times in Flash Player, so I set each test to run 10 times and calculated the mean in order to stabilize the results. JavaScript tests were much more consistent, so I ran those only five times each. The routines I ported were 3D Cube, 3D Morph, Cordic Math, Partial Sums Math, and Spectral Norm Math (see Figure 2).

The browsers I used were the latest nightly builds of WebKit (Safari), Minefield (Firefox), and Chromium (Chrome), so as to represent the current state of JavaScript engine performance. I did not include Internet Explorer in the tests, partially because native versions for my hardware do not exist and testing in a VM would skew the results, but also because WebGL is currently not supported by this family of browsers, making the comparison, at least for now, irrelevant to my investigation.

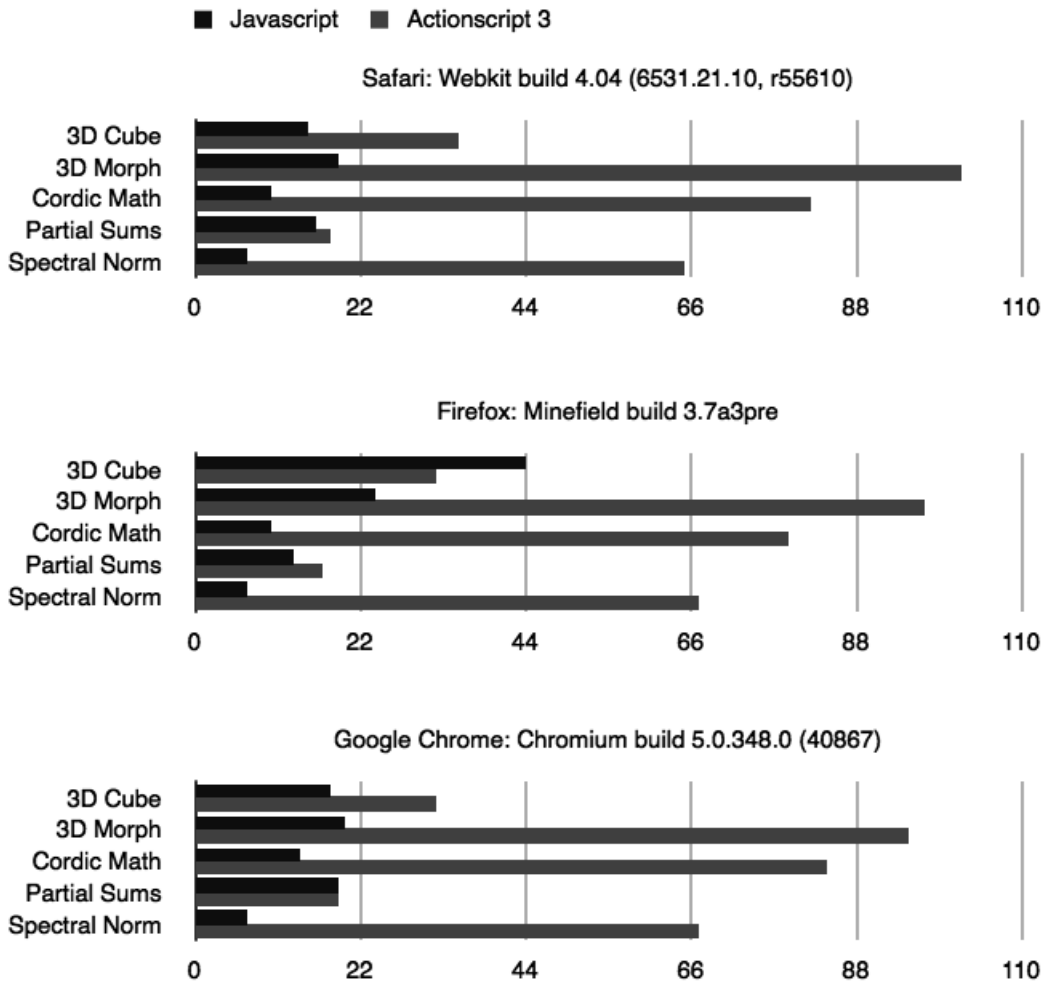


FIGURE 2: THESE CHARTS COMPARE THE RESULTS OF 5 SUNSPIDER BENCHMARK TESTS RUN IN JAVASCRIPT AND ACTIONSCRIPT ON EACH OF THREE PLATFORMS, WITH TIME MEASURED IN MILLISECONDS.

As one would expect, the performance of FlashPlayer was extremely consistent across all browsers in all tests. There was some variance in JavaScript performance, particularly in the case of Firefox, which came out worst, but overall it is very clear that JavaScript is considerably faster than ActionScript

at performing these kinds of calculations. This result was hardly surprising to me, considering the extra overhead involved in running the FlashPlayer. It did, however, suggest that WebGL might indeed be a very strong alternative, at least to what many consider to be the current king of Web 3D solutions, Papervision3D and Flash.

My next test was very crude, and certainly not a comprehensive or even particularly fair comparison, due to the massive differences between the systems, but I think it gives some indication of the usefulness of each, particularly for the kind of undemanding use that the majority of Web developers will put them to. I created three simple projects in ActionScript, WebGL, and O3D, each containing only three elements: a simply shaded sphere, a light, and a camera. I then animated the spheres, rotating them around their Y axis, and experimented with increasing mesh densities to get a very rough feel for the kind of polygon load each technology could handle. The results were pretty interesting.

I only managed to complete all three tests in Minefield; I couldn't get O3D running in WebKit, and neither WebGL nor O3D ran in my version of Chromium. The ActionScript results were again extremely consistent across the three browsers, but WebGL's JavaScript scene processing completed 5 seconds quicker in WebKit than in Minefield, with no noticeable difference in frame rate.

My results in Minefield were as follows:

- ActionScript dropped to 4 frames per second with a sphere of 20,000 triangles and took 2 seconds to load.
- The O3D plug-in couldn't handle a triangle count much greater than 150,000 before exiting with the message "ERROR: The maximum number of elements in a buffer is 1048575." However, a consistent frame rate of 60 fps was maintained with no drop at all up to this limit, and the scene took only 2 seconds to process. It turns out that since some hardware can only handle a maximum of 1048575 vertices per object, the O3D team decided to impose that as a standard limit to ensure the portability of applications. So I ran another test, this time using 10 spheres for a total triangle count of 1,512,500. O3D managed to maintain a frame rate of 15 fps but the scene took 51 seconds to load.
- The real surprise was the WebGL solution, which managed an extremely impressive 15 frames per second with a triangle count of 6,480,000 in a single sphere. The scene took a fairly acceptable 15 seconds to process (and only 10 in WebKit), going a long way towards allaying my fears as to the practicality of handling fairly complex scenes in JavaScript.

Where Do We Go from Here?

In my opinion there are a number of points that need to be considered when choosing among the three solutions I have discussed.

Firstly, there is the question of accessibility. For any technology to succeed on the Web, it must work for the vast majority of users, especially if it is to be considered for serious commercial projects. In this respect, Flash has the obvious advantage as a mature technology that has been around for a very long time in Internet terms and that has achieved (at least according to Adobe) almost total penetration. WebGL, despite still being in beta phase, claims to be supported by most of the major browsers, but the lack of Internet Explorer support will undoubtedly be a deal-breaker in the eyes of many professional developers. Unfortunately, Microsoft has hinted that they have no intention of supporting WebGL in the upcoming Internet Explorer

9. In contrast, O3D claims support for all the major browsers (although my experience did not seem to confirm this), IE included. Also, O3D supports a greater range of hardware than WebGL. However, O3D is just another plug-in, and unless it gets inclusion in the off-the-shelf browser versions like WebGL, it will be an uphill struggle to achieve the kind of penetration Adobe Flash claims.

Another very important consideration is ease of use. The Web is a different environment from the desktop application world, requiring a different set of skills, and Web developers in general are not likely to have experience with 3D animation and physics or shader programming. If 3D is to be adopted for commercial Web projects, it will have to prove itself to be both stable and quick to develop. Agile development has become something of a mantra in the Web application world, and there will likely be strong resistance in commercial projects to any technology that represents a significant cost in training and time. Again, Flash stands out here, thanks to the Papervision3D libraries with which it is incredibly easy to get something up and working with remarkably little code. There is already a vibrant community, and the number of online examples and tutorials is growing. O3D is, out of the box, easier to pick up than WebGL, but still daunting to the uninitiated. However, I see this as a temporary problem, and certainly not a game-winning factor. There are already libraries popping up that encapsulate and automate common tasks in easy-to-implement frameworks, even for the as yet unreleased WebGL. Over time, I believe that the barrier to entry will lower, and for those who wish to, there will still be the option to dive into OpenGL coding.

This leads nicely into the third criterion: power. Here, O3D claims to have the edge over WebGL, and certainly over Flash. As I said earlier, my mesh load experiment was very crude and did nothing to highlight or test specific features of any of the technologies, but to my mind it seems to demonstrate that JavaScript is already more than capable of shifting around some serious numbers at a speed that will allow developers to do much more than is currently possible with Flash and Papervision3D. Perhaps it is not yet possible to produce the next Quake Arena title using only JavaScript and WebGL, but I personally doubt whether that is an appropriate use of the technology right now. I feel that the Web needs to get used to 3D, and that it will be quite some time before developers and users figure out how best to incorporate it into our online world. JavaScript has already increased performance incredibly, and if it becomes popular, WebGL might just be the motivation browser vendors need to invest further in improving their engines and in standardizing their feature support.

Which brings me to my final point: What is better for the Web? Here, I think there can only be one answer: WebGL. While competition can stimulate creativity and progress, I feel that the current chaos of browsers that refuse to die and reinventions of existing technology by plug-in developers are only serving to hold back the progress of the Web. I am almost as passionate about 3D as I am about the Web, and I really want the two to come together. I fear that the plug-in wars that are already beginning will delay the integration of 3D by making it impossible for anything other than Flash to be implemented in applications where reaching a large audience is a critical factor (as it is in the vast majority of projects). Therefore, having a technology that is available to everyone straight from the browser, and whose implementation is controlled by means of industry standards, perhaps even as part of the HTML5 spec, just has to be the best way forward in my view. There might be better options in terms of power or capabilities, but I feel that is a less important factor in the survival of this fledgling relationship.

REFERENCES

[1] Wikipedia, "Comparison of Layout Engines (ECMAScript)": http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(ECMAScript).

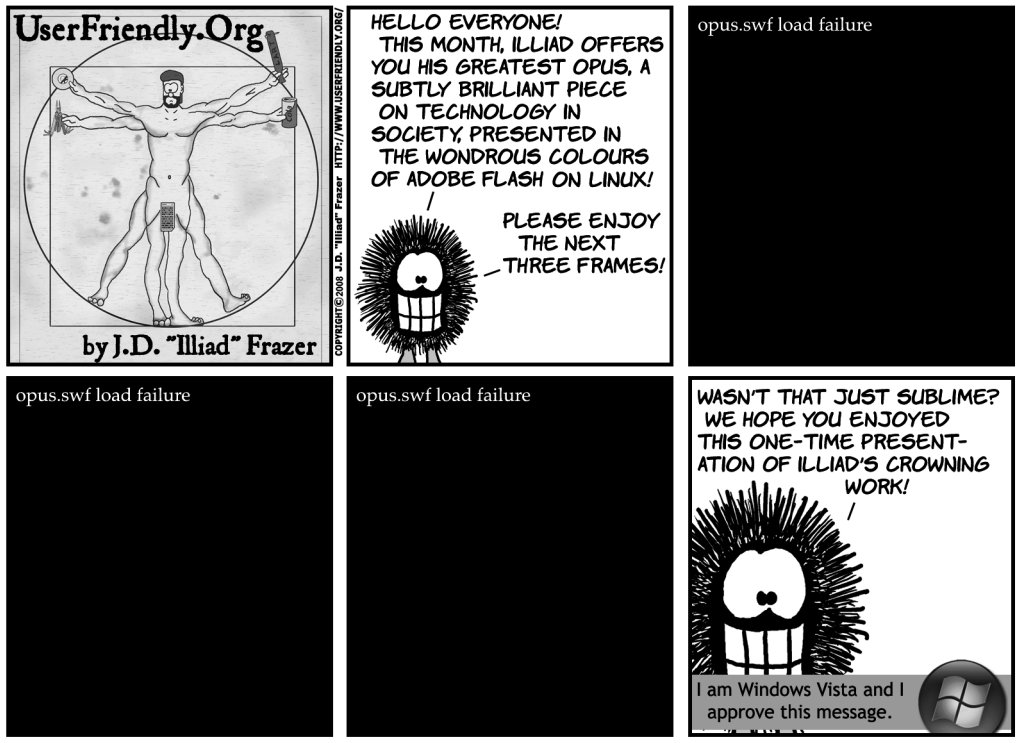
[2] W3Schools, "Web Statistics and Trends—Browser Statistics Month by Month": http://www.w3schools.com/browsers/browsers_stats.asp.

[3] Adobe, "Flash Player Version Penetration": http://www.adobe.com/products/player_census/flashplayer/version_penetration.html.

[4] GE, Smart Grid Augmented Reality: http://ge.ecomagination.com/smartgrid/?c_id=Huff#/augmented_reality.

[5] http://groups.google.com/group/o3d-discuss/browse_thread/thread/7bfa31efcc03b5f6.

[6] http://www2.Webkit.org/perf/sunspider-0.9/sunspider.html.



ALVA L. COUCH

programming with technological ritual and alchemy

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

couch@cs.tufts.edu

I HAVE TAUGHT PROGRAMMING-intensive courses to college students for over 20 years, and I cannot help but notice that the nature of programming has drastically changed in recent years. I teach a generation of students unfettered by my compulsion to understand and, as a substitute, fettered by their own compulsion to do and experience. Unlike the simple languages I employed to learn programming, my students employ complex frameworks, exploit primitive crowdsourcing to come up with solutions, and engage in a shamanistic ritual of dance and pattern to produce software that works, but whose complete function they do not and cannot fully understand. So? As they might say, "What's wrong with that?"

Technological Shamanism

I call the practice of creating software systems from ritual *technological shamanism*. Programming via socially derived ritual makes a surprising amount of sense and is often productive. But to understand its nature, we need to carefully draw some important distinctions between the concepts of "pattern," "example," and "ritual."

A *design pattern* is a proven and reusable approach to solving a specific kind of programming problem [1]. It is called a "pattern" because it specifies the nature of a programming solution without the details. The original description of patterns expressed a pattern as an object-oriented program that operates on objects that implement specified interfaces. These objects represent details the programmer must supply. For example, a "sorting" pattern works on objects that implement a comparison interface. One uses a pattern by implementing the interfaces it requires.

More recently, "design patterns" have acquired a general popular meaning outside the strictures of object-oriented programming, as reusable program fragments with some details missing, in which the method whereby a programmer fills in details is well documented. A pattern is a code fragment that has been proven to work through widespread application, and where its applicability, proven uses, and limits are carefully documented [2]. There are several Internet repositories in which one can search for patterns to solve various problems.

Patterns are a powerful way of archiving programming knowledge in an accessible form. One key part of a pattern—independent of the multiple ways that one can be described—is a statement of its limits, i.e., "how far you can stretch it without breaking it." Applying a pattern—in principle—substitutes for having the knowledge to create it. Using patterns creates code that is, in many cases, better than code written from scratch and based upon full knowledge of system function.

However, the contemporary programmer is not always fortunate enough to have true design patterns in hand. Documenting a pattern is a labor-intensive task. Thus, programmers turn to Internet sources containing "examples" that substitute for patterns, in the sense that the programmer can twist an example to accomplish a goal and/or combine examples toward a new purpose. But *examples are not patterns*. There is no documentation of what can be changed. There is no documentation as to applicability, scope, or limits; there is no claim of broad applicability. There is not even the implicit claim that an example has been rigorously tested.

But an example that has proven useful in a broad variety of contexts—even in a limited way—is not quite an example anymore. It is not yet a pattern and lacks many kinds of documentation. But it is a "ritual" that "usually works." There is some value in distinguishing between "examples," as untried, versus "rituals," as partly validated.

One might compare patterns, examples, and rituals with pharmaceuticals, folk medicines, and folk remedies. A pharmaceutical—like a pattern—has well-documented effects. A folk medicine, like a programming example, might create some effects when, e.g., made into tea. A folk remedy—like a programming ritual—is contrived from a folk medicine through some experience of success, but *without complete understanding of its effects*. In other words, a programming "ritual" is somewhat like a new drug without FDA approval. A ritual "just works," for some partly understood reason.

One weakness of using rituals is that finding new rituals requires some mix of guesswork, experimentation, and/or deep knowledge. Modifications to existing rituals—however minor—are *not* guaranteed to work, any more than modifications of gourmet recipes are; only the master chef knows what can be substituted.

As an example of this, I assigned my students to write a simple document search program in Hadoop, thinking that this was a straightforward program. Not! We got caught in the netherworld between Hadoop 0.19 and Hadoop 0.20, in which the syntax changed enough in 0.20 to invalidate all of the 0.19 tutorials. The tutorial examples said nothing about how to propagate the search terms to the mapper processes. Worse, the *only* candidate variable that we knew enables that kind of propagation had a type that was marked as deprecated! Through some educated guesswork and experimentation, we found out how to do it, though others before us did not fare as well,and we found one Internet lament that text search—which Hadoop was designed to do well—was impractical in Hadoop!

How did we succeed? Well, it is difficult to admit it, but we succeeded by locating a shamanistic ritual with a closely related outcome, searched the Web for related rituals, and then guessed what to do and verified the guess through experimentation, *thus creating our own personalized ritual*. I call this "ritual" and not "pattern," because in the process of making the program work, *we did not obtain a comprehensive idea of why it works, or its limits!*

A little play with modern Web frameworks such as Ruby on Rails, Symfony, and Cake will demonstrate why modern programmers think this

way: one cannot deviate from predefined rituals without courting disaster and/or inconceivable behavior. Frameworks have reached the complexity point where documenting their complete function to a programmer is impractical, or perhaps I should call it "unempowering." The total workings of modern programming frameworks are not that useful to know for someone using them. So we resort to shamanism and employ rituals that the creators of the frameworks kindly provide for us, or we guess and, from experimentation, create our own rituals to taste.

Engaging in ritual rather than understanding is not just exhibited by programmers. System administrators often crowdsource their configurations of software, for desktops or servers, based upon what rituals are found to work by others. The job of the system administrator is *not* to understand but, rather, just to repair the problem. Time pressure often limits personal exploration, so that successful repairs by others are important to know. Often, the quickest approach to troubleshooting is to mine "rituals that work" from the Internet. The mailing lists are full of these simple—but generally effective—uses of crowdsourcing.

From Shamanism to Alchemy

By this time, the reader may think I am advocating a return to barbarism, throwing away the knowledge of the past. I am instead pointing out that we are *already* barbaric, in the sense that our technology has already vastly surpassed our understanding. How empowering is it, for example, to take apart a cell phone? The physical structure of a cell phone is not empowering to someone looking to understand its function, which is mostly hidden.

And the barbarians are winning, because they can produce programs faster than the civilized programmers!

The modern technological shaman, like the primitive shaman, trusts his or her senses and engages in a kind of science. The difference between primitive shamanism and technological shamanism lies in what the shaman's senses include. The technological shaman has the observational powers of the Internet-connected world available and can crowdsource a solution to a mystifying problem simply by querying the proper mailing lists. The appropriate response to "Doctor, it hurts if I do this" is usually "Don't do that; do this"; a non-working ritual is countered with a working ritual, but without a satisfying explanation of why one ritual does not work while the other does.

Like a folk remedy, a modern ritual gains validity through direct observation of when it does and doesn't work. Thus its validity grows with application and observation, including observation of what requirements it does not meet. One severe downside is that there is no such thing as a "secure ritual"; a reasonably complete security analysis would transform it into a pattern!

Crowdsourced solutions are laced with shamanistic rituals that might do nothing, but were part of an initial pattern. I had a student who always put the statement "X=X" into his Matlab program before using a variable "X." This was ritual rather than knowledge (the statement does nothing); but it was extremely difficult to convince him—even with objective evidence to the contrary—that the statement was not needed. This shamanistic ritual reminded me of the rituals of aboriginal tribes who feed wooden birds because it seems to help the crops. Why do it? Because it might help and does not hurt!

One thing that greatly influenced my thinking on social ritual in technology was Jim Waldo's Project Darkstar at Sun Microsystems. Darkstar attempted

to analyze the requirements for interactive role-playing games [3]. To me, the most surprising finding from Darkstar is that young people do not approach interactive RPGs as adults do; bugs are "features," workarounds are "rituals," and software quality is defined in terms of not losing long-term state (although losing short-term state is acceptable). In other words, if you engage in a ritual, the game should not erase your character (a matter of weeks of development), but erasing your character's development for the past day is relatively acceptable! The quality of the RPG system is relative to how it reacts to ritual and whether its reactions to ritual are appropriately bounded. Some Web frameworks could learn from this definition of quality!

So, what is my advice to young students of programming? I do not advise them to program as a "civilized" adult like myself; I am less facile than they are! I do not advise them to reject shamanism and ritual; technological ritual is a basic part of modern survival. I do tell them to develop their own observational skills to a high art, so that they can track a "personal alchemy" that describes *how their rituals interact*. The result of crowdsourcing this "personal alchemy" is a shared "technological alchemy" describing how rituals can be combined to achieve new goals. This social emergence of order—and not the traditional practice of reading and understanding the source code—is already the key to productive programming in the new world of large-scale frameworks.

From Alchemy to Chemistry

It is with some poetic justice that I—having shown no aptitude for undergraduate chemistry—am now put in the position of advocating its value! In the same way that alchemy became chemistry, we need a new "chemistry of programming" that describes how to combine "elements" (our rituals) into "molecules" (more involved rituals) that achieve our ends.

By chemistry, I am not referring to the precise rules of component-based programming or object-oriented programming but, instead, to the fuzzily defined rules by which rituals are combined into new rituals, without full knowledge of the structure behind the rituals. "Programming chemistry" is a matter of classifying what rituals are safe to combine, what combinations are questionable, and what combinations are likely to explode!

Am I advocating throwing away detailed knowledge? Certainly not. But this kind of knowledge—while valuable to the developers of a framework—is not empowering to the average programmer of a framework. Or, rather, it is as empowering as knowing the physics of electrons is to the chemical engineer. This is not a matter of throwing away knowledge but, instead, packaging it in a much more digestible form, in terms of what programmers should and should not do.

Generations

The difference between me and my students is quite generational. I was taught to be compulsive about knowing, in the sense that I will not stop peeling away layers of a thing until I know everything about how it works. This compulsion was empowering for me but is not empowering for my students. Or, it would be better to say, it is about as empowering for them as taking apart their cell phones to see how they work! To my students, I am somewhat of a dinosaur, and to me, my students are the new shamans of technology, engaging in dance and ritual to produce working code.

But my students are not lacking in ambition; they engage in their own unique flavor of lifelong learning. They learn the rituals that work, and the alchemy between rituals: their own descriptions of how to transmute base software components into "gold." But, somehow, it all works, and often it does result in "gold."

REFERENCES

[1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional, illustrated edition, 1994.

[2] See, e.g., Chapter 12 of Roger Pressman, *Software Engineering, a Practitioners' Approach*, 7th edition, McGraw-Hill, 2009.

[3] Project Darkstar: http://www.projectdarkstar.com.

ANDREW SEELY

building a virtual DNS appliance using Solaris 10, BIND, and VMware

Andy works for Science Applications International Corporation as the Senior Command and Control UNIX Engineer at USCENTCOM-J6, and after hours he teaches computer studies courses for the University of Maryland University College. His wife Heather is his init process and his son Marek's first word was "#!".

seelya@saic.com

I WAS FACED WITH THE TASK OF CREATing a new DNS server solution that would be simple to build and maintain, would be easy to deploy to remote locations, and would bring the site into compliance with DoD security requirements. The goal was to improve performance and maintainability while diversifying the DNS architecture and to accomplish it for free. The tools on the table were BIND, Solaris 10, and the site's existing VMware ESX server installation. In this article I explain the background and requirements and detail the installation procedure and scripts developed to deliver a virtual appliance solution.

A Unique Customer with Unique Challenges

I work for Science Applications International Corporation (SAIC) on contract to support the Global Command and Control System (GCCS) at the HQ US Central Command (CENTCOM) in Tampa, Florida. The GCCS shop has the majority of UNIX servers in the organization, so I am also the lead UNIX system administrator. Our contract overall supports the entire spectrum of systems support for the Command, from configuration management to systems engineering, from service desk to cable installation. You'll find members of our team in Tampa writing custom military applications software, in Kabul configuring routers, and everything in between, all in support of the CENTCOM mission.

CENTCOM, like many other government organizations [1], is rapidly expanding its use of virtualization, with a particular focus on server consolidation due to limited datacenter space. If it's a new system coming into the building, the first thing that gets asked is if it can be virtualized. CENTCOM also employs network appliance technologies when appropriate, especially when the technology may be intended for a remote site with disadvantaged communications and limited on-site technical skills. New technical services for the Command will be considered for virtualization, for viable appliance solution, and for commercial off-the-shelf (COTS) purchase. In some cases, new requirements will be candidates for in-house development. In rare circumstances, a solution touches all four.

An appliance-oriented approach to virtualization can bring together the benefits of network appliances and virtualized servers while reducing risk

and cost. This is the approach we took when faced with a customer challenge to improve Domain Name Service (DNS) architecture. By leveraging the latest Solaris 10 x86, Berkeley Internet Name Domain (BIND) software, and the Command's existing VMware ESX environment, a DNS appliance-equivalent was developed, tested, and brought into service for external authoritative and internal recursive DNS requirements.

Motivators for Change

DNS is the kind of service that can start out easy and, without significant effort, scale over time to support an expanding organization. But without a roadmap, DNS can devolve into a patchwork that no one understands, with inherent fragility that can baffle even the most seasoned sysadmin. This was the situation at our customer site. DNS had been grown rather than planned and, due to the natural turnover of military leadership over the course of years, there was no single guiding authority or principle to direct that growth; in military terms, there was no Concept of Operations (CONOPS). DNS worked fine for years, until suddenly it didn't work at all for a few days. The entire site was effectively down while the operations team essentially had to spelunk the DNS configuration to find the failure point and repair it. While workarounds were immediately implemented to prevent mission compromise during the outage, all involved understood that a military network is a vital tool and that this type of fragility is unacceptable.

One of the findings that surfaced out of the after-action review was that the site's DNS architecture was not aligned with industry best practices and the Defense Information Systems Agency (DISA) Security Technical Implementation Guide (STIG) [2] requirements for split-DNS and heterogeneous implementations. During a redesign discussion it was noted that both problems could be fixed at once by deploying a new and different set of DNS servers for the relatively simple external authoritative and internal recursive roles, leaving the more complex internal authoritative zones to be served by the site's existing Microsoft implementation.

Classic Build-Versus-Buy Decision

There are proven DNS solutions in the appliance market, with a focus on security, ease of use, and total cost of ownership [3]. These appliances tend to use a hardened Linux kernel and a purpose-built Linux distribution to reduce the total number of system "knobs that may be turned." The Infoblox [4] appliance was strongly promoted at the redesign meeting as the best way to simplify management and integration. Others at the table were anti-appliance, reminding leadership that the failure case for an appliance means a field engineer service call and expensive support contracts compared to in-house expertise that can respond immediately. This caused everyone to discuss what it would mean to build something in-house. Everyone agreed that BIND on Solaris would be an ideal solution from a performance standpoint but was still not as attractive as the appliance from a security and simplicity point of view. I raised a hand and volunteered to build BIND on Solaris, virtualize it in VMware, meet all the factors that made the appliance appear attractive, and do it at no cost.

The DNS "virtualized appliance" project resulted in an appliance-like DNS capability that allows the Command to comply with DNS security requirements, simplifies overall configuration, and significantly improves configuration management control of the external authoritative DNS function. Total cost of ownership may ultimately be higher due to the relatively few Solaris

experts on the team, but clear documentation of design and the very terse installation leave very few moving parts to debug and should prevent any perception of added expense. Using the virtualization technique, we were able to save a little bit of space and power in an already cramped datacenter, and by focusing on security issues as a design parameter I was able to make a hardened solution that easily conformed to Department of Defense (DoD) requirements for UNIX systems [5].

Initial Design Parameters and Assumptions

The Command leadership's biggest complaint (after, of course, DNS having suffered an outage) was that configuration management (CM) of the DNS was almost nonexistent. Multiple people in different shops had access to zone files, there was little accountability for changes, and there was no revision control. For serious CM, the DNS appliance technical solution needed to be considered from the start.

Command Information Assurance (IA) immediately had concerns about STIG-compliance and overall accreditation of this type of system. Without IA approval, no solution, no matter how good, would see a network light. The project needed to be STIG-compliant as it grew, preferably with IA involvement at each step of the development.

The solution had to be small, agile, and portable, easily deployable to other networks, and easy to upgrade and maintain. We developed this into a DNS "multiple master" concept, with a plan for potentially multiple DNS master authoritative servers without any slaves, increasing the simplicity of design and deployment while obviously trading off the flexibility inherent in a master-slaves architecture. We determined that there would be no requirement for any BIND server cross-talk from other areas of the DNS architecture, which effectively eliminated any requirement for an authorization list or key management and made the product highly and rapidly deployable. There were very few tunable knobs in the design parameters.

The Technical Approach

To limit the attack surface of the system we agreed that the only networking ports that would be available would be UDP/53 and TCP/53 in and out for processing DNS requests, and UDP/123 out for Network Time Protocol (NTP). When a syslog host was implemented, then UDP/514 would need to be opened between the DNS host and the syslog host. Otherwise, all network ports would remain closed, even TCP/22. All command-line access was limited to console-only and would be controlled and audited tightly by access from the VMware console.

To keep the footprint as small as possible, I installed the system with Solaris 10's minimum possible disk allocation of 1.2GB. This could be shrunk down after the operating system was installed, but there was no requirement to have a smaller installation.

My initial development environment was VMware Workstation 6.5.3. I added a new VM with parameters for Workstation 5 and ESX server compatibility enabled. The initial concept was to build the vmdk image file in VMware Workstation and then import it into ESX. This worked well for initial proof-of-concept testing, but for production I built from scratch in the native environment.

I built a Solaris 10 U8 system with ZFS, no naming service defined and no remote services enabled. I selected the "Reduced Networking Core System

Support" group and customized it to remove unnecessary tools and to add in essential options:

- Remove audio drivers and applications
- Select BIND nameserver manifest
- Select Basic Audit Reporting Tool (BART)
- Select Basic IP Commands (root)
- Select Basic IP Commands (usr)
- Remove Kerberos version 5 support
- Remove Network Information System (NIS) support
- Select Network time protocol
- Select Programming Tools
- Expand Remote Network Services and Commands and select Remote Network Client Commands
- Select secure shell
- Remove non-mandatory Universal serial bus software
- Remove libexpat
- Remove xvm paravirtualized drivers

This configuration creates dependency warnings for USB, GSSAPI v2, and Kerberos. These warnings may be safely ignored. This build process results in a very terse Solaris 10 installation. But there's more work to be done to make it ready for action.

In a separate VM I installed a full OEM Solaris 10 with all defaults as a development environment. In that environment I installed the Solaris Freeware GNU C compiler [6] and all related dependencies. In the development environment I built several scripts and archives to make deployment of the DNS appliance rapid, simple, and controlled. The pre-configured system files I set up were:

- profile for root to set umask, mesg, stty, and TMOUT values for STIG compliance and usability
- syslog.conf to set standard site logging specifications
- Customized DNS server manifest to set chroot for BIND [7]
- resolv.conf to set the site domain and the local host as the nameserver
- ntp.conf with the site's NTP server IP, and defined locations for driftfile and statsdir
- motd to meet DoD and STIG requirements
- db.0.0.127.in-addr.arpa.txt with a typical loopback PTR record
- db.roothints.txt with DoD root servers
- named.conf: The initial default configuration is for a recursive server only; authoritative installations will require the named.conf and zone files to be updated. Set logging options for syslog to support a remote log host.
- bart.rules, with directories to be monitored by the Solaris 10 Basic Audit Reporting Tool (BART) [8]: /var/named, /etc, /usr, /opt, /bin, /boot, /sbin, /lib
- lock: A script (see Listing 1, below) to disable useful tools before a host is placed into production. Note that whenever the machine is locked down, BART is executed, and a sysadmin is expected to follow up on any BART-reported changes before the system goes live.

```
#!/bin/sh
svcadm disable ftp
svcadm disable ssh
chmod 400 /usr/bin/truss
chmod 400 /usr/sbin/ping
chmod 400 /usr/sbin/traceroute
chmod 400 /usr/bin/ftp
chmod 400 /usr/bin/telnet
chmod 400 /usr/bin/ssh
```

```
chmod 400 /usr/sbin/snoop
chown –R root:root /root
chmod –R o-rwx,g-rwx /root
if [ -d bart ]
then
echo Creating BART baseline
cd bart
latest=`ls -tr1`
rightnow=bart.`hostname`.`date %Y%j%H%M`
bart create -r bart.rules > $rightnow
echo Comparing BART baseline
bart compare $latest $rightnow
[ $? -ne 0 ] && echo BART discrepancies found || echo BART integrity OK
cd ..
fi
```

LISTING 1: THE LOCK SCRIPT DISABLES SERVICES, CHANGES PERMISSIONS TO ROOT ONLY FOR SOME COMMANDS, AND RUNS BART.

- unlock: A script that enables the services and utilities that are disabled by the lock script.
- recursive.tar: A tar of the /var/named directory, used to quickly recreate the chroot file environment on the production system.
- configure.sh: The script (Listing 2) to execute on the production build that completes the configuration and makes the host ready for operations.

```
#!/bin/sh
groupadd named
useradd -m -d /var/named -c "BIND User" -s /bin/false -g named named
tar -xf bind.binaries.tar
tar -xf recursive.tar
cp syslog.conf /etc/syslog.conf
cp .profile /root/.profile
cp motd /etc/motd
cp ntp.conf /etc/inet/ntp.conf
cp server-chroot.xml /var/svc/manifest/network/dns/
cp resolv.conf /etc/resolv.conf
mkdir -p /var/named/var/named
mkdir /root/bart
mkdir /var/named/var/log
mkdir /var/named/var/run
cp /etc/rndc.key /var/named/etc
chown -R named:named /var/named
mkdir /var/named/dev
mknod /var/named/dev/poll c 135 0
chmod 666 /var/named/dev/poll
chmod 640 /var/named/etc/named.conf /var/named/var/named/*
svccfg validate /var/svc/manifest/network/dns/server-chroot.xml
svccfg delete dns/server
svccfg import server-chroot.xml
svcadm enable dns/server
svcadm enable ntp
svcadm disable network/inetd
svcadm disable cron
svcadm disable name-service-cache
svcadm disable iscsi/initiator
svcadm refresh system-log
```

LISTING 2: THE CONFIGURE.SH SCRIPT SETS UP BIND AND DISABLES UNNECESSARY NETWORK SERVICES.

In order to keep the production system as clean as possible and maintain the ability to independently upgrade the BIND version without waiting for Solaris patches, I did an offline compile and transfer. In the development environment, I downloaded the latest BIND source distribution from ISC [9] and compiled with prefix of /usr and sysconfig of /etc. After compiling and testing, I made a tar roll-up into bind.binaries.tar of all the newly created binaries from /usr/sbin, /usr/lib, and /usr/bin.

Finally, I made a single config.tar.Z containing all the files created in the development environment. This is the "secret sauce" that is required to complete the transition from generic "terse Solaris 10" to hardened DNS appliance. To complete the configuration, enable ssh, set up root's $HOME, run the configuration script, and run the lockdown script:

Enable ssh for root. This is required to copy the configuration file in; ssh will be disabled in the lock script before the host is deployed in a live environment.

```
/etc/ssh/sshd_config, change "PermitRootLogin no" to "PermitRootLogin yes"
svcadm refresh ssh
scp config.tar.Z to /root/config
```

Set up root's home directory:

```
mkdir –p /root/config
Edit /etc/passwd, change ":/:" to ":/root:"
```

Configure the host and lock it down:

```
cd /root/config
uncompress config.tar.Z
tar –xf config.tar
rm config.tar
./configure.sh
./lock
```

The result is what I consider to be a "versioned release." Only copies will be given live IP addresses and placed into production, while a read-only archive of this and future versions will be made for reference. To turn this into an authoritative DNS server the named.conf and zone files must be updated to reflect the authoritative zones and to remove the root hints reference to prevent recursive queries.

A Close Look at the Running System

The resulting DNS appliance presents a very low-drag surface. Required VM server farm resources are limited to 2GB of disk device storage and 1024MB of system memory per installation. The only actively listening port is for BIND and the system is protected from unauthorized network connections in or out by an external firewall monitored by the security team. System boot time in the VM is less than a minute. DNS response time is rapid and scales well; using first perfquery and then a scripted local test harness we successfully tested the installation under loads exceeding that experienced by the previous DNS. The only processes running are essential kernel functions, ntpd, and named, as shown in the following process table:

```
# ps -ef
     UID   PID  PPID   C    STIME TTY         TIME CMD
    root     0     0   0 18:31:17 ?         119:31 sched
    root     1     0   0 18:31:18 ?           0:00 /sbin/init
    root     2     0   0 18:31:18 ?           0:00 pageout
    root     3     0   0 18:31:18 ?           0:00 fsflush
    root   240     1   0 18:31:29 ?           0:00 /usr/sbin/syslogd
    root     7     1   0 18:31:19 ?           0:02 /lib/svc/bin/svc.startd
    root     9     1   0 18:31:19 ?           0:05 /lib/svc/bin/svc.configd
  daemon   130     1   0 18:31:26 ?           0:01 /usr/lib/crypto/kcfd
    root   201     7   0 18:31:28 ?           0:00 /usr/lib/saf/sac -t 300
    root   222     1   0 18:31:29 ?           0:00 /usr/lib/utmpd
    root   336   232   0 18:44:34 console     0:00 ps -ef
    root   232     7   0 18:31:29 console     0:00 -sh
    root   110     1   0 18:31:25 ?           0:00
/usr/lib/sysevent/syseventd
    root   263     1   0 18:31:32 ?           0:00 /usr/lib/inet/xntpd
    root   205   201   0 18:31:28 ?           0:00 /usr/lib/saf/ttymon
   named   267     1   0 18:31:33 ?           0:01 /usr/sbin/named -t
/var/named
    root   248     1   0 18:31:30 ?           0:03 /usr/lib/fm/fmd/fmd
#
```

The CM Plan

One of the biggest concerns leadership had with the original DNS way of doing business was the lack of configuration management or attribution for system changes. By using the VMware images, a "golden image" concept, extremely limited direct access, and a strict workflow, the DNS zones will be extremely stable. The key to this approach is that all configuration changes will be made offline to a candidate release; only after the candidate release is tested and validated will it be cloned for production and then archived for reference. Candidate change lists will be approved by the DNS, UNIX, and Security teams before implementation. The planned battle rhythm is:

- Monthly: New numbered release incorporating updated DNS configurations, security patches, antivirus updates as required by DISA standards [5, 10], and other approved configuration changes. Root password is changed monthly.
- Annually: New major-number release for major operating system updates, as released.
- As needed: Independent security scans as new scanning tools are updated.
- As needed: Emergency updates for security patches or DNS configurations.

What About User Accounts?

In the DoD space, passwords have extreme complexity, length, and change frequency requirements. Given that each deployment of the DNS appliance is expected to stand alone and that direct access will be limited by the VMware console, a conscious decision was made to completely remove the requirement for user accounts and thus the requirement for accounts management. As delivered, this virtualized DNS appliance is limited to the root login with user audit requirements satisfied at the VMware console.

Part of the roadmap for the DNS appliance project is to completely lock the root account, effectively removing *any* command-line access from an operational system. This remains a controversial topic with most of the site's system administrators, but consider the possible needs for logging in to the operational system:

- Inspect system logs: No need because logging is sent to a syslog host.
- Clear file systems: The limited applications running on this system do not generate files and logs are sent to the loghost.
- Performance monitoring: Utilization of CPU, memory, and I/O all are accomplished from the VMware hypervisor monitor.
- Firewall log analysis: Port-level protection has been outsourced to the external firewall, where monitoring is already being performed.
- Performance tuning or zone file editing: All system changes should be done in a candidate release image that gets tested, cloned, and swapped out with the running system; there should never be an edit of the configuration of a running system.
- General troubleshooting: In the event that a system needs troubleshooting, a new, unlocked clone will be put on the live network to gather data and then taken back offline for analysis.
- Log in to change passwords every 90 days: No need to do this if there are no enabled accounts on the system!

Results of the Virtualized DNS Appliance Project

There is no maintenance or licensing sustainment fee, no additional charges for increasing the installation base, and the only cost of sustaining ownership is the maintenance of a Solaris skill set within the Command. Solaris 10 is used with a cost-free license, BIND is used with a cost-free license, and the existing VMware ESX covers the expanded use. Total additional cost to the customer: $0.

The DNS appliance project produced a hardened, reliable, scalable, DNS solution that meets all design parameters and is compliant with DoD and Command information assurance requirements. The appliance concept has been deployed operationally for eight DNS server installations, without any loss or degradation of service, and more deployments are expected as the requirement grows and the product matures.

REFERENCES

[1] "Virtualization and Consolidation in the Federal Government," *Government Computer News*: http://gcn.com/microsites/virtualization-consolidation/efficiency-gains.aspx.

[2] DNS STIG: http://iase.disa.mil/stigs/checklist/unclassified_dns_checklist_v4r1-8_20100226.doc.

[3] "Do It Yourself DNS," *Network Computing*: http://www.networkcomputing.com/1406/1406f3.html.

[4] Infoblox: http://www.infoblox.com/.

[5] UNIX STIG: http://iase.disa.mil/stigs/checklist/unclassified_unix_checklist_v5r1-23_20100226.zip.

[6] Solaris freeware compiler: http://www.sunfreeware.com/indexintel10.html.

[7] BIND chroot manifest example: http://blogs.sun.com/anay/resource/server-chroot.xml.

[8] Basic Audit Reporting Tool (BART): http://docs.sun.com/app/docs/doc/816-4557/bart-1?a=view.

[9] BIND download site: http://ftp.isc.org/isc/bind9/.

[10] Jim Laurent's commentary on the DISA requirement for UNIX antivirus: http://blogs.sun.com/jimlaurent/entry/update_anti_virus_software_for.

MATT SIMMONS

LISA: beginning the journey

Matt Simmons is an IT administrator with several years' experience on small and medium-sized networks. He is currently employed in the financial services industry and has previously worked in logistics and Internet services.

standalone.sysadmin@gmail.com

TWENTY-FOUR. THAT'S THE NUMBER OF hours I spent in structured training in 2009. Zero hours were spent the year before that. And the year before that. Not coincidentally, twenty-four is the number of hours I spent in classrooms at LISA '09, being trained by some of the best in the world.

Prior to attending LISA '09 in Baltimore, MD, I had been to exactly one conference, and that was the second year of the Ohio Linux Festival. Visiting an established conference that was professionally organized and executed was a new experience for me, and it opened my eyes to a world of system administration whose existence I had little suspected.

I learned several lessons. First, the LISA acronym is really a misnomer. Large Infrastructures are relative. I met people who measured their computing power by rooms, and others who made my paltry 100 nodes seem complex. Both were there to learn how to better manage their infrastructures. Both could learn from each other, and for the most part saw each other as equals.

I learned that during a time of tight budgets, there are a lot of companies around the world that see the value in having their people trained by experts who not only know the field but, in some cases, invented it. And yet, though the administrators were from diverse backgrounds, countries, and tongues, we all shared the same types of problems. We could commiserate, tell war stories, and complain about (and be thankful for) users.

I also learned that maybe the most important component of success in the long run is a sufficiently deep support network. Although we use mailing lists, IRC, and instant messaging to communicate with other administrators in the field, making a physical connection to someone is different, somehow. Shaking someone's hand and looking them in the eye says more than a dozen emails can. Making contacts, acquaintances, and friends increases our network of resources when we have problems, and gives us someone to talk to when we aren't getting answers.

All of this I learned before I ever set foot in a classroom.

These lessons not only make me a better administrator, they make me a saner administrator. I extended my experience through the people that I met, and made contacts and friends that I expect to keep for years to come. The extracurricular

activities held at LISA were the flip side of the coin, where you really got to know the people you met during the hallway track.

The Birds of a Feather I attended were fascinating, and as informative as classes, themselves. Even though it was my first LISA, I got my feet wet and hosted a couple of BoFs myself. One of the forms of evening entertainment was unique to my experience. BigFix, a conference sponsor, held a Sysadmin of the Year contest with a rock star theme, the results of which were to be announced at LISA. No celebration is complete without a cake, and no rock star themed party is complete without a rock band. To satisfy both requirements, BigFix got Baltimore native Duff Goldman, of Food Network's Ace of Cakes, to make a cake, and he and his band played the party. I was even lucky enough to get an honorable mention!

The combination of the hallway track, BoFs, vendor floor, and being surrounded by sysadmins was incredible, especially for someone who hasn't been in that situation before. As amazing as all of these things were, what was going on inside the classrooms was even more amazing.

In a few days, I attended eight classes, and got deeper knowledge of the technical subjects than if I'd spent an entire week teaching myself the topics. I suspect that all system administrators are autodidacts to one extent or another, but receiving first-hand information from someone who has been in the field 20 years is a fairly efficient method of knowledge transfer. I started with Maurita Plouff's class, "Management Skills, or Don't Panic!" The title of this class might as well have been "Directed Social Engineering," because the omnipresent theme was efficiently getting people to do what you wanted, whether they are your supervisor, users, or some other stakeholder. These skills are valuable, particularly in a field where labels like "people person" aren't thrown around often.

;login: editor Rik Farrow spoke to my heart with his SELinux course. The first slide in his deck was "Re-enabling SELinux," which made the absolutely correct assumption that the first course of action most of us take when building a new machine is to turn it off. Working from that premise, Rik built his case for the usefulness of this feature from the ground up, discussing secure-computing history, as well as the background for the early decisions that led to the Linux kernel modifications enabling this software. Rik gave us several tools for evaluating the SELinux configuration for our machines, and encouraged us to move our policy setting from Disabled to Permissive, at least until we can develop a working security policy that will allow us to function under Enforcing.

It's beyond the scope of this article (and your patience, I'm sure) to give a detailed review of every course I took at LISA, but one really stands out in my mind. Before going into Linux Performance Tuning, I hadn't heard of Theodore Ts'o, which says more about me than it does him.

If you use EXT3, EXT4, or Linux in general, you're using Ted's code. According to his Wikipedia page, he was the first North American Linux kernel developer. It's probably better for me that I didn't research him before I took the class, because I might have been too in awe to fill up the eight-plus pages of notes that I managed to scribble during the class. Prior to this class, I was under the impression that I understood a decent amount about how the operating system worked. I was incorrect.

This course would have been worth the entire trip by itself. It is directly because of my experiences in this class that I have received numerous accolades from my supervisor and the heads of other departments in my company. By applying the lessons I learned (and information that I researched

because of it), I am in a much better position to plan, implement, and maintain my infrastructure. I am a better administrator because I was in this class. It's as simple as that.

The aggregate effect of my experience is relatively easy to measure. After seeing the value of formal training, and thanks in large part to my experiences at LISA, my company's management has taken a new track. We have already scheduled over forty hours of training in the coming year, several of which will be at LISA '10 in San Jose.

RUDI VAN DRUNEN

small embedded systems

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

rudi-usenix@xlexit.com

EMBEDDED SYSTEMS CAN BE FOUND virtually everywhere. In this article I will describe some of the features of these small computing engines and will take you along the road to building and programming one yourself to ease simple control tasks in your daily life. In the embedded world, understanding issues of small systems is much easier when you have encountered them yourself.

Nearly all consumer electronics nowadays have some kind of microcontroller built in. Often these devices have small 8- or even 16-bit processors with on-chip peripherals, such as analog to digital (A/D) converters, input-output (I/O) ports, RAM, or ROM (flash), called microcontrollers, on board. These processor systems serve basic tasks that used to be taken care of by either mechanical devices or analog electronics. An embedded system is a small, reliable system specially designed and built to do one specific task, such as control your appliance, and cannot be user-instructed to perform other tasks easily.

A typical example of such an embedded system is in your microwave oven. The microwave often has a display, keyboard, and a number of sensors and actuators. Sensors can be a temperature sensor or the sensor that detects whether the door is closed. Actuators can be the electronic switch that controls a microwave tube or a system that controls the rotational speed of a fan. Other examples of embedded systems are the motor management systems found in all modern vehicles or controllers of all sorts, the power and temperature controller of your laptop or server, WiFi routers, and, to a lesser extent, game consoles, which feature a more complex software environment, including an operating system. You can see that the NTP server I described in [1] is also in this category.

Layout

Most, if not all, of the low-end embedded controller systems do not run an operating system that hides all devices for the user or developer. Instead, the user program is directly in full control of all hardware and there are no separate processes or tasks. Also, all interrupts on the system are to be handled directly by the user software. Often the interrupts are triggered by the hardware when asserting the interrupt pin on the chip to a specific logic level. This can be done by an external device or switch.

More complex embedded systems run specialized microkernels capable of job scheduling and doing real-time interrupt processing. In some cases this can be some flavor of UNIX, most commonly Linux or BSD. In this article we will focus on the systems that have no operating system, as these are the simplest to work with.

Hardware

There are a number of different families of embedded processors, all with their own architecture. Often the chip vendor manufactures a large number of different chips with the same core processor architecture, but with an enormous diversity of hardware interfaces, such as serial and parallel ports, analog inputs and outputs or even USB, or complete Ethernet interfaces with a hardware TCP/IP stack on the chip. This makes the development of hardware devices with these processors fairly easy. You don't need many chips to build a complete processor system. The downside is that the different families of processors all have their own architecture and instruction sets and need different development tools, so you may select a chip that has far too many I/O devices for your specific task.

Software

As I said before, the described embedded systems do not have an operating system, so development of software for embedded systems almost always takes place on a host system that has cross-compilers/assemblers to generate code for the target device. The target device is a board containing (at least) the processor and peripheral chips and sensors/actuators that will be used in the final application.

Lots of different vendors supply their own software or even hardware development environment for the processor family they make. These toolsets (and boards) often consist of an editor, a compiler/assembler, and a (down) loader. Unfortunately, almost all of these software tools run only under the Microsoft Windows environment. WINE can be of use here, but your mileage may vary, as vendor support may be very limited; especially when using complex debugging or download hardware that communicates to the IDE, you might encounter some problems.

Nowadays, it is quite common to do development for embedded systems in a higher language such as C or C++, in contrast to earlier development, which was completely done in assembly. By selecting C or C++ as a development language, it is fairly easy to use the well-known open source GNU toolchain (gcc) to generate code for a different architecture than is used to compile on. This process is called cross-compiling/assembling. After cross-compiling, assembling, and linking your program, libraries, and start-up code, the result is either a full binary file or an ASCII hex-file that contains the application and supplemental code. This file can be downloaded into the on-board or on-chip flash in the target system. A hex file is an ASCII file with the hex representation of the addresses and bytes that need to be there, with an additional checksum [2].

System Robustness

While writing software for embedded devices it is important to remember that these systems often will be running 24x7 for many years without being rebooted. Also, often these systems will be controlling devices that need safety precautions. Recently, embedded systems in cars have been in

the news because of lack of software robustness [3]. If the software or the hardware of an embedded system fails, often dangerous situations can occur. Keeping this in mind, software robustness in embedded systems is very important. Also, the robustness of the hardware is a key factor. In software it is important not to use uninitialized variables, so be careful with dynamic allocation of memory, for example. On the hardware side, it is important to use a quality platform, good PCBs, and good manufacturing processes, and allow for timing and temperature margins in the hardware design.

Watchdog

To enhance system robustness it is wise to build in a watchdog system. A watchdog is a combination of a hardware timer, which resets the processor when it counts down, and some software statements throughout the main loop of the program. When running, the software statements reset the hardware timer before it runs out and resets the processor. If for some reason the software crashes, the hardware timer will not be reset and the system processor will be reset and the software restarted.

Downloading

Downloading the code onto the target processor or target board will get you to a bootstrap problem, as the processor generally has no code at that very moment and cannot help you with downloading the image into its flash memory. There is no code present to take a byte from an (e.g., serial) interface and write it to flash. So JTAG is often used to bring up a board from scratch. JTAG [4] is a bit-bang protocol that is supported by a number of different hardware components to do in-circuit production testing, but downloading bytes into memory devices or on-chip memory without using the processor is also supported. After downloading your binary image into the memory of the processor or on the board, the reset line of the processor can be released and the processor starts executing the just downloaded code.

Some embedded evaluation boards come complete with a processor that has a piece of code in its ROM that is either factory programmed or programmed into the memory using a chip-programmer. This piece of code is called the bootloader. It runs after reset and instructs the processor to accept databytes, often in a HEX file, and put them in memory. Then it instructs the processor to start executing the just downloaded code. This is by far the easiest way to program/download code into a processor or board. Often, the toolchain used contains special download software to do just this.

Debugging

As the target system often has no operating system, it can be quite difficult to debug an embedded program running on a piece of target hardware. A number of options are available here:

1. Use of a simulator/emulator running on a host machine, simulating the target processor and all peripherals. This system provides a complete simulation of the target in software and thus all software debugging features.
2. Use of an in-circuit emulator. This piece of hardware replaces the hardware processor and gives full control of all innards of the processor, including breakpoints.
3. Use of an in-circuit debugger, mostly connected to the system using JTAG (see above) if your target processor has features for debugging using these kind of tools.

4. Use of I/O, adding statements to your software to show the state on the available I/O: for example, displaying the state on an LCD display if available, or flashing LEDs on the board if your software reaches a certain state. The problem here is that you are actively interfering with the program code you are debugging.

Processor Families

In the embedded world there are a number of popular processor families. Next to the standard processors, some large vendors design their own processor using semi-custom chips and (VHDL) processor cores that they integrate. The most important and popular processor families are:

- **Intel 8051** [5] The 8051 is one of the earliest embedded processors controlling appliances. It is an 8-bit processor built on the Princeton architecture.
- **ARM** [6] The ARM family features a 32-bit core and is licensed to a number of different manufacturers that integrate the core with different peripherals on one chip. The ARM chip is a RISC architecture and capable of running at a fairly high clock speed.
- **PIC** [7] The PIC (Programmable Interrupt Controller) is a controller family built by the Microchip Corporation. They are available in different subfamilies featuring 8-, 12-, or 14-bit memory addresses, but all feature a small set of 8-bit instructions and have a Harvard architecture. PICs are generally popular with hobbyists and industrial developers. Microchip, as well as other vendors, supplies IDEs and compilers/assemblers for different languages. Some of them are open source, such as JAL [8].
- **AVR** [9] The AVR microcontrollers are 8/16-bit RISC devices built by Atmel. There are three subfamilies of the AVR chips: TinyAVR, MegaAVR, and xMegaAVR, each featuring more memory and complex peripherals. The latest addition to the AVR family is a 32-bit DSP-like architecture.

Of course there are many more microcontroller families with their own specific features and instruction sets, often targeted to a specific application area. Nowadays most applications requiring an embedded microcontroller are often built in an Application Specific Integrated Circuit (ASIC), where the processor often is integrated as a VHDL module together within the custom chip.

Practicalities

To get started building a small system and developing software for an embedded system, you need to consider a number of different things. First of all, you need to select the processor family for the job. This can be difficult, since many parameters are in play. One of the major issues is the number of resulting devices you need to produce. If you are building a one-off device for a project, you may want to focus on ease of learning to program the device. Next, the hardware interface possibilities (interface ports, analog and digital on the selected chip) and software development environment are important.

AN EXAMPLE

As a starting point, I suggest a readily available small controller board. This lowers the threshold to get started, especially for non-electronics engineers. The Arduino [10] board as shown in Figure 1 is a reasonably cheap AVR board with a powerful processor with many interface possibilities. This

popular board features an AVR ATmega328 processor and an open source development environment. Following the Arduino concept and the same microprocessor, there are also a number of other boards available in the public domain, such as the JeeNode [11] or StickDuino [12].

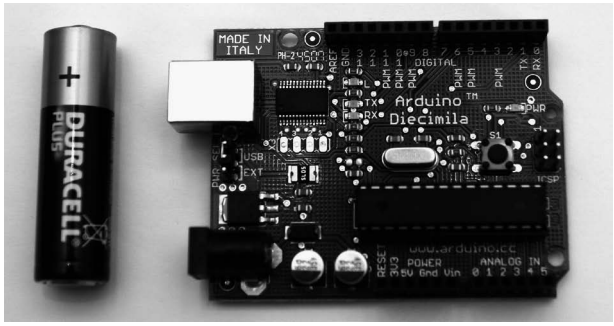


FIGURE 1: AN ARDUINO DIECIMILA BOARD WITH AN AA CELL AS SIZE REFERENCE

First of all, design your hardware and build a prototype target system. This can be as easy as using an on-board LED to experiment with. The Arduino Diecimila has an on-board LED connected to I/O pin number 13. With other boards you might need to wire up an LED as in Figure 2. The I/O ports of the processor can drive an LED directly. I recommend starting with something as simple as an LED to get used to the development cycle and the programming language.

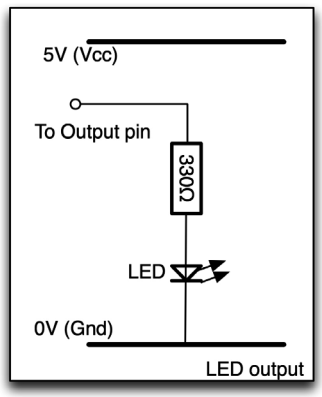


FIGURE 2: CONNECTING AN LED

The next phase is installing the development software on your host. The integrated development environment for Arduino is open source and uses the Processing [13] programming language. Processing looks a lot like C and was developed in a visual arts environment, but has matured to a production-ready development environment. In addition, a gcc toolchain is available to generate code for the Arduino [14]. The easiest way to start is to get the Processing integrated processing development environment for your platform (Windows, Linux, Mac OS X) and work with that. The Arduino IDE supports editing Processing with syntax highlighting and compiling/linking and uploading the code, as well as a simple terminal feature. The terminal can be used to connect to the target board, when you choose to output data using the serial interface. It also features a system to keep track of your libraries. The modern Arduino boards feature a USB interface to do the download and serve as I/O for the serial port, so you may also need to install a driver for the FTDI USB to serial interface.

The processor chip that is mounted on the Arduino board already has a boot loader program in part of the on-chip flash, so there is no need to add

software on the target to communicate with the host machine. If you use an Arduino Diecimila board, the power can be delivered through the USB interface if you set the jumper accordingly, so you are ready to roll.

Now, as the Arduino Diecimila board has an on-board user-programmable LED, a good first program, which is called a *sketch* in Processing, should blink this LED. Therefore we initialize the I/O port, turn on the LED, wait and turn off the LED again, as shown in Listing 1. This code is available as a demo in the Arduino IDE under File -> examples -> Digital -> Blink.

```
/* Blink a LED on the Arduino Using Processing */

int ledPin=13;

void setup() {
  PinMode (ledPin, OUTPUT);
}

void loop(){
  digitalWrite (ledPin, HIGH);
  delay (1000);
  digitalWrite (ledPin, LOW);
  delay (1000);
}
```

LISTING 1: YOUR FIRST ARDUINO PROGRAM: BLINK THE LED CONNECTED TO I/O PIN NUMBER 13

The setup() method runs once, at start of the sketch, and the loop() method runs as long as the board has power or until you download another sketch.

After compiling and downloading the sketch, the board should start the sketch and the LED should blink.

You also can use standard UNIX utilities and a gcc toolchain that has been configured to cross-compile (avd-gcc) and generate code for the ATmega processor, if you do not like the Processing language or the IDE. Downloading code and data is done using an utility called AVRDUDE, the AVR Downloader/UploaDEr [15]. This program reads a hex-file, connects to the virtual serial port behind the USB interface, and speaks the Arduino bootloader protocol, which is standardized by Atmel as STK500 [16]. In addition, this program can use JTAG to program the bootloader in an empty chip.

Another relatively unknown language that supports the Arduino platform is called concurrency.cc [17]. It allows running Occam-like parallel programs on tiny devices.

Interfacing

There are libraries for other input and output devices, as well as a vast number of boards, called *shields* for the Arduino or *plugs* for a JeeNode, that connect to the processor board and have special functions, including Ethernet connectivity, zigbee interfaces, and interesting sensors such as temperature, GPS position, compass heading, barometric pressure, or force to connect your system to the real world.

CONNECTING INPUT

The ATmega processor that is used on the Arduino board has pins that can be configured as digital inputs. The status of these pins shows as the bits in

a word that you can read in software. Standard practice is to pull the port high with a resistor and connect it to ground using the switch (see Figure 3). Also, use a small series resistor in case you make a programming mistake and program the port as output and set it high, so that you do not completely short-circuit it to ground and destroy your Arduino I/O pin.

Small switches can be read this way. An important thing to note here is that if your software polls the switch line, you will often detect multiple opens and closures when just pressing the button once. This phenomenon is called *bounce* and is caused by the mechanical nature of the switch. Building in a small delay when polling a switch will help de-bounce the input reading.

There are also a number of pins that take analog input voltages; the value of the voltage on the pin is shown as the value of a word that can be read in software. These pins use the on-chip analog-to-digital converter.

Loads of sensors nowadays have the analog-to-digital conversion built into the chip. The sensor measures an inherent analog value, such as temperature or barometric pressure, but has output pins that serially output the value using a protocol such as I2C, SPI, or 1-wire [18]. I2C, SPI, and 1-wire are all bus-like systems where you can serially transmit data from and to multiple devices using a small number of lines (wires). All of these protocols have their own software library that can be used in your user programs to communicate according to the protocol.

If you use the sensors with integrated AD conversion you do not need to take a lot of precautions to condition the (often very low) voltage analog signals using sensitive amplifiers and converters. Figure 4 shows a setup of how to connect a I2C device to an Arduino board. Here you see a compass sensor. The I2C library uses two analog input lines as digital inputs.

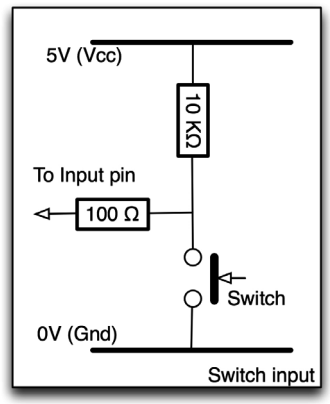


FIGURE 3: SWITCH INPUT

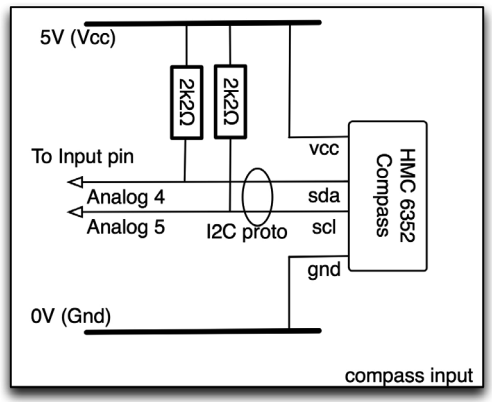


FIGURE 4: A SERIAL CONNECTION TO A COMPASS SENSOR

CONNECTING OUTPUT

The Arduino outputs a standard 0 or 5 volts level at its output pins. These output pins can drive small loads. For larger loads you need to amplify the signal or provide indirect control using a relay. There are special add-on boards for the Arduino that allow you to control small low-voltage motors or LED arrays. There are even small LCD displays with ASCII only or bit-image graphics that are supported by different libraries [19]. Often these displays are connected in parallel to the Arduino board, using four or eight data lines and some control lines connected to I/O ports.

You should be very careful if you want to connect the Arduino board to control a device that is at mains voltage. For this case a device called a solid-state relay (SSR) should be used, separating the low voltage from the mains. This building block accepts standard low-voltage output from an Arduino board and separates it optically, using an LED and a photo (light sensitive) transistor from the part that actually switches 110 or 230 volts. These devices come in a hermetically sealed and insulated housing, so apart from the terminals there are no live voltage-carrying parts outside. Figures 5 and 6 show such a device and the way it is connected. The additional advantage of using such a solid-state relay is that the device switches the load on or off at the 0 volts crossing of the AC mains voltage, reducing noise and power surges.

A note of caution is important here: *Make sure the mains connections are properly insulated and protected from accidental touch.*

FIGURE 5: A SOLID-STATE RELAY

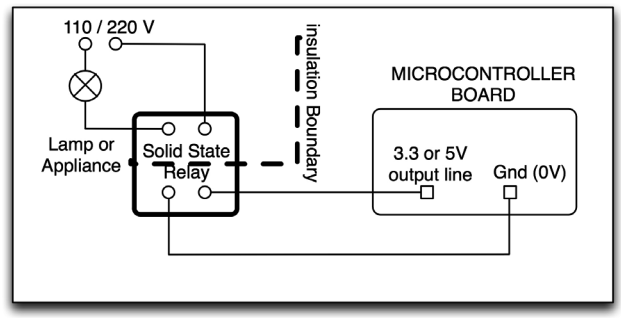


FIGURE 6: SETUP OF A SOLID-STATE RELAY TO CONTROL A HIGH-VOLTAGE DEVICE

CONNECTING TO A NETWORK

For the Arduino boards, there are different methods for connecting to an Ethernet network. All rely on separate boards that add an Ethernet interface. The most commonly used board holds an Ethernet chip (Whizznet W5100) that also implements the low levels of the TCP/IP stack [20]. It communicates over a three-line serial (SPI) protocol to the processor board and is supported by a library to read from and write to the network (address). The advantage of using just three lines is that a vast number of I/O lines of the processor stay available for your own I/O. The disadvantage is that more processor cycles are used to implement the SPI protocol.

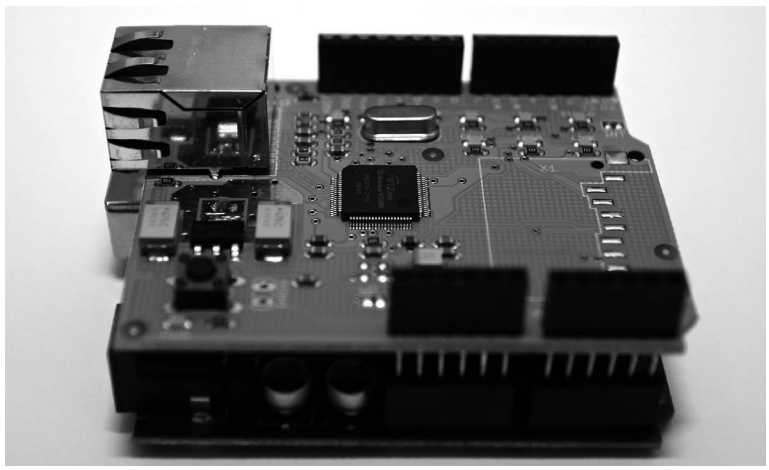


FIGURE 7: AN ARDUINO BOARD (BOTTOM) WITH AN ETHERNET SHIELD (TOP)

Applications

Applications for these small embedded systems are plentiful. They range from small systems such as an alarm clock or a controller board, to control devices such as fans in a server rack or electronic locks, to a more complex system that controls an autonomous robot or a sensor network measuring different environmental parameters at different locations and communicating wirelessly to a master device. Applications are not restricted to the technical field, as the creative crowd has also discovered the power of small controller systems to, for example, build art installations that emit sound or light and work cooperatively [21].

Conclusion

In contrast to the more complex systems we work with every day, there is a complete world of small controllers and computing devices that help us in daily life. Building a hardware-oriented small controller to help you with simple tasks is not difficult. There are a number of modern tools and ready-to-be-used hardware platforms that can be implemented easily, enabling you to build embedded systems with little effort while yielding great results.

ACKNOWLEDGMENTS

Thanks go to Rik Farrow for the comments on the initial draft of this article. A word of thanks also goes to Competa IT, my employer, for giving me the freedom to write this article

REFERENCES

[1] Rudi van Drunen, "A Home-Built NTP Appliance," *;login:*, vol. 34, no. 4, August 2009.

[2] Intel HEX file: http://pages.interlog.com/~speff/usefulinfo/Hexfrmt.pdf.

[3] Toyota: http://developers.slashdot.org/story/10/03/13/1611248/Toyota-acceleration-and-Embedded-System-Bugs.

[4] JTAG: http://en.wikipedia.org/wiki/Joint_Test_Action_Group.

[5] 8051 series: http://www.eg3.com/8051.htm.

[6] ARM: http://www.arm.com.

[7] PIC overview: http://en.wikipedia.org/wiki/PIC_microcontroller.

[8] JAL: http://www.voti.nl/jal/index.html; JALv2: http://www.casadeyork.com/jalv2/.

[9] AVR Atmel: http://www.atmel.com/products/AVR/.

[10] Arduino: http://www.arduino.cc.

[11] JeeNode: http://news.jeelabs.org/2009/03/05/jeenode-v2-pcb/.

[12] StickDuino: http://spiffie.org/kits/stickduino/start.shtml.

[13] Processing: http://processing.org/; http://hardware.processing.org/.

[14] avr-gcc: http://www.symbolx.org/robotics/107-arduinoavr-command-line-dev-environment.

[15] AVRDUDE: http://www.ladyada.net/learn/avr/avrdude.html.

[16] STK500 Protocol: http://www.atmel.com/dyn/resources/prod_documents/doc2525.pdf.

[17] Concurrency.cc: http://concurrency.cc/.

[18] Arduino and Two-Wire Interface: http://www.nearfuturelaboratory.com/2007/01/11/arduino-and-twi/.

[19] LCD Library: http://www.arduino.cc/playground/Code/LCD4BitLibrary.

[20] Ethernet Shield: http://www.arduino.cc/en/Main/ArduinoEthernetShield.

[21] Arduino in Vancouver: http://vimeo.com/9821419.

DAVID N. BLANK-EDELMAN

practical Perl tools: making stuff up

David N. Blank-Edelman is the director of technology at the Northeastern University College of Computer and Information Science and the author of the O'Reilly book *Automating System Administration with Perl* (the second edition of the Otter book), available at purveyors of fine dead trees everywhere. He has spent the past 24+ years as a system/network administrator in large multi-platform environments, including Brandeis University, Cambridge Technology Group, and the MIT Media Laboratory. He was the program chair of the LISA '05 conference and one of the LISA '06 Invited Talks co-chairs. David is honored to be the recipient of the 2009 SAGE Outstanding Achievement Award and to serve on the USENIX Board of Directors beginning in June of 2010.

dnb@ccs.neu.edu

BEING THE TRUTHFUL SORT OF SYS-admin (you know, the kind that would have the System Administration Code of Ethics tattooed on my thigh if the process wasn't a bit on the painful side), I don't usually find myself intentionally fabricating false data. But sometimes that's entirely appropriate. We'll look at a case where this is true and how Perl can help with an endeavor that isn't nearly as nefarious as it sounds.

The scenario I have in mind is where you test something you built or configured with "real" data before you let it get anywhere near the production data. For example, you might want to test a database, a data processing script, a Web application, etc., with pseudo-real data. In those cases, it is easy to hand-create ten or even 100 sample records, but that won't tell you much about how your code or server will handle your *N*-million-record production data. One way to get a comparable data set is to create it yourself, and that's where Perl comes in.

Data::Generate

There are (at least) three Perl modules that can help with this task. We're going to look at all three because they all do things in slightly different manners. This should give you a few options, making it more likely you'll find a good one when you next find yourself in this situation. The first module I'd like to show you is potentially the most flexible out of the box but may, paradoxically, not be the most useful one.

Data::Generate works like this: you give it the type or types of data you want generated (i.e., a string, integer or float number, date, time, etc.), a specification for that data, and the percentage of that type (if more than one type is being requested), and it will attempt to hand you back a data set that satisfies this request. One of the neatest things about this and the other module (Data::Maker) that supports similar functionality is the format of the specification. Data::Generate expects you to provide the specification in something that looks like a subset of Perl's regular expression syntax. We're used to using regular expressions as a selection mechanism; in this case imagine you could run things in reverse and have them produce output instead of filter it. For example, if you had the regular expression:

```
# match all 5-characters strings containing a to z, A to Z and 1, 2, and 3
/[a-zA-Z123]{5}/
```

the equivalent Data::Generate code would be:

```
use Data::Generate;
my $dg_rule       = q { STRING [a-zA-Z123]{5} };
my $dg_generator  = Data::Generate::parse( $dg_rule ) or die $!;
# get a 10 element data set
my $Results       = $dg_generator->get_unique_data(10);
```

That second line of code requests a data set consisting of five character strings using characters from the specified character class. When we run this code, the variable $Results contains a reference to an anonymous array of the specified length (10) with contents like this:

```
x $Results
0 ARRAY(0xb1c8d4)
   0  'UK1k3'
   1  'UK1l3'
   2  'UKNk3'
   3  'Uj1k3'
   4  'UjNk3'
   5  'lK1k3'
   6  'lK1l3'
   7  'lKNk3'
   8  'lj1k3'
   9  'ljNk3'
```

If you are curious how many unique possible values could be generated given your specification, Data::Generate can tell you:

```
print $dg_generator->get_degrees_of_freedom(); # prints '503284375'
```

Earlier on I mentioned that you could ask Data::Generate to produce a data set with multiple types. Here's some example code of that from the documentation:

```
use Data::Generate;

# generate varchar data with 2 kinds of values:
#   -> 36% values like ( 12222,15222, ...)
#   -> 64% values like (AAXQ,BAXQ,...)
my $input_rule = q {
      VC(24)   [14][2579]{4}      (36%)
               | [A-G]{2}[X-Z][QN] (64%)
      };

my $generator  = Data::Generate::parse($input_rule);
my $Data       = $generator->get_unique_data(10);
```

Data::Generate is also happy to take a date specification that will yield ranges of dates, like (again from the doc):

```
my $input_rule = q {
      DATE '1999' 'nov' [07,thu-fri] '09' : '09' : '09'
};
my $generator  = Data::Generate::parse($input_rule);

#   ->   returns a set of date values (format 'YYYYMMDD HH:MI:SS')
#        corresponding to the 7th and all Thursdays and Fridays
```

```
#       of November 1999.
my $Data= $generator->get_unique_data(10);
```

One last cool feature of this module: in addition to standard types such as STRING, INTEGER, VARCHAR, this module also allows you to provide a "filehandle" type. It will then attempt to read the file listed in that type and suck in all of the values in that file. The cool part is that it will only suck in the data values from the file that match the regexp-like specification you provided.

So why isn't this module the bee's knees? Well, it might be for you if you are happy with the random nature of the data it will be returning. If you don't mind operating with data that doesn't look even remotely real, then you may want to brush your palms off a couple of times and just call it a day. But if you are creating a Web application that will display street addresses or people's names, then this module will provide you with Web screens that list people and addresses from Mister Mxyzptlk's dimension.

Data::Maker and Data::Faker

Here's where Data::Maker and Data::Faker come in handy. Both were created (apparently independently of each other) to generate data that is more "realistic" than the random stuff you would get out of Data::Generate unless you were pre-seeding its data. Data::Maker and Data::Faker are pretty similar in their approach. If asked to pick one of the two to use, I'd probably suggest using Data::Maker because it is actively maintained (Data::Faker was last updated in 2005) and more feature-full. I mention both because Data::Faker has a slightly different set of data it is prepared to mock up out of the box. If for some reason you needed that kind of data, and you needed it in a hurry (vs. writing custom code Data::Maker could use), you could consider using Data::Faker instead. Given this preference, let's take only a quick glance at Data::Faker before we explore Data::Maker in depth. To use Data::Faker, you write code like this (example from its doc):

```
use Data::Faker;

my $faker = Data::Faker->new();

print "Name:      ".$faker->name."\n";
print "Company:   ".$faker->company."\n";
print "Address:   ".$faker->street_address."\n";
print "           ".$faker->city.", ".$faker->state."  ".$faker->zip."\n";
```

Data::Faker also has a number of plug-ins, such as Data::Faker::Internet, that expand the kind of data it can produce. For example, if you load it like this instead:

```
use Data::Faker qw(Internet);
```

you would be able to write:

```
# return a fake domain name
print "Domain Name:" . $faker->domain_name . "\n";
# return a fake IP address
print "IP Address: " . $faker->ip_address  . "\n";
```

Let's move on to Data::Maker so we can see how this general idea can be improved. Like Data::Faker, it has a number of plug-in–like modules that ship with the base package. We'll talk about those in a moment.

The Data::Maker interface is a bit different and slightly more complex than the previous two we've seen. Like many other modules, you give it a full

description of just what you want it to do as part of the object constructor (Data::Maker->new) and then use an iterator to request the generated data, one record at a time. Let's start with the specification.

The first thing we specify is the different fields we want our generated record to contain. Like our first Data::Generate example, we can specify this field using a regexp-like description:

```
{name => 'zipcode', format =>'\d\d\d\d\d'},
```

This says there will be a field called "zipcode" whose format consists of five digits. The only other argument we'll want to give is the number of records we hope to generate:

```
record_count => 10000,
```

Once we have a Data::Maker object, we can request data from it:

```
while ( my $record = $dm->next_record ){
  print $record->zipcode->value . "\n";
}
```

In context, the code looks like this:

```
use Data::Maker;

my $dm = Data::Maker->new(
  fields => [
     {name => 'zipcode', format =>'\d\d\d\d\d'}
  ],
  record_count => 10000,
);

while ( my $record = $dm->next_record ){
  print $record->zipcode->value . "\n";
}
```

The use of an iterator here instead of just a function that returns all of the data (like Data::Generate uses) makes a great deal of sense when you start to generate huge data sets. Data::Generate's get_unique_data() is guaranteed to eat memory like it is going out of style if the data set gets big (this is one of several memory-chewing implementation issues it has, perhaps a good reason to avoid it for some people). Data::Maker's next_record() will not have this problem.

You might notice that the code called $record->zipcode->value is used to get the generated zipcode instead of just $record->zipcode. That's necessary because we get back a Field object through which we'll access the value of the specific field. The author of the module tells me he'd like to make the $record->zipcode code return just the value in a scalar context (because that's what people want most of the time), but that's not yet in the module.

If this were all Data::Maker did, it wouldn't be all that interesting. More interesting is its plug-in–like system (although it doesn't call them plug-ins). With this system, you can write modules or use the author's provided submodules to extend the number of data types that are available. Each module provides additional classes that will produce data. Here's a sample of this from the documentation:

```
use Data::Maker;
use Data::Maker::Field::Person::LastName;
use Data::Maker::Field::Person::FirstName;
```

```
my $maker = Data::Maker->new(
  record_count => 10_000,
  delimiter => "\t",
  fields => [
    {
      name => 'lastname',
      class => 'Data::Maker::Field::Person::LastName'
    },
    {
      name => 'firstname',
      class => 'Data::Maker::Field::Person::FirstName'
    },
    {
      name => 'phone',
      class => 'Data::Maker::Field::Format',
      args => {
        format => '(\d\d\d)\d\d\d-\d\d\d\d',
      }
    },
  ]
);

while (my $record = $maker->next_record) {
  print $record->delimited . "\n";
}
```

In this sample, you can see three fields being defined using classes, each being provided by either Data::Maker or a secondary module that is loaded. The third field defined (phone) is very similar to zipcode in our last example except that it is being described in a more explicit way by referencing the class and passing in a specific argument. We're also throwing in another nicety in the data set definition by indicating that we'll want records to be returned (by $record->delimited) with fields separated by a tab character.

Data::Maker (as of this writing) ships with additional field types that include dates/times, first/last names, middle initials, social security numbers, "random" text (Lorem Ipsum, courtesy of Cicero), and file contents (so you can pre-seed the data). To make this even cooler, the author has also provided data types that can depend on other fields. For example, if you use code like this in your field definition:

```
{
  name => 'initials',
  class => 'Data::Maker::Field::Initials',
  args => {
  from_field_set => [ 'firstname', 'lastname']
}
```

the initials field will be filled in based on the first name and last name fields. If you use the Data::Maker::Field::Person::Gender class, it will attempt to guess in a mechanical fashion the gender of the fake person based on their first name. Anyone who has any experience with gender issues knows that humans can't get their act together around gender, so I wouldn't expect very much from that guess. Still, it is cool that the module makes provision for creating internally consistent fake data based on relationships between fields. It is pretty clear after talking to the module author that he is dedi-

cated to improving the module and adding more excellent features to it in the future.

Now that you have a cool tool for creating a bunch of fake data, let's draw this column to a close. Take care, and I'll see you next time.

PETER BAER GALVIN

Pete's all things Sun: the Exadata V2 architecture and why it matters

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

pbg@cptech.com

EVEN BEFORE ORACLE CONSUMMATED its purchase of Sun, the two companies announced their first combined product, the Exadata V2. The product was announced in October 2009 and started shipping in March of 2010. This "appliance" is designed to execute both OLTP and data warehouse operations, either individually or concurrently. It is based on Sun hardware and runs the Oracle Linux operating system, the Oracle 11GR2 database, RAC clustering, Oracle Exadata software, and, optionally, other software components.

The Exadata V2 is interesting on its own merits and provides an implementation of Sun and Oracle best practices for optimal database performance that is worth exploring. Exadata V2 becomes compelling to study when it is viewed as the first of probably many applications from the combined Oracle/Sun company.

In this column I will describe the various aspects of the Exadata V2 architecture from both points of view—database implementation best practice blueprint and first of a likely generation of Oracle/Sun appliances. There are several surprises along the way, including outstanding (claimed) performance, ease of migration, and, under some circumstances, low cost of adoption.

Exadata Background

The first Exadata came out in September 2008. It was based on HP hardware, ran Oracle software, and lacked some of the innovations found in Exadata V2. In fact, its lower performance made it appropriate only for data warehousing (DW) uses, not OLTP, pitting it head-to-head against other DW appliances such as those from Teradata and Netezza. Oracle has not published information on how many Exadata sold, but there is a clue in their product decisions. Approximately 12 months after Exadata V1 was announced it was put out to pasture in favor of Exadata V2, and Oracle terminated all sales of Exadata V1 [1, 2].

Oracle seems to be greatly emphasizing V2, with a large announcement, advertising, and many informational events. As of Oracle's financial statements covering its third quarter (ending February 2010), Oracle expects to sell $100m worth of Exadata in its fourth quarter [3]. It appears that Oracle is very

serious about Exadata V2 and expects strong sales, but does the architecture support such exuberance?

Exadata V2 Appliance

Fundamentally, Exadata V2 consists of several components in several "supported" configurations. It *is* an appliance in that it comes pre-built from the factory and only takes a few days for the on-site hardware and software configuration. It is also an appliance in that there are only certain configurations that are supported. Other custom configurations are possible, but a custom Exadata V2 has less support available than the standard configurations. That is, the components are supported, but a customer can no longer call the Oracle 800 number and say "my Exadata V2 is slow" and have Oracle handle the debugging and optimizing. Also in that lesser-support category are solutions for the Exadata V2 not blessed by Oracle. For instance, it is possible to use the InfiniBand interconnect infrastructure for data access or for backups, but that again makes a configuration "custom" (although it appears that Oracle might now be supporting InfiniBand use for backups).

Hardware Architecture

There are two major hardware components to the Exadata V2. The Database Machine is the server node where the database runs, and it accesses data stored in the Storage Server. The servers are interconnected by a redundant 40Gb/sec InfiniBand network. Application access to the appliance is via the database servers' multiple, redundant, front-end 1Gb NICs.

Exadata V2 comes in "basic system," "quarter rack," "half rack," and "full rack" versions. Further, you can add up to seven full racks to any of the rack configurations to create a really, really big database server. The "basic system" is just a database server and a storage server, designed for development or QA rather than production use. A quarter rack of Exadata V2 includes two database servers and three storage servers. It can be upgraded to a half or full rack. The half-rack includes four DB servers and seven storage servers. The full rack includes eight DB servers and 14 storage servers.

Each database server is a Sun x4170 with 72GB of memory, dual 146GB SAS boot disks, hardware RAID, two four-port QDR (Quad Data Rate) InfiniBand ports, four Gigabit Ethernet ports, and one Ethernet ILOM management port. Each provides two quad core Intel Nehalem CPUs. The storage servers come in two flavors. One has SAS drives for maximum performance, while the other has SATA drives for maximum storage capacity. Both versions are based on the Sun x4275 server with two quad core Intel Nehalem CPUs, 24GB of memory, 12 drives, and four 96GB Sun Flash Accelerator F20 cards. Each also has hardware RAID, two four-port QDR InfiniBand HBAs, four Gigabit Ethernet ports, and one Ethernet ILOM management port. Each storage node has 384GB of "smart flash cache" storage capacity. The flash memory is not in the form of disk drives, but, rather, plugs into the PCI bus slots for improved throughput. The SAS nodes provide 7.2TB (twelve 600GB 15K RPM drives) of raw storage, while the SATA nodes provide 24TB (twelve 2TB 7.2K RPM drives).

A full rack of Exadata V2 therefore provides 5.3TB of Smart Flash Cache and 100TB of SAS or 336TB of SATA (or a mix of the two) raw storage. The storage is by default configured to be striped within a storage server and mirrored between storage servers (via ASM, Oracle's Automated Storage Management software), yielding 28TB SAS or 100TB SATA of usable storage. According to Oracle, a full rack of the Exadata V2 appliance provides 21GB/

sec of disk I/O, 50GB/sec of flash I/O, and one million I/Os per second aggregate. Oracle initially advertised benchmark results, but those numbers were unaudited and Oracle was forced to stop publicizing them until the audit could be completed. In the meantime we have only Oracle's promise that the full rack can perform millions of database transactions per minute and tens of millions of queries per minute.

Software Architecture

The blueprint of how to architect an optimal database server is certainly thought-provoking. Rather than take the path of some vendors of configuring flash disks as a fast tier of storage, Oracle/Sun is continuing to forge a role for flash as a new tier of cache. By definition, a new storage tier would be faster and more expensive than the lower tier while being slower and lower-cost than the upper tier. That is exactly the case for flash memory. Such a change in storage hierarchy hasn't taken place in dozens of years, so expect a profound effect on price/performance as flash is integrated into hardware, operating systems, and applications. Within Exadata V2, the flash memory is used as an LRU cache by default (with optimizations to avoid sequential I/O from evacuating the cache). However, the software allows for tuning of the use of the cache, including pinning tables into flash (in essence treating it like fast disks) and preventing tables from being cached in flash. This approach makes a lot of sense and should be a model for other vendors (or customers) to follow.

Other aspects of the Exadata V2 architecture also align with fundamental best practices. Oracle RAC provides clustering for performance and availability. Database instances need not be RAC-clustered, but by default they are. Intel Nehalem CPUs provide great performance in small footprint servers. Rather than have a central storage unit (and central I/O bottleneck), the storage is grid-shaped to match the database server grid architecture. The database server talks to the storage server using iDB (Intelligent Database protocol) and offloads operations to the storage server using that protocol. iDB is itself based on RDSv3 (the industry-standard Reliable Datagram Sockets protocol). The interconnect between the nodes is low-latency, high-throughput InfiniBand. The Exadata V2 runs Oracle's Enterprise Linux (essentially Red Hat Enterprise Linux but supported by Oracle), which has a lot of RDP optimization built in.

Added to these standard components are some Exadata V2–only ones. Hybrid Columnar Compression (HCC) is a new compression technology. It avoids the problems of other compression methods, in essence allowing data to be compressed and still used for OLTP transactions rather than being limited to just data warehouse operations [4]. This form of compression is especially important considering that data warehouses can use much more storage than OLTP databases and that the Exadata V2 has fairly low storage capacity options. If the HCC technology is as effective as Oracle claims, in many cases compressing data 10x and sometimes even 50x, then this compression can provide effectively very large storage capacities. Further, the HCC pertains to flash memory as well as on-disk, potentially greatly increasing the flash capacity. Also included is "Exadata Storage Server Software." This new package configures, manages, and monitors the storage nodes (including flash memory use). One final Exadata V2-only feature is "Exadata Smart Scan" [5]. This optimization offloads certain operations to the storage servers, allowing them to perform operations on their stored data and return only the results, rather than returning bulk data to the database servers for

them to sift through. Operations that can be offloaded (and are by default) include table scans, join filtering, backups, and tablespace creation.

Analysis

The blueprint provided by the Exadata V2 is certainly solid. There are a few surprises and conclusions that are worth noting as well:

- Existing Oracle licenses are transferable to Exadata (including Oracle DB, RAC, and Partitioning). That can greatly reduce the cost of an Exadata that is being used for database consolidation, for example.
- The Exadata looks to be an excellent consolidation engine. Included with the Exadata software are resource management tools that can, for example, give some databases resource priority over others. These tools also allow the use of the flash storage to be fine-tuned, pinning specific tables into flash or letting Oracle use the flash as an extended cache.
- The Exadata V2 is designed to be able to perform OLTP and data warehouse transactions concurrently. If a single system can be used both ways, consider the implications compared to stand-alone, separate data warehouse solutions. Normally data must be extracted from the OLTP system, copied to the DW system, imported there, and then processed. The extraction and copying are overhead, on both the OLTP and DW systems. And any reports or queries on the DW system are performed against "stale data," data from the time the extraction started. Now consider being able to do DW operations against live, current OLTP data. And according to the performance numbers published by Oracle, those operations could run much faster than on most DW systems. That speed could result in completing more complex reports, allow more ad hoc queries, and so on. Such a change could be a fundamental advantage to DW consumers (finance and senior management, for example).
- Consider the cost of Oracle database software licenses. Now consider the hardware on which they run. Increasing the performance of that software gains your site more database performance at the same database license cost. The Exadata V2 is optimized to run OLTP and data warehouses very quickly. The resource management software included with the Exadata and its use as a consolidation engine probably lead to the appliance running with more databases using more resources and with less reserved headroom than having a non-Exadata database environment. That means that, for a given number of Oracle database licenses, your site would get more database performance.
- As mentioned above, customization of the pre-defined Exadata V2 configuration is allowed. For example, if your business required fewer database engines and more storage, it would be possible to get such a configuration from Oracle. Also, some sites might want to use the included Infiniband interconnect for fast backup of the data. However, the support model for custom configurations is likely to be different from the pre-defined ones. At the moment, even splitting a full rack of Exadata V2 into two racks (to prevent the rack from being a single point of failure) is a custom configuration.
- You can't build your own Exadata V2 system. Even though the hardware components of Exadata V2 are off-the-shelf Sun servers and networking, there is "magic sauce" in the Exadata. The Exadata storage software manages the storage nodes; the Exadata servers offload storage-centric operations to the storage nodes (again increasing the database performance you get with those Oracle licenses); and "Hybrid Columnar Compression," a new method for compressing columns of data while still making them available for OLTP access, is an Exadata V2–only feature. Following the Oracle/Sun

best practices and blueprints and using the same hardware components, could lead to something similar to the Exadata V2 in terms of features and performance, but the lack of those features means that it will not match the features and performance of Exadata V2. Further, the appliances are delivered pre-configured, and once the delivery team hands it over to the customer (a few days' effort), Oracle 11GR2, RAC clustering, InfiniBand configuration, and storage layout are all in place and performance is pre-tuned. Getting that to work from scratch on a build-your-own database server can add many weeks (and some risks) to a project. That should be considered when a datacenter is considering the buy-vs.-build decision.

The choice of SPARC vs. x86 and Linux vs. Solaris is difficult to interpret. Exadata V1 was based on the same technologies, Intel and Linux, so the shortest path to a new release was staying with them. And Oracle has stated that they are enthusiastic about SPARC and Solaris and plan to invest more in them than Sun did, so I don't believe there are any hidden messages in these selections.

Conclusions and the Future

Until real-life field use experience with Exadata V2 appliances is gained, it is difficult to determine what the future will hold. I believe the Exadata V2 will be a successful offering from Oracle/Sun, given the compelling architecture and claimed break-through performance. Exadata V2 does seem to be a great marriage of Oracle and Sun and bodes well for future combined-technology products. They will certainly be worth considering, and if the price, performance, and feature set of each one make sense for a given datacenter, the datacenter could also gain from deployment cost savings and the simpler support allowed by calling one vendor rather than many. When I think of the potential cross-pollination of features between Oracle and Sun, the excitement increases. Just imagining Sun 7000-style DTrace-based analytics being applied to other types of applications (databases or virtualization, for example) makes me hopeful for the marriage of Oracle and Sun. Time, as always, will be the final arbiter of whether this is the reality or whether some lesser one comes to pass. In the meantime, watch the blogs for breaking information [6, 7, 8, 9].

REFERENCES

[1] http://www.oracle.com/us/products/database/exadata/index.htm.

[2] http://www.reuters.com/article/idUSTRE58E80D20090915.

[3] http://www.theregister.co.uk/2010/03/26/oracle_q3_f2010_numbers/.

[4] http://www.oracle.com/technology/products/bi/db/exadata/pdf/ehcc_twp.pdf.

[5] http://www.oracle.com/technology/products/bi/db/exadata/pdf/exadata-technical-whitepaper.pdf.

[6] http://kevinclosson.wordpress.com/.

[7] http://structureddata.org/.

[8] http://blogs.oracle.com/databaseinsider/.

[9] http://ctistrategy.com/.

DAVE JOSEPHSEN

iVoyeur: pockets-o-packets, part 1

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

dave-usenix@skeptech.org

THE WIND IS EVERYWHERE, A CONSTANT roar that threatens to rip my hat off my head and send it tearing 500 feet into the canyon below. Even the birds can't seem to fly in it, so I'm surprised by the shadow cast by the turkey vulture that's now circling us above the pass. It's the first bird I've seen aloft in days, and it's obviously having a hard time remaining so. We're nearly to the top of the Caprock Canyons south prong when my pager goes off. I glance up at my wife who, thankfully, didn't hear it through the wind.

I take the pager from my pocket and have a quick look. Office users are complaining of network slowness; I should ignore them. This is supposed to be a day off in a place 300 miles distant. A place of rocks and juniper. But a small voice in the back of my head says simply, "But you can." It's true . . . here in the middle of a scramble up to the ridgeline and despite this infernal wind, it's possible for me to diagnose network latency at the office. I can. The thought simultaneously amuses and offends me.

I bring up the ssh client on my pager, ssh to the pcap box, and run a quick racluster [1] command. Seeing the result, I hastily type my reply ("Tell Larry to stop downloading My Little Pony episodes"), just before my wife looks down and asks if I'm okay.

"Yep," I shout in reply.

"Tell me you're not on that pager," she yells, raising an eyebrow.

"I'm not," I shout, returning it to my pocket.

As we continue our hike my mind mulls all the pieces that make what just happened possible, from the cellular infrastructure back on down to Ken and Dennis. So much work. It occurs to me to wonder if it's a miracle or a curse. I suspect the latter but can't say for sure. One thing I do know, however, is good article fodder when I see it, so let's spend the next couple of issues talking about the pieces of a decent packet capture framework.

Collecting IP packets for offline analysis is a bread-and-butter sort of monitoring infrastructure. If you aren't centrally collecting packets, you probably have lots of little tools that work for a single type of device and can tell you only about a small piece of the network. One gains a lot from centralizing packet and flow data. IDS/IPS, network utilization

and trending, and a slew of other buzzword activities become greatly simplified, and efforts that used to be complicated and intertwined can be made to function to each other's benefit.

For example, Snort IDS alerts are great [2]. They catch all sorts of questionable network behavior, but without a centralized packet repository and the tools to analyze it, these alerts lack context. For example, is it in fact aberrant that server A conducted what appears to be a replay attack against server B, or is that just what happens the first Wednesday of every month because of some weirdo backup software doing weirdo things? If the packets are localized in such a way that Snort and Argus [1] can both use them, then you don't need to spend hours running down context behind Snort alerts (if you can at all).

Having an offline packet repository can really be a life-saver in all sorts of ways. Getting DoS'd and can't log into the router? Ask Argus. Broken project planners asking for answers by tomorrow to questions that take months to answer? Ask Argus. Ex-girlfriend who also happens to be the head NOC sysop ignoring your BGP looking-glass RFIs? Ask . . . well, you get the point.

The easiest way to collect IP packets in a central location and to ensure that the greatest number of tools can make use of them is, in my opinion, to simply redirect them all to a single interface (or series of interfaces for larger environments) on a single host. This can be a bit of a trick, but it can be done, and once you're there, you're golden, because pretty much any tool designed to analyze packet traffic can listen to a named interface.

Normally, the goal is to capture any packet that traverses a network segment, and for most folks there are three ways to do that. Assuming you use Cisco gear or something else that supports netflow, you could use flow data instead of raw packet dumps and export the flows to a pcap box. Several tools, including Argus, can read flow data, but this does limit your options later on and incurs a bit of utilization on the router or firewall in question.

Next, you could use a span port on the switch. Span ports are great; you get real packets, you can consolidate packet dumps from several devices to one port, and they don't cost anything extra. Their primary disadvantage is that they may impact the performance of the switch, and this is highly architecture-dependent. A breakdown of span port impact on performance for various Cisco switch architectures may be found at [3]. If you have a mid- to high-end Cisco switch, you're fine.

The third and most expensive is a hardware network tap. These are really great; they're inserted between a device and the switch and provide a duplicate of every packet on a separate port (or set of ports). We use aggregating taps from NetOptics [4]. They're rack-mountable boxes that can tap multiple 10/100 links and aggregate them all to a single 1gbps link. They fail open, so they're not a point of failure if something happens to them (short of physical explosion). There are much larger, much more expensive taps [5] that can aggregate multiple gbps interfaces, for ISPs and very large environments (you guys know who you are), but in my experience it's easy to overestimate what you actually need here. Most environments, surprisingly, can get their pcap traffic down to a single interface on a single box without much trouble.

If you use software routers, then you have an additional option: a software tap. We use OpenBSD routers quite a bit, and I very much like daemonlogger [6] for this purpose. Daemonlogger, written by Marty Roesch (who also wrote Snort), can be thought of as a daemonized tcpdump. It listens to a network interface and either logs the packets to disk or sends them to a remote machine. Daemonlogger comes in handy on the pcap machine too,

since it's likely the pcap machine will need to be plugged into a combination of span ports, network taps, and other devices. Daemonlogger can be used to consolidate all of these interfaces by starting an instance per interface and telling them all to forward to the same interface. It also comes in handy for those super-sensitive boxes for which it's not good enough to only capture traffic that traverses a network segment. If you need every packet that a given database server sends and receives, Daemonlogger is a great way to go.

Since our goal is to aggregate all of our captured packets to a single interface, we should put some thought into that interface. The DAG network cards from Endace [5] are just the thing here. They're expensive but are generally considered to be the best available for pcap and network audit work [7].

So now that we have packets from span ports or flow data or taps or all of the above coming to our pcap box, what do we do with them? There are many answers to this question, but the first three that spring to mind are Argus, Snort, and Daemonlogger (yet again). And none of these are mutually exclusive; all of these tools can listen to the same port at the same time and get what they need, provided the machine has the horsepower to run them. Some other popular answers, in no particular order, are Wireshark [8], NTOP [9], and Bro [10].

I mention Daemonlogger here again because in its disk-logging mode it writes binary pcap files, just like tcpdump, and has built-in options for log file naming and rotation, so it's a great way to provide a lowest-common-denominator online archive of pcap data. Every tool that works with packet-data supports this format, and Daemonlogger is so lightweight it's just about free.

Snort and Bro are both awesome IDS tools that will do a bang-up job listening to the pcap interface. In our setup, Snort is configured to listen to the pcap interface and alert via syslog.

Argus describes itself as a Real Time Flow Monitor that is designed to perform comprehensive data network traffic auditing. In my opinion it's about the coolest network-centric monitoring tool that was ever invented and the entire reason this infrastructure should be built.

Argus may be run as a daemon, reading live packets from a network interface, or as a user program, reading packets from a packet capture file. The default behavior is to run as a daemon, which is what we're interested in here. Point the Argus daemon at your pcap interface, and it'll read in and transform the incoming packets into stream data which it then stores in a database. To analyze the data you need the Argus client ra ("read Argus"), which is available as a separate Argus-clients package on most OSes and distros.

The client programs may either read from an Argus server's data files on localhost, or from a remote Argus server if the server has been set up to listen to remote requests. If you want Argus to listen over the network for client requests, simply pass it a `-P` switch. There are obvious security ramifications to doing this, so Argus may be compiled with SASL support to provide authentication and authorization.

Ra has several partner programs, the two most important being rasort and racluster. There are also a slew of third-party Argus clients to do everything from logging to graphing and visualization. The sky is the limit with the Argus clients; you can interact with them to discover anything you might want to know about the network traffic contained within them. Questions such as, "How much data was transferred in the last 20 min?" "Who are the top 10 users of the bittorrent ports?" and "What hosts are trying to infect

other hosts with virus X?" are all straightforward queries to racluster and rasort.

Stay tuned for much more detail on Argus in my next article, including file management basics and a primer on using ra. Between you, me, and the vulture, the racluster command I typed on the south prong trail was:

```
racluster -M rmon -m saddr -r <my_data_file> - ip | rasort -M bytes -r - -w - | ra -N 10
```

. . . or, in English, "Give me a list of the top 10 bandwidth users sorted by byte-count in the last hour."

Take it easy.

REFERENCES

[1] Argus Real-Time Flow Monitor: http://www.qosient.com/argus.

[2] Snort IDS: http://www.snort.org.

[3] Catalyst Switched Port Analyzer (SPAN) Configuration: http://www9.cisco.com/en/US/products/hw/switches/ps708/products_tech_note09186a008015c612.shtml.

[4] NetOptics hardware taps: http://www.netoptics.com/.

[5] Endace high-speed packet capture devices: http://www.endace.com/high-speed-packet-capture-hardware.html.

[6] Daemonlogger: http://www.snort.org/users/roesch/Site/Daemonlogger/Daemonlogger.html.

[7] Endace recommendation details: http://www.qosient.com/argus/sensorPerformance.htm.

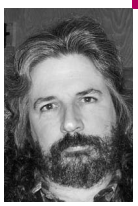
[8] Wireshark Packet analyzer: http://www.wireshark.org.

[9] NTOP Network Management framework: http://www.ntop.org/news.php.

[10] Bro IDS: http://www.bro-ids.org.

ROBERT G. FERRELL

/dev/random: five common misconceptions about information security



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

rgferrell@gmail.com

1. TELNET IS INSECURE.

Telnet is just another protocol. It is no more "insecure" than leaving a loaded revolver with no safety in the nursery is "irresponsible." If you're worried about passwords being transmitted in clear text, just disable authentication altogether. Problem solved. Stop sniffling, you big baby. Port 23 is your friend.

2. Botnets will steal your identity.

Worrying about your identity being stolen by malicious software on your computer is perfectly valid. The malicious software in question will not be installed surreptitiously as a result of an ill-advised visit to a hacked Web site, however. It will be installed by you or your computer's retailer in the form of a Web browser. The likelihood is that the culprit will obtain your information from a "trusted" company's hacked database or stolen laptop/backup media/USB device, not some vast network of hapless zombie computers. Botnets run a distant second to misplaced trust and good ol' fashioned physical larceny.

3. Increasing the number of characters in your password makes it more secure.

The concept of diminishing marginal returns, which I vaguely recall from a macroeconomics class in college in 1978, has never been more clearly demonstrated (does it bother you that "demonstrate" contains "demons"? It does me) than with institutions who keep jacking up password length in the badly mistaken belief that this increases security. Human beings, especially those brought up in a 60-spasmodically-disjointed-images-in-a-30-second-commercial world, have considerable difficulty retaining anything longer than three or four characters. This memory shortfall is further exacerbated by the ubiquity of portable data storage devices and speed dial lists. Presented with a 12 or 14 member string of more or less arbitrary characters they are told they must regurgitate in order to log on, it is absolutely guaranteed that the vast, vast majority of people will at some point commit that password to paper or non-volatile electronic memory. Unless this written record is now scrupulously stored in a safe or equivalent environment at all times, the security of that system just plummeted precipitously. Add to that the fact that it has recently been shown that 14-character password hashes meeting industry standard complexity requirements can be broken by optimized Rainbow Tables in under 6 seconds on a mediocre processor, and it should be fairly apparent that

relying solely on longer passwords in fact dramatically decreases data confidentiality. Oh, and just for the record, "two-factor authentication" does *not* mean "user ID and password."

4. Antivirus software will keep my computer secure.

This is a load of fetid dingo kidneys, to borrow one of my favorite phrases from the late Douglas Adams. I will clarify my point with a metaphorical examination of exactly how signature-based antivirus software works. Let's say you're a contracted bouncer at a popular club. You have a list of the names of people not allowed to come in and, because we're really trying to be secure here, a physical description of each. Now, so long as the people you're trying to keep out don't give you a false name and change their appearance, this filtration system works pretty well. However, it is in the nature of people who are likely to end up on a "no-entry" list to be duplicitous, so the efficacy of this approach is somewhat less than optimal. Not to worry, though—you can watch the guests and eject any whose actions are suspect. Or, rather, you could if that part of your behavioral repertoire hadn't been disabled by an employer who makes most of its money from selling no-entry list subscriptions to clubs. If you can simply throw out the bad apples, why would club management need to spend money on a new list of miscreants every week?

To recap, antivirus software keeps your computer secure so long as it only encounters well-known malware that makes no effort to disguise itself. This practice goes hand in hand with the usual operating system–supplied firewall that blocks everything except the email and Web traffic where 99% of all malicious software of concern to the average user originates. Congratulations, your illusion of security is now complete. Don't forget to take your blue pill every morning.

5. A Nigerian government official wants to give you money.

I honestly thought this threat would go the way of smallpox and the American ivory-billed woodpecker as an increasingly connected world facilitated the widespread dissemination of warnings thereto appertaining. No one ever lost money overestimating human greed and ignorance, though. Repeat after me: I will never, ever, under any circumstances be asked to help an actual corrupt official of an actual sub-Saharan African nation launder 37 million dollars, for one simple reason: they don't have that kind of money. Even if they did, it would be going to buy private jets and Mediterranean vacation homes for ruthless dictators, not sitting unnoticed in some forgotten bank account waiting to be slipped quietly to an avaricious idiot in the U.S. Even Third World nations have government auditors. The same goes for international lotteries based on random email addresses, scam victims reimbursement funds, intestate wealthy persons tragically killed in transportation disasters, and kindly old women dying of cancer who want total strangers to invest their sizeable fortunes in charitable causes on their behalf. The single most useful dictum I learned in that college economics class was TANSTAAFL: *There Ain't No Such Thing As A Free Lunch*. Ferrell's First Law of Fiscal Dynamics states that money does not fall from the sky, nor does it flow freely from regions of low concentration to regions of high concentration without the application of reverse monetary osmosis (also known as international commerce). Large sums of money are Luddite in nature: they tend to announce themselves with certified snail mail, not SMTP. If an offer seems too good to be true, at least you're paying attention.

Keeping computers reasonably secure is akin to fending off flies in defense of a dung pile. The job is a lot easier if you patch those gaping holes in your fly swatter.

And hold your nose.

book reviews

BRANDON CHING AND SAM STOVER

THE SECOND LIFE GRID: THE OFFICIAL GUIDE TO COMMUNICATION, COLLABORATION, AND COMMUNITY ENGAGEMENT

Kimberly Rufer-Bach

Sybex Publishing, 2009. 368 pp.
ISBN 978-0470412916

REVIEWED BY BRANDON CHING

To the uninitiated, Second Life can be somewhat of a mystery. Most people that I have talked to about Second Life have heard of it, maybe have even tried it once or twice, but generally don't know much beyond that. While Second Life can serve whatever use the individual user wants it to, organizations from academia, government, business, and the nonprofit sector have for years been trying to gain a foothold in one of the most popular virtual worlds.

Unfortunately, these organizations have struggled to maintain a meaningful presence in Second Life and the challenges they have faced have been dynamic and difficult to identify. Enter Kimberly Rufer-Bach, whose *Second Life Grid* was written to identify and help organizations surmount these challenges.

The book has three parts and fifteen chapters in all. Part 1 introduces Second Life and outlines the effects that government, nonprofits, and educational institutions can achieve in-world. These chapters are full of real examples of organizations creating a successful presence in Second Life. While a little light on practical advice, this section is full of resources that any organization new to Second Life should find useful.

Part 2 covers in-world cultural issues and guides the reader through merging organizational culture with the norms and customs of the Second Life world. I found this to be the most important section of the book. Any organization embarking on a new venture needs to know as much as it can about the environment it is entering. The author provides valuable insights into topics such as etiquette, communication, avatar appearance, and current events.

Finally, Part 3 is where you get into the nitty-gritty of building your presence in Second Life. Setting up a virtual office, running an event, marketing, and resource management are all well covered. The remaining six chapters in this section take up roughly half the book and are loaded with expert information. While the technical details of content creation are largely absent from this text (as is appropriate), I cannot begin to express the level of excellent administrative and planning information these chapters contain.

All told, Rufer-Bach is certainly a Second Life expert, and this fact shows itself through the amazing amount of nuanced and valuable information packed into this book. Weighing in at nearly 370 pages, with a relatively small font and covering a very broad topic, this book is not for the faint-of-heart, but don't kid yourself by thinking that you'll be an expert at running a successful event/presence in Second Life after reading this book. Many of Rufer-Bach's recommendations revolve around simply getting your hands dirty and learning first-hand what goes on in Second Life.

While this book is targeted towards organizations seeking an in-world presence, it would also be of value to individuals looking to more fully understand the revolutions that are virtual worlds.

CLOUD SECURITY AND PRIVACY

Tim Mather, Subra Kumaraswamy, and Shahed Latif

O'Reilly Media, 2009. 336 pp.
ISBN 978-0596802769

REVIEWED BY SAM STOVER

As everyone knows, "The Cloud" is the next big thing, but since security always seems to lag behind Big Things, I was pleasantly surprised to find a decent primer on the subject. Weighing in at only 330 some-odd pages, the book seems a bit slight at first, but the authors do a good job within that footprint.

Chapter 1 is a brief, and I mean *brief*, introduction to the topic and a description of how this all came to be. It goes over how ISPs evolved into colos, then ASPs, which set the stage for cloud computing. Chapter 2 goes into the aspects of different

types of cloud computing, as well as some of the players in the game (Google, Amazon, Microsoft, etc.). The SPI framework is introduced: Software as a Service, Platform as a Service, and Infrastructure as a Service (SaaS, PaaS, and IaaS, respectively). Lots of technical jargon here to digest, but more importantly, this sets the stage for Chapter 3, "Infrastructure Security," Chapter 4, "Data Security and Storage," Chapter 5, "Identity and Access Management," and Chapter 6, "Security Management in the Cloud." These four chapters are the meat of what was interesting to me (but surely the audit and policy wonks will jump quickly to Chapter 8, "Audit and Compliance"). Chapter 7, "Privacy," ties in very nicely with the security issues presented in Chapters 3–6.

Being a network guy, I was particularly drawn to the "Infrastructure Security" chapter. The risks are broken down into familiar groups: data confidentiality and integrity, access control, and availability. The cloud-specific spin takes into account the difference between normal network "zones/tiers" and domains. The usual suspects include cleartext HTTP communications between the client and the provider, improper IP caching/reusability, BGP prefix hijacking, and DNS attacks. Also, as with anything Internet-connected, DDoS is always a potential threat. Not content with simply enumerating threats, the authors spend half of the chapter going over security countermeasures for hosts and networks, specific to the different aspects of the SPI framework. This all makes for good reading.

Chapter 9 starts to pull everything together by presenting eight different cloud providers and laying out their offerings. Amazon (IaaS), Google (SaaS, PaaS), Microsoft Azure (PaaS), Proofpoint (SaaS, IaaS), RightScale (IaaS), Salesforce.com (SaaS, PaaS), Sun Open Cloud Platform (all), and Workday (SaaS) are all briefly presented and, to a small degree, compared. I would like to have seen a little more depth in this particular chapter, but in some cases (e.g., Sun), the offerings are pretty new, so it will take some time for everything to fall out. Plus, if I want to be a stickler, this is really a book on *security*—there are plenty of other resources out there if I wanted to learn more about cloud providers and their core competencies.

Chapter 10 broaches the topic of Security as a Service (another SaaS). To this point, a lot of security issues have been discussed, from both the provider and the client sides, using the cloud. Security as a Service, however, can be divided into two main groups: InfoSec vendors who are migrating or encompassing delivery methods utilizing cloud mechanisms, and companies who provide "security only as a cloud service, and do not provide traditional client/server security products for networks, hosts, and/or applications." This was probably my least favorite chapter, but YMMV.

Chapter 11 deals with "The Impact of Cloud Computing on the Role of Corporate IT," and Chapter 12 rounds out the book as the "Conclusion, and the Future of the Cloud." Three appendices—a SAS 70 Report, a SysTrust Report, and "Open Security Architecture for Cloud Computing"—complete the work. The OSACC is a very interesting model devised by the Open Security Architecture group (http://www.opensecurityarchitecture.org/), which attempts to "illustrate core cloud functions, the key roles for oversight and risk mitigation, collaboration across various internal organizations and the controls that require additional emphasis." Whew, that's a mouthful, but it's an interesting read nonetheless.

Overall, this is a solid book both for security folks who want to learn more about cloud computing and for cloud computing users who want to learn more about the security behind the technology. It's pretty obvious that the authors are both passionate and knowledgeable, which is great to see in any book. There's plenty here to learn from, and I sincerely hope that this team of authors keeps putting out new editions as the cloudscape changes.

USENIX notes

USENIX MEMBER BENEFITS

Members of the USENIX Association receive the following benefits:

FREE SUBSCRIPTION to *;login:*, the Association's magazine, published six times a year, featuring technical articles, system administration articles, tips and techniques, practical columns on such topics as security, Perl, networks, and operating systems, book reviews, and summaries of sessions at USENIX conferences.

ACCESS TO ;LOGIN: online from October 1997 to this month: www.usenix.org/publications/login/.

ACCESS TO CONFERENCE VIDEOS: See http://www.usenix.org/publications/multimedia/ for a listing of all videos, or http://www.usenix.org/events/byname/ for the the Web sites of the conferences you're interested in.

DISCOUNTS on registration fees for all USENIX conferences.

SPECIAL DISCOUNTS on a variety of products, books, software, and periodicals: www.usenix.org/membership/specialdisc.html.

THE RIGHT TO VOTE on matters affecting the Association, its bylaws, and election of its directors and officers.

FOR MORE INFORMATION regarding membership or benefits, please see www.usenix.org/membership/ or contact office@usenix.org. Phone: 510-528-8649

USENIX BOARD OF DIRECTORS

Communicate directly with the USENIX Board of Directors by writing to board@usenix.org.

PRESIDENT

Clem Cole, *Intel*
clem@usenix.org

VICE PRESIDENT

Margo Seltzer, *Harvard University*
margo@usenix.org

SECRETARY

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

TREASURER

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

DIRECTORS

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

Gerald Carter,
Samba.org/Likewise Software
jerry@usenix.org

Rémy Evard, *Novartis*
remy@usenix.org

Niels Provos, *Google*
niels@usenix.org

EXECUTIVE DIRECTOR

Ellie Young,
ellie@usenix.org

NOTICE OF ANNUAL MEETING

The USENIX Association's Annual Meeting with the membership and the Board of Directors will be held during USENIX Federated Conferences Week, June 22–25, 2010, Boston, MA. The time and place of the meeting will be announced onsite and on the conference Web site, www.usenix.org/confweek10.

RESULTS OF THE ELECTION FOR THE USENIX BOARD OF DIRECTORS, 2010–2012

The newly elected Board will take office at the end of the Board meeting on June 21, 2010.

PRESIDENT

Clem Cole, *Intel*
clem@usenix.org

VICE PRESIDENT

Margo Seltzer, *Harvard School of Engineering and Applied Sciences*
margo@usenix.org

SECRETARY

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

TREASURER

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

DIRECTORS

John Arrasjid, *VMware*
johna@usenix.org

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

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

Niels Provos, *Google*
niels@usenix.org

Not elected:

Jacob Farmer, *Cambridge Computer Services*

writing for *;login:*

Writing is not easy for most of us. The way to get your articles published in *;login:*, with the least effort on your part and on the part of the staff of *;login:*, is to submit a proposal to login@usenix.org.

PROPOSALS

;login: proposals are not like paper submission abstracts. We are not asking you to write a draft of the article as the proposal, but instead to describe the article you wish to write. Some elements are essential in any proposal:

- The topic of the article
- The type of article (e.g., case study, tutorial, editorial, minipaper)
- The intended audience (e.g., syadmins, programmers, security wonks, network admins)
- Why this article is useful
- List of any non-text elements (e.g., illustrations, code, diagrams)
- Approximate length of the article

We suggest that you try to keep your article between two and five pages, as this matches the attention span of many people.

The answer to the question about why the article needs to be read is the place for your most eloquent explanation of why this article would be important to the members of USENIX.

UNACCEPTABLE ARTICLES

;login: will not publish certain articles. These include but are not limited to:

- Previously published articles. A piece that has appeared on your own Web server but not been posted to USENET or slashdot is not considered to have been published.
- Marketing pieces of any type. We don't accept articles about products. "Marketing" does not include being enthusiastic about a new tool or software that you can download for free, and you are encouraged to write case studies of hardware or software you helped install and configure, as long as you are not affiliated with or paid by the company you are writing about.
- Personal attacks

FORMAT

Please send us plain-text proposals: simple email is fine. Send proposals to login@usenix.org.

DEADLINES

For our publishing deadlines, including the time you can expect to be asked to read proofs of your article, see the online schedule at http://www.usenix.org/publications/login/sched.html.

COPYRIGHT

You own the copyright to your work and grant USENIX permission to publish it in *;login:* and on the Web. USENIX owns the copyright on the collection that is each issue of *;login:*. You have control over who may reprint your text; financial negotiations are a private matter between you and any reprinter.

FOCUS ISSUES

Each issue may have one or more suggested focuses, tied either to events that will happen soon after *;login:* has been delivered or events that are summarized in that edition. See http://www.usenix.org/publications/login/sched.html for the proposed topics for upcoming issues.

conference reports

THANKS TO OUR SUMMARIZERS

8th USENIX Conference on File and Storage Technologies (FAST '10)

Sponsored by USENIX, the Advanced Computing Systems Association, in cooperation with ACM SIGOPS

San Jose, CA
February 23–26, 2010

OPENING REMARKS AND BEST PAPER AWARDS

FAST '10 Program Co-Chairs: Randal Burns, Johns Hopkins University; Kimberly Keeton, Hewlett-Packard Labs

Summarized by Dutch Meyers (dmeyer@cs.ubc.ca)

Conference Co-Chair Randal Burns opened the 2010 File and Storage Technologies conference by thanking his fellow chair Kimberly Keeton and the individuals and groups that made the conference a success. He also announced the chairs for FAST '11: John Wilkes of Google and Greg Ganger of Carnegie Mellon University.

Kimberly Keeton followed Randal to present the Best Paper awards for the year. Kaushik Veeraraghavan accepted one award for "quFiles: The Right File at the Right Time." The paper considers a new data abstraction that allows multiple different physical representations of data to be held in a single logical container. His coauthors include Jason Flinn and Brian Noble at the University of Michigan and Edmund B. Nightingale of Microsoft Research. The second award was accepted by Swaminathan Sundararaman, Sriram Subramanian, and Remzi H. Arpaci-Dusseau. Their paper, "Membrane: Operating System Support for Restartable File Systems" details a set of operating system changes sufficient to support a file system that can transparently restart after an error and continue servicing all requests. They and their co-authors Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, and Michael M. Swift all hail from the University of Wisconsin.

The opening ceremony also brought attention to the recent passing of Tom Clark. Mr. Clark was an innovator in SAN over IP, whose distinguished 20-year career took him to the forefront of companies such as Brocade, McDATA, and Nishan. He was an active member of SNIA, where he chaired the interoperability committee. He was also the author of three books that detail the technologies, protocols, and designs that constitute contemporary SANs and other storage virtualization systems.

Mr. Clark continues to be honored in an online memorial space at http://wtomclark.blogspot.com/ in lieu of a traditional funeral service. This site contains photographs, writing, and the memories of his many friends, colleagues, and admirers.

KEYNOTE ADDRESS

- ***Technology for Developing Regions***
  Eric Brewer, University of California, Berkeley

Summarized by Dutch Meyers (dmeyer@cs.ubc.ca)

Eric Brewer provided insights into the challenges and successes that his group, TIER (Technology and Infrastructure for Emerging Regions), faces in working around the globe and the role of research in technology that aids the health and economies of developing areas. His presentation spoke to technology's potential to change the lives of the majority of the world's population—those who live on less than two dollars per day.

Dr. Brewer began by providing an overview of development work and described how his focus relates to traditional approaches. Technological research generally caters to the more than 1 billion people in the developed world to the exclusion of the rest of the world's population, which is expected to reach 6–8 billion in the next 25 years. For most of the past half-century, development has taken a top-down view, with agencies operating at large scale to improve macroeconomic indicators. Leveraging technology in such an approach is very difficult. Existing agencies lack experience with disseminating technological advances and the commercial distribution channels used in the developed world are a poor fit for the fragmented, rural, and impoverished markets that characterize the developing world. However, Dr. Brewer sees potential in fostering development in rural areas through technological advances that grow from the bottom up.

One of his group's goals is to improve rural connectivity. To do this, they have developed a WiFi-based system called WiLDNet. Although WiFi ranges are typically shorter, the cost is an order of magnitude lower, it is incrementally deployable, and it operates in unlicensed frequencies. In deploying this system, they developed a new IP stack that better utilizes the spectrum and set a new WiFi distance record at 382 kilometers. In another project, 25,000 patients have recovered their sight though small vision centers connected to doctors through a rural telemedicine system. These clinics in Tamil Nadu, India, are staffed by a nurse who can facilitate a computer-mediated consultation between a local patient and a doctor in a remote location. This project is expanding to 50 centers that, when complete, will service 2.5 million people.

Storage concerns were highlighted in the presentation, including lack of high-quality electrical service, use of flash and optical media to transmit data, and the urgent need to deploy more storage to safeguard history and culture. For example, radio stations in Guinea-Bissau and Madagascar broadcast in local languages but are unable to record their broadcasts. Meanwhile, most of the 6000 languages in Africa are dying and few recordings exist. A storage system to address these needs may have interesting characteristics. It would synchronize infrequently with a remote system, perhaps going weeks without connectivity. Thus, the focus must be on locally self-consistent versioning that can intermittently upload deltas to bulk data repositories.

Many members of the audience had questions for Dr. Brewer. When Margo Seltzer of Harvard asked how he got started, Dr. Brewer acknowledged that it was difficult. Three years after they began, none of their three initial projects was still in operation. However, their experiences can help other researchers enter the field more easily. Good early choices included starting in a country like India that is easy to work in and partnering with established NGOs with proven records.

In response to questions from Kimberly Keeton of HP Labs and others, Dr. Brewer described the Ph.D. path for his students. Considerable field work and human subjects approval from Berkeley are required. Theses focus on technical issues and contain "one or two chapters that any technologist could call their own." While each student may produce fewer publications, Dr. Brewer argues that they have more impact. In the last two years, seven doctoral students have followed this technology-oriented track, along with three to four social science students. It was also mentioned that while the initial funding for the project came from an NSF ITR grant, that grant is no longer available, which necessitates a larger number of smaller area-specific grants.

In closing, Dr. Brewer stressed that more emphasis should be placed on technology's ability to influence development efforts and that rural areas are currently the best place to focus. Decentralized development does work, but challenges such as connectivity and power need to continue to be addressed, and storage has a significant role to play.

BUILD A BETTER FILE SYSTEM AND THE WORLD WILL BEAT A PATH TO YOUR DOOR.

Summarized by Mike Kasick (mkasick@andrew.cmu.edu)

- ***quFiles: The Right File at the Right Time***
  Kaushik Veeraraghavan and Jason Flinn, University of Michigan; Edmund B. Nightingale, Microsoft Research, Redmond; Brian Noble, University of Michigan

  Awarded Best Paper!

Kaushik Veeraraghavan described quFiles, a file storage abstraction that enables users to access different views of data in different contexts. He presented the running example of a user accessing a video file from a desktop, laptop, TiVo, or smartphone: access from each of these has different demands on resolution, decoding complexity, and required network bandwidth and latency. quFiles serves as a unifying abstraction that multiplexes different views of a single file for each of these contexts.

At its core, quFiles consists of a context-aware mechanism that selects the best representation (view) of a file for a given context. The selection mechanism is encoded in four policies provided by a type-specific quFile creation utility.

Given a particular accessing device or context, the name policy serves zero or more file names for a given quFile, which also serves to specify the file type. The content policy provides a context-specific view (file content) for a given file name. The edit policy specifies whether a particular view is allowed or disallowed to be edited, or if an edit should automatically be versioned. Finally, the cache policy determines which views of a file should be cached on devices. These policies enable views of a quFile to be generated statically at quFile-creation time, or dynamically on access.

Kaushik then described four case studies on the use of quFiles in power management, copy-on-write versioning, resource-aware directory listings, and application-aware adaptation. Each of these studies illustrates the ease with which quFile policies may be implemented—each with less than 100 lines of code, written in a week or two. He then evaluated the cost of quFiles by showing they have application-level overheads of 1% and 6% in the warm and cold cache cases, respectively.

During the question period, Garth Gibson (Carnegie Mellon) asked if Kaushik thought about providing users with a means of locating and parsing all the policies supported by a quFile system, in order to understand the data manipulation being performed. Kaushik suggested that either a debugging mode or an additional view may be added to show, for a particular quFile, what data has been statically generated and what data can be generated dynamically.

- ***Tracking Back References in a Write-Anywhere File System***
  Peter Macko and Margo Seltzer, Harvard University; Keith A. Smith, NetApp, Inc.

Peter Macko presented a method for implementing data-block to inode back references that is generally applicable to write-anywhere file systems. Consolidating free space, data migration, partition resizing, and file defragmentation all require a costly-to-generate mapping of data blocks to the inodes referring to them. By encoding such a mapping within the file system itself, back references eliminate the need to generate such a mapping.

Macko described a set of challenges faced when implementing back references: the need to ensure a low, stable overhead of operations and to support deduplication (block sharing), to enable both inode versioning (read-only snapshots) and copy-on-write block duplicates (writable clones). He then described his log-structured back reference approach, using data block allocation and deallocation records stored in multiple B+ trees to adequately address each of the implementational challenges. He stated that this approach has been implemented in the btrfs (replacing the native back reference implementation) and ext3 file systems. Finally, he evaluated the time and space overheads of his back reference implementation, concluding that the space overhead is stable and the time overhead is less than 2%.

The audience expressed enthusiasm for the approach. Chris Small (NetApp) commented that the real advantage of the scheme is that instead of reading a huge data volume into memory, potentially taking hours, back references can be read into memory incrementally as needed. Rick Spillane (Stony Brook) asked if back references are implemented using a separate mechanism from file systems for snapshotting. Macko claimed that the mechanism is separate for the sake of a generalizable implementation, but that using existing snapshotting mechanisms would be a good optimization.

- ***End-to-end Data Integrity for File Systems: A ZFS Case Study***
  Yupu Zhang, Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau, University of Wisconsin—Madison

Zhang claimed that, according to the end-to-end argument, applications should be in charge of data integrity. However, since applications must share data and therefore must agree on a way to verify data integrity, such support is most feasibly placed in the file system. While current file systems attempt to preserve data integrity in the face of imperfect storage components (media, controllers, etc.) memory corruption due to hardware faults or software bugs is also a concern, particularly as memory capacity grows and file systems continue to cache a large amount of data in memory for performance.

In this case study, Zhang performed fault injection experiments on ZFS and showed that ZFS is robust against a wide range of disk corruption but fails to maintain data integrity in the presence of memory corruption. In particular, one-bit-flip memory errors are shown to have a non-negligible chance of corruption, which—depending on workload—may manifest when reading corrupt data (up to 7%), permanent writing of corrupt data (up to 4%), system crashes (up to 2%), and even operation errors due to corrupt interpretation of metadata. Zhang stated that, in most cases, the more memory consumed by the page cache, the more likely a corruption event occurs. He concluded that, as effort has gone into protecting against disk failures, it is worth the effort to protect against memory corruption as well.

When Dominic Giampaolo asked about the rate of bit flips in workloads, Zhang described their method of injecting k bit-flips (k>1) per workload and deriving the one-bit-flip probability from the k flip data. Another audience member asked how the rate of bit flip injections matches the rate of bit flip failures seen in recent memory corruption studies. Zhang explained that they did not yet determine how often corruption and crashes happen given real bit flip data, but suggested how that could be estimated. Bill Bolosky (Microsoft) asked whether the probability of memory corruption is uniform whether the system is busy or idle, and how much corruption could be improved by pushing the checksum earlier on reads. Zhang stated that more data is needed to draw a conclusion.

Finally, Roger Haskin (IBM) asked when one stops worrying about memory corruption, since that corruption could happen before any usage, in user buffers, and even in the

very code responsible (with reduced probability due to size, but significant risk due to impact) for verifying integrity. He asked, more generally, why the file system should be the responsible entity for keeping memory in check versus hardware. Yupu answered that there are instances (e.g., memory corruptions induced by software bugs) in which higher-level techniques might be more appropriate.

LOOKING FOR TROUBLE

Summarized by Lianghong Xu (lianghon@andrew.cmu.edu)

- ***Black-Box Problem Diagnosis in Parallel File Systems***
  Michael P. Kasick, Carnegie Mellon University; Jiaqi Tan, DSO National Labs, Singapore; Rajeev Gandhi and Priya Narasimhan, Carnegie Mellon University

Michael Kasick presented his results in diagnosing problems typically encountered by PVFS developers in off-the-shelf parallel file systems: for example, the "limping-but-alive" server problems where a single faulty server impacts overall system performance but can't be identified with logs. The problems people are particularly interested in are storage-related (e.g., an accidental launch of rogue processes) and network-related (e.g., packet loss). The target parallel file systems in the paper are PVFS and Lustre. The key insight about these system is that clients typically communicate with all the servers in a request, while servers are isolated. With this important feature, it is assumed that fault-free servers exhibit similar performance metrics, while faulty servers exhibit dissimilarities only in certain metrics.

The proposed diagnostic algorithm is based on three assumptions. First, hardware is homogeneous and identically configured. Second, workloads are non-pathological, that is, requests are well distributed across the servers. Large requests touch almost every server, and small requests, which may exhibit migrating load imbalances, are well balanced over a period of time. Third, the majority of servers exhibit fault-free behavior. The diagnostic algorithm contains two phases: node indictment to find the faulty node, and root-cause analysis to find the cause of the faulty behavior. A histogram-based approach is used for most metrics in node indictment. This is done in four steps: computing a probability distribution function metric for each server over a sliding window, computing a Kullback-Leibler divergence for each server pair, flagging an anomalous pair if its divergence exceeds the threshold, and flagging a server if over half of its server pairs are anomalous. The threshold is selected in a fault-free training session. For root-cause analysis, the approach is to build a predefined table of metrics and faults. For example, if storage throughput is diagnosed as a problem in the first phase, then by simply looking up the table we know the root cause is probably a disk hog.

Someone asked about the scalability of the algorithm, since it has to compute the K-L divergence for each server pair. Kasick gave a potential solution in which the nodes can be partitioned into several groups and the computation only happens within a group. This way the complexity of the algorithm can be reduce to O(N). Another concern was the assumption that workload across all the disks would be evenly distributed, which may not hold in real systems. Kasick argued that this is acceptable as long as the workload is well balanced over a period of time.

- ***A Clean-Slate Look at Disk Scrubbing***
  Alina Oprea and Ari Juels, RSA Laboratories

Arkady Kanevsky from EMC gave the presentation.

The authors propose a smart disk-scrubbing strategy called "staggering" to adaptively change the scrubbing rate following an error event and to sample across disk regions in order to discover errors faster than by sequential reading. Latent sector errors (LSEs) are discovered only when a sector is read and so have significant impact on a system without redundancy or even RAID 5. The primary approaches used to counter LSEs are intra-disk redundancy and disk scrubbing; this paper focuses on the latter. Traditional disk scrubbing uses sequential reading of disk sectors with a fixed predetermined rate. However, according to the Sigmetrics 2007 study, LSEs exhibit temporal decay, temporal locality, and spatial locality. These features enlarge the design space of smarter scrubbing strategies to account for distribution of LSEs and disk history.

According to Bairavasundaram et al. (2007), inter-arrival time distribution has very long tails; more LSEs develop shortly after a first LSE, and LSEs develop clustered on disk at block logical level. Taking advantage of these observations, the staggering strategy partitions the disk into multiple regions, each consisting of a number of segments. Disk scrubbing is done in multiple rounds of scanning. In every round the regions are accessed one by one, but for every region only one segment is touched at a time. Once an error is detected in a segment, the entire region containing the segment is considered suspect and scanned. Due to the Sigmetrics 2007 result that all errors are clustered in a 128MB region with a very high probability, the region size is set to 128MB, while the segment size is set to 1MB to amortize the overhead of positioning. Also according to Bairavasundaram et al., the LSE rate is fairly low and constant in the first two months of drive operation, after which it increases but remains fairly constant. As a result, the staggered strategy adaptively changes the scrubbing rate. Disk lifetime is divided into four periods. In the first two months the scrubbing rate is fairly low. After this the disk enters the pre-LSE period, with a higher scrubbing rate. When an error is detected, the rate is raised to the highest level. After a period during which no further errors are detected, the disk enters the post-LSE period, when the scrubbing rate is decreased but still higher than in the pre-LSE period. Note that the authors use a new metric, mean latent error time, for single drive reliability in evaluating their simulation results.

Arkady referred the audience to the authors for answers to their questions.

- ***Understanding Latent Sector Errors and How to Protect Against Them***
  Bianca Schroeder, Sotirios Damouras, and Phillipa Gill, University of Toronto

Bianca Schroeder examined the effectiveness of current protection schemes against latent sector errors (LSEs) and provided detailed insight into LSEs' characteristics. Currently there are mainly two ways to protect against LSEs: periodic scrubbing and intra-disk redundancy. In order to evaluate how effective these approaches are, the authors leveraged the data from NetApp storage systems, which provides time of detection and logical block number for each LSE in 1.5 million drives over a period of 32 months. Analysis of this real-world data revealed that among disk scrubbing schemes, localized scrubbing and accelerated scrubbing don't bring significant improvement, but staggered scrubbing performs very well. For intra-disk redundancy, the simplest approach is single parity check (SPC); however, this fails to recover 20–25% of the disks. A stronger approach is to introduce additional parity, which, unfortunately, also adds more overhead in updating parity but may be useful in certain environments. Interleaved Parity Check (IPC) requires only one parity update per data update, can tolerate up to m consecutive errors, and is claimed to perform as well as MDS due to simulation results. Nevertheless, the result in the paper shows this scheme is much weaker than MDS, which implies the importance of using real-world data to evaluate these algorithms.

Besides the high-level summaries above, Schroeder also talked about a number of interesting questions in practice. First of all, what level of protection is it appropriate to use when? Data analysis shows that more previous errors implies a higher chance and higher number of future errors; the number of errors in the first error interval increases the expected number of future errors but doesn't significantly increase the probability of future occurrence; the probability of errors since the first occurrence drops off exponentially. Taking all these pieces together allows people to do more careful adaptive design. The second concern is whether all areas of the drive are equally likely to develop errors. According to the analysis, up to 50% of errors are concentrated in the top and bottom area of a drive; the cause of this, however, remains an open question. Is scrubbing potentially harmful to drives? NetApp doesn't provide workload data, so the authors use data collected in the Google datacenter. The result shows that there is no correlation between LSEs and the number of reads or writes. However, Schroeder stated that this conclusion needs more investigation. What is the common distance between errors? The answer is that there does exist a probability concentration, which potentially implies the insufficiency of SPC. Are errors that are close in space also close in time? Yes.

Brent Callaghan from Apple asked whether the high error rate in the beginning of the drive is caused by heavy use of this region—for example, frequent metadata operations. Shroeder answered that this question needs further investigation, because she didn't see support for this from Google's data. Another person asked about the number of parity sectors in MDS that were needed to tolerate such centralized errors. Shroeder said that spreading the parity group may help but MDS doesn't really need very large regions to protect. The real issue about MDS is its expensive update, but that is fine as long as update operations are not frequent.

WORK-IN-PROGRESS REPORTS (WIPS)

Summarized by Avani Wildani (agadani@gmail.com)

- ***MTTDLs Are Meaningless: Searching for a Better Metric for Storage Reliability***
  James S. Plank and William E. Pierce, University of Tennessee

James Plank made the case that Mean Time To Data Loss (MTTDL) is a useless, malicious metric for conveying information about disk failure probabilities. He pointed out that it is difficult to create honestly, since it requires foreknowledge about when a particular device will fail and is sensitive to parameters that are difficult to get. MTTDLs are alluring because they are relatively easy to calculate from data and derive from analytical models. Instead, Plank proposed that we look for a metric that focuses on the probability that a given disk will fail in a given, near-term timeframe. He ended the talk by listing existing competitors to MTTDL, such as bit half-life (which he describes as a median measurement), DDF per 1000 RAID groups, and data loss per petabyte-year. A commentator added "bits lost per year" during the questions phase. Plank ended his talk by mentioning a simulation environment in place to evaluate different failure time estimates comparatively.

- ***Upgrades-as-a-Service in Distributed Systems***
  Tudor Dumitras and Priya Narasimhan, Carnegie Mellon University

Dumitras addressed the problem of having to take down large data stores for scheduled maintenance such as system upgrades. He pointed out that most episodes of downtime in distributed systems are a result of these scheduled events. Using Wikipedia as a sample service, he examined what prevents an in-place upgrade. It turned out that the leading reason for offline upgrades is incompatible database schemas between releases. Over time, this results in more and more downtime for larger upgrades.

The proposed solution to this problem is to isolate the production system from the upgrade. This can be expressed as trading extra resources (which most large distributed systems have) for a reduction in scheduled downtime. Doing an upgrade online requires having two copies in production, typically, to avoid inconsistencies as data keeps coming in. The new solution allows you to put a new version on a different hardware or virtualization platform entirely and not worry about its consistency until it's swapped into place

for the running system. He finished by describing case studies.

- ***Down with the VFS and His Insidious Caches!***
  Richard P. Spillane and Erez Zadok, Stony Brook University

Spillane made a strong case for taking VFS out of the kernel and putting its functionality into user space. It's simple but unsafe for clients to directly access the server cache, since they are accessing each other's shared memory. The solution Spillane proposes is to take advantage of an unused ring in the CPU to introduce a new security level between user space and kernel space that lets you trap calls and essentially gives every client their own message stack. Spillane's results show that working in user space has almost no effect on performance with this construction: his system rivals ReiserFS for speed. In the future, he intends to generalize the construct so that anyone can be a privileged library, in addition to further extending mmap. He points out that letting you drop in alternative caching would let the cache be customized for the workload and include additions such as provenance information that are less widely implemented in current caches, and he ended with a description of how one size doesn't fit all for VFS. A questioner pointed out that some CPU architectures, such as the Alpha and MIPS, only have two rings.

- ***DiskReduce: RAIDing the Cloud***
  Bin Fan, Wittawat Tantisiriroj, Lin Xiao, and Garth Gibson, Carnegie Mellon University

Even in the cloud, there is a strong need to save space as we keep generating more and more data. Yet, given the number of unknowns in cloud storage, data reliability is at the forefront of customers' minds. Unfortunately, the triplication of GFS or HDFS is not space-efficient, and traditional RAID is complex for developers. Xiao proposed a simple solution to address this complexity to bring RAID into the cloud. The core idea is that the data would be triplicated first and then gradually RAID-ed as the data became staler or resources got tight. Pushing off the encoding step has the additional benefit of letting you encode while the system is idle, hiding the computation cost. Xiao's implementation employs a metadata server to keep track of what data blocks need to be grouped together and what to repair in case of failure.

- ***Enabling Scientific Application I/O on Cloud File Systems***
  Milo Polte, Carnegie Mellon University; Esteban Molina-Estolano, University of California, Santa Cruz; John Bent, Los Alamos National Laboratory; Scott Brandt, University of California, Santa Cruz; Garth Gibson, Carnegie Mellon University; Maya Gokhale, Lawrence Livermore National Laboratory; Carlos Maltzahn, University of California, Santa Cruz; Meghan Wingate, Los Alamos National Laboratory

Polte addressed the problem of running legacy high-performance computing software on modern cloud-based supercomputers such as those at Google. He pointed out that these computers can, in some areas, vastly outperform localized supercomputers, even large ones like Jaguar which can reach 1.75 petaflops. In contrast, he estimates Google at maybe 100 petaflops four years ago. Parallel scientific applications are written assuming a standard POSIX model with MPI that cloud systems cannot directly replicate. To counter this, Polte proposes the addition of a transition layer to easily port these legacy applications to take advantage of modern cloud systems. Polte's virtual interposition layer allows the applications to pass their writes to the underlying hardware without any application modifications. The transition layer creates a directory in the cloud with one data-log per layer and permits reopens and concurrent writes. Writes are decoupled and reads are aggregated.

- ***Non-Volatile Transactional Memory***
  Joel Coburn, Adrian M. Caulfied, Laura M. Grupp, Ameen Akel, Rajesh K. Gupta, and Steven Swanson, University of California, San Diego

Coburn introduced non-volatile transactional memory (NVTM). He claimed that storage will eventually be as fast as DRAM, even though the current cost in latency of going to disk is prohibitive. As new storage-class memory devices such as phase change RAM, scalable two-transistor memory, and memristors arrive, we will be faced with technologies that are so fast that system latency is OS-dominated instead of disk-dominated. In this situation, not addressing the OS dominance is equivalent to leaving 100x performance on the table. Coburn proposed to revise the entire stack with a focus on OS latency. The application would directly access the NVTM while the OS would handle allocation, mapping, and other administrative tasks. This was compared with Stasis, which is half as fast at best.

- ***Verifying Massively Data Parallel Multi-Stage Computations***
  Osama Khan and Randal Burns, Johns Hopkins University

Khan concentrated on the assurance of the correctness of parallel computations. Since parallelizing constructs are popular over untrusted machines, there is a ready need for the security of a verified model. Khan proposes a graph-based verification model that encapsulates parallel computations without restricting the message format. His mechanism is adaptable for other platforms since it is purely probabilistic. He proceeded to demonstrate his model and describe the underlying hash tree mechanism. His system shows 99% problem detection with only 5% overhead.

- ***Hifire: A High Fidelity Trace Replayer for Large-Scale Storage Systems***
  Lei Xu and Hong Jiang, University of Nebraska; Lei Tian, Huazhong University of Science and Technology

Accurate replaying of traces is critical to researching a new system. Xu pointed out that current approaches are not scalable or portable, and they have performance issues. He maintained that a good storage benchmarking tool needs high fidelity, scalability, performance, and portability. Hifire combines different sources of inputs and issues I/Os to external devices. Hifire primarily consists of a scheduler

and I/O device layer. Currently, Hifire is implemented on a Linux server in 1500 lines of C++ as a user-space process. The preliminary results show that 96% of I/Os were issued in 10 microseconds, with 80% in just one microsecond. This is significantly better than Buttress, which issued 90% of I/Os in 50 microseconds. Xu intends to extend this work to support large-scale systems and filesystem imaging. Finally, he intends to open source Hifire for the research community.

- ***Energy Efficient Striping in the Energy Aware Virtual File System***
  Adam Manzanares and Xiao Qin, Auburn University

Manzanares talked about energy-efficient striping in virtual file systems. The VFS manages locations of files, load balances, manages disk states, and allows clients to access data in a storage system. Manzanares introduced the energy-aware virtual filesystem (EAVFS) server, which distributes data across storage nodes. Instead of following the current model of putting disks into standby mode, which can be inefficient, Manzanares proposes a tiered approach that simplifies management of files and disks. The file system tracks file accesses, and groups of storage nodes together handle writing and striping a file. Popular data ends up on the buffer disk, and the number of disks can be matched to the nearest bottleneck. The net result of these tactics was a 12.5% energy savings per storage node.

- ***The Hot Pages Associative Translation Layer for Solid State Drives***
  Luke McNeese, Guanying Wu, and Xubin He, Tennessee Technological University

Solid state drives typically need a translation layer to map data to blocks of flash to ensure sufficient wear leveling on an SSD. Unsurprisingly, the mapping unit of the flash translation layer (FTL) has a direct correlation with its performance. For a write workload, overwrites dominate. In hot pages, the uneven accesses can cause significant overhead, whereas cold pages need fewer resources. The main idea of McNeese's work is to separate hot and cold pages so that the best schemes are used for the workload in question. His system, HPAT, sends the workload to the appropriate handler. HPAT also decides where random writes go, ensuring even wear leveling.

- ***Security-Aware Partitioning for Efficient File Systems Search***
  Aleatha Parker-Wood, Christina Strong, Ethan L. Miller, and Darrell D. E. Long, University of California, Santa Cruz

Parker-Wood described her system for security-aware partitioning for fast filesystem search. Partitioning techniques that subdivide indices are a proven way to improve metadata search speeds and scalability for large file systems, permitting early triage of the file system. A partitioned metadata index can rule out irrelevant files and quickly focus on files that are more likely to match the search criteria. However, security is also a concern: in a multi-user file system, a user's search should not include files the user doesn't have permission to view.

Security-aware partitioning, unlike Smartstore or Spyglass, incorporates security from the ground up. The system is set up such that if you can see anything in a partition, you can see everything in a partition. Partitions are created and re-arranged as dictated by changes in permissions in user data. Parker-Wood also outlined a set of evaluation criteria for evaluating partitioning algorithms focusing on information-theoretic indicators instead of statistics on synthesized queries.

- ***InfoGarden: A Casual-Game Approach to Digital Archive Management***
  Carlos Maltzahn, Michael Mateas, and Jim Whitehead, University of California, Santa Cruz

Maltzahn presented InfoGarden, a video game designed to make the task of tagging personal archives entertaining, thus leading to richer metadata for the archived system. People find digital archives overwhelming; it's easy to store data, and it's easy to lose it. Since much of this data is private, it cannot be crowd-sourced using currently available technologies. Tagging metadata as a game follows in the footsteps of successful programs such as PSDoom, which enticed system administrators to kill rogue processes by presenting those processes in the context of a first-person shooter. Similarly, InfoGarden shows untagged files as weeds in a garden representing a personal archive. Crosshairs allow you to tag a weed, turning it into a plant. Points are awarded based on the number of weeds successfully converted.

- ***Fuse for Windows***
  Mark Cariddi, Tony Mason, Scott Noone, and Peter Viscarola, OSR—Open Systems Resources

Mason talked about helping developers build kernel technologies under Windows. The goal is to enable customers to build customized file systems that run in user mode. Their system, FUSE, was designed to look as much like a Microsoft-provided framework as possible. Mason claims that much of the effort was in figuring out cygwin and gcc, among other tools from the UNIX community. Currently, they've implemented a system that can handle much of their intended domain, though chown and extended attributes are not yet implemented. The performance impact is significant: up to 40% for operations such as "open" that are small and involve a significant amount of context switching. This system will be freely available for non-commercial use.

- ***Revisiting I/O Middleware for the Cloud***
  Karthik Kambatla, Naresh Rapolu, Jalaja Padma, Patrick Eugster, and Ananth Grama, Purdue University

Kambatla discussed I/O middleware for the cloud. Applications in the cloud have different consistency and availability requirements, and current scenarios use specialized systems for each of these requirements. Poor resource utilization can increase cost. Although many sophisticated storage systems have been proposed, many of them are overkill in terms of resource overhead. Kambatla compared writes over different

cloud file systems to track memory usage. The key idea is to just index the data and store it in a log-structured file system. This gives you efficient random reads, writes, and lock-free reads. One goal that they have met is to get sequential access performance better than a standard key-value store such as Bigtable, but they are still working on an efficient implementation for random accesses.

- ***Determining SLO Violations at Compile Time***
  Kristal Curtis, Peter Bodik, Michael Armbrust, Armando Fox, Michael Franklin, Michael Jordan, and David Patterson, University of California, Berkeley

Web applications are a competitive field, and the most interesting applications answer non-trivial queries which in turn imply heavy computation costs. Everyone wants the interesting applications, and they want some idea of how well they will work. Curtis introduced the idea of SLOs for Web applications to gain some understanding of level of service being provided. SLOs could examine the trade-offs between strong consistency and availability that applications make. PICL is a query language that can estimate, at compile time, the latency of a query by sampling from a historical histogram of operators. While it's easy to combine latency estimates, it's difficult to make the model portable across queries and load conditions. Currently, the system is written and works for queries that have already been seen. Curtis is working to extend the system to new queries.

- ***LazyBase: Freshness, Performance, and Scale***
  Craig A.N. Soules, Kimberly Keeton, and Charles B. Morrey III, Hewlett-Packard Laboratories

Soules discussed enterprise information management in the context of the LazyBase management system. Typically, one runs analysis, collects metadata, and stuffs it all into one giant database. This ends up with lots of updates, and ideally these updates do not update queries dramatically. If the new data was continuously in demand, the system would fall over, but luckily many applications can work with stale data. The solution Soules proposes involves batching updates to increase throughput and isolating queries from the ingest pipeline. In LazyBase, applications can specify the level of freshness that they desire. When applications send in queries, they will initially get staler data, and as they wait LazyBase will return fresher and fresher data. There are several questions remaining in this work. Soules is working on understanding the trade-off between ingest and query performance, as well as the consistency and security ramifications of the system.

- ***Gridmix3: Emulating Production IO Workload for Apache Hadoop***
  Chris Douglas and Hong Tang, Yahoo! Inc.

Douglas introduced Gridmix3, a system to emulate production I/O for Hadoop. The goal is to take conditions seen in production and replicate them in a controlled environment to predict how these conditions will affect the system when it is deployed. Basically, they want to catch bottlenecks and bugs using a synthetic workload, since it's difficult to mirror actual production servers. There are many different types of jobs that they hope to emulate in this system, and they need to accommodate many factors to reliably predict performance of the cluster. They hope to distill sample workloads into an anonymized digest and make it available.

- ***The Case for a New Sequential Prefetching Technique***
  Mingju Li, Swapnil Bhatia, and Elizabeth Varki, University of New Hampshire

Li made the case for sequential prefetching techniques. These techniques are simple but effective and commonly used in storage caches. The system has some extra costs: the system will see some extra traffic from un-needed prefetches and early eviction, and cache pollution will continue to be a concern. However, sequential prefetching has the benefit of low mean response time along with the possibility of combating the extra costs with piggybacked prefetching, which could reduce the net system traffic. She compares several techniques and comes to the conclusion that while prefetch on hit (PoH) is the best technique naively, this is workload-dependent and could fail spectacularly on workloads that, say, tend to use data once sequentially. Li proposes to take advantage of a combination of techniques based on using a minimal amount of cache space to help reduce workload dependency and increase the hit ratio. The system load is auto-detected so that there is less prefetching when the load is high. Her ongoing work involves performance evaluation and further design details.

POSTER SESSION

First set summarized by Simona Boboila (simona@ccs.neu.edu)

- ***Router Caching for Video Streaming Systems***
  Jiawu Zhong, Zhicong He, Jun Li, Xin Wang, and Jin Zhao, Fudan University

Xin Wang proposed using the capabilities of storage and routers to increase the performance of applications. Preliminary evaluation was performed for content distribution networks and peer-to-peer systems. In content distribution networks, results show reduction of bandwidth cost with the use of caching routers. For peer-to-peer systems, the authors obtained a reduction of the server load of approximately 30%.

- ***Router-supported Data Regeneration in Distributed Storage Systems***
  Jun Li, Tiegang Zeng, Lei Liu, Xin Wang, and Xiangyang Xue, Fudan University

Xin Wang observed that in a distributed network, links have to share bandwidth, which can generate too many flows in the network. To address this problem, the authors propose the use of routers which encode the flows along the same link in a single flow. Preliminary results on a network

topology with 100 routers and 197 links show that traffic was significantly reduced with supporting routers.

- ***P-Warn: Adaptive Modeling of Energy Consumption Predictor and Early-Warning in Datacenters***
  Jianzong Wang, Rice University; Changsheng Xie, Jiguang Wan, Zhuo Liu, and Peng Wang, Huazhong University of Science and Technology, Wuhan National Laboratory for Optoelectronics

Jianzong Wang noticed that energy consumption changes with configuration and usage in datacenters. The authors built P-Warn, whose goal is to provide power predictors and give early warnings about energy consumption. The prediction model is based on several parameters: CPU utilization, I/O bandwidth, and temperature. Current work shows a trade-off between the performance overhead and the accuracy of predictions.

- ***Unix-like Access Permissions in Fully Decentralized File Systems***
  Bernhard Amann and Thomas Fuhrmann, Technische Universität Munchen

Bernhard Amann proposed securing a directory tree in a file system with a hash tree, thus providing confidentiality, authenticity, and access permissions for fully decentralized untrusted storage. This approach has several advantages: it achieves fork consistency against rollback attacks, uses fast asymmetric cryptography for improved performance, and enables ACLs to be layered on top.

- ***An Adaptive Chunking Method for Personal Data Backup and Sharing***
  Woojoong Lee and Chanik Park, Pohang University of Science and Technology

Woojoong Lee proposed an adaptive chunking model to optimize file I/Os using data deduplication. With this approach, the appropriate chunking method of a file (fixed-size static chunking, SC, or content-defined chunking, CDC) is dynamically determined based on deduplication efficiency in accordance with file types and the device's capabilities for computation. The authors also propose an algorithm to minimize the switching overhead from SC to CDC.

- ***PosFFS2: A New NAND Flash Memory File System Supporting Snapshot in Embedded Linux***
  Woojoong Lee and Chanik Park, Pohang University of Science and Technology

Sejin Park pointed out the problem of recovering data that users erase by mistake in a flash-based file system for embedded Linux. The proposed solution is building a file system, PosFFS2, which supports snapshots. With this approach, the old copy of a page remains in the system, as a snapshot and can be recovered later by the user. Preliminary results show a low storage overhead, about 5%, created by stored metadata.

- ***SLIM: Network Decongestion for Storage Systems***
  Madalin Mihailescu, Gokul Soundararajan, and Cristiana Amza, University of Toronto

Madalin Mihailescu pointed out the problem of increasing I/O needs in datacenters. To address this issue the authors propose SLIM, which uses rack-level resources to reduce network-storage traffic, thus increasing performance for low-cost network storage. In particular, SLIM uses rack-level persistent write-back cache to facilitate I/O optimizations. Preliminary results show reduced network traffic of up to 80%.

- ***Protecting a File System from Itself***
  Daniel Fryer, Angela Demke Brown, and Ashvin Goel, University of Toronto

Daniel Fryer noticed that even stable file systems have bugs which can corrupt data. He proposes to address this problem by checking transactions against some invariants before committing to disk. This approach raises a few research questions: what kind of invariants can be checked quickly, how do we specify the invariants, how thoroughly can we specify file system correction? In the current prototype implementation, the authors choose reproducible bugs, identify violated invariants, and implement checking functions for the invariants.

- ***Performance Assurance of Distributed Storage by Application-driven, Advanced and Time-based Reservation***
  Yusuke Tanimura, Hidetaka Koie, Tomohiro Kudoh, Isao Kojima, and Yoshio Tanaka, National Institute of Advanced Industrial Science and Technology

Yusuke Tanimura argued that storage access is an important bottleneck of IT systems. The proposed solution is to allow application users to explicitly reserve I/O throughput in advance. The allocation is done on a first-come reservation basis. The authors presented the system architecture and two case studies of simultaneous access to a server, which may result in access conflict. The implementation uses EBOFS developed by Ceph, enhanced with space reservation functionality.

- ***Design and Implementation of a Metadata-Rich File System***
  Sasha Ames, University of California, Santa Cruz/Lawrence Livermore National Laboratory; Maya B. Gokhale, Lawrence Livermore National Laboratory; Carlos Maltzahn, University of California, Santa Cruz

Sasha Ames pointed out the problem of separating raw data stored in traditional file systems from related, application-specific metadata stored in relational databases. He argued that this separation triggers consistency and efficiency concerns and proposed a metadata-rich file system in which files, user-defined attributes, and file relationships are all first class objects. Unlike previous approaches, the authors use a graph data model composed of files and their relationships. Preliminary results show an increase in performance

of up to 20 times obtained with the current prototype, compared to relational databases, e.g., PostgreSQL.

- ***Frequency Based Chunking for Backup Streams***
  Guanlin Lu, Yu Jin, and David H.C. Du, University of Minnesota

Guanlin Lu argued that data backup is a necessity in several contexts—for example, to minimize financial and business loss. This work focuses on chunking deduplication, a method to eliminate duplication among data backups. The authors designed a chunking algorithm with the goal of identifying as much duplicate data as possible while delivering a small number of chunks. Their approach significantly reduces the overhead of data processing and the cost of metadata.

- ***Upgrades-as-a-Service in Distributed Systems***
  Tudor Dumitras and Priya Narasimhan, Carnegie Mellon University

Tudor Dumitras argued that current approaches addressing dependable, online updates in enterprise systems are prone to failures, because the upgrade is not an atomic operation. Thus, hidden dependencies among the distributed system components may break during the update. To ensure atomicity, the authors propose isolating the old version from the upgrade procedure and dedicating separate resources to the new version. Current results obtained through fault injection prove that their system is more reliable than online-upgrade approaches.

Second set summarized by Priya Sehgal (priya.sehgal@gmail.com)

- ***NSFv4 Implementations: Who Performs Better, When, and Why***
  Vasily Tarasov, Sujay Godbole, and Erez Zadok, Stony Brook University

In this study, Vasily Tarasov and his team performed an extensive end-to-end NFSv4 performance evaluation across the multi-dimensional space of client and server implementations, workloads, NFS topology, and OS parameters. They created AuDiN, an automated evaluation framework capable of distributed filesystem benchmarking across diverse workloads. Based on the configuration file, AuDiN automatically installs the required OSes, sets up local file systems and other server- and client-specific parameters, prepares NFS export and mount points, then runs the benchmarks. Vasily observed a lot of variation under different setups. Under certain workloads (e.g., Web server) NFS performance was sensitive to the client's selection, whereas in another case (file server) performance was not affected by the client selection. He also observed performance improvements across different server platforms(~2–3x).

- ***Enabling Scientific Application I/O on Cloud FileSystems***
  Milo Polte, Carnegie Mellon University; Esteban Molina-Estolano, University of California, Santa Cruz; John Bent, Los Alamos National Laboratory; Scott Brandt, University of California, Santa Cruz; Garth Gibson, Carnegie Mellon University; Maya Gokhale, Lawrence Livermore National Laboratory; Carlos Maltzahn, University of California, Santa Cruz; Meghan Wingate, Los Alamos National Laboratory

Milo Polte claimed that there are a large variety of scientific applications, such as climate simulations or astrophysics, that require a POSIX file system or MPI I/O interface. These semantics are not supported by cloud file systems (e.g., HDFS). So, the goal of their research was to allow unmodified scientific applications to run on a cloud file system. They achieved this goal by creating an interposition layer between the applications and the cloud file system that provides a POSIX interface to the applications and performs HDFS translations on the other side. The interposition layer is implemented in the parallel-log structured file system.

- ***Verifying Massively Data Parallel Multi-Stage Computations***
  Osama Khan and Randal Burns, Johns Hopkins University

Osama Khan presented a technique to ensure the correctness of massively parallel computations and data analysis that take place in remote untrusted machines. This approach uses a graph-based model, where the vertices denote sequential code blocks, while the edges denote data paths in the system. The verification mechanism is based on collecting commitments to the input and output data at each stage of the computation and then redundantly computing the results of a small subset of computations in that stage. Their technique relies on random sampling and outputs probabilistic guarantees. Their verification mechanism can be easily adapted to a variety of platforms such as Dryad and MapReduce.

- ***An Erase and Destage-Efficient Write-Buffer Management Algorithm for Flash Memory SSD***
  Jian Hu and Hong Jiang, University of Nebraska; Lei Tian, Huazhong University of Science and Technology

Jian Hu presented a flash-aware write-buffer management algorithm called PUD-aware LRU algorithm (PUD-LRU) that was based on the Predicted average Update Distance (PUD) as the key block replacement criterion on top of FTL schemes. This work was prompted by the strong temporal locality observed in a few server workloads (e.g., TPC). The main idea of PUD-LRU is to differentiate blocks and destage them judiciously based on their frequency and recency so as to avoid unnecessary erasures due to repetitive updates. They have implemented PUD-LRU in FlashSim. The preliminary results showed that PUD-LRU reduced the number of erasures and average response time over BPLRU by up to 65% and 64%, respectively.

- ***Energy Efficient Striping in the Energy Aware Virtual File System***
  Adam Manzanares and Xiao Qin, Auburn University

Adam Manzanares presented the energy-aware virtual file system (EAVFS) and energy-aware striping within it. EAVFS consists of a server node and one or more storage nodes. The server node is responsible for distributing data across the storage nodes, while the storage nodes consist of the

actual data disks and manage the placement of the data on these disks. The storage nodes consist of a "buffer disk," which contains the most popular data. These storage nodes are divided into groups; a file is contained or striped within one group of storage nodes. Adam explained that to implement energy-aware striping, the storage nodes duplicate the stripes of the popular files into the buffer disks, putting the other disks into standby mode or powering them off, thereby reducing energy.

- ***MIND: Modeling Power Consumptions for Disk Arrays***
  Zhuo Liu, Fei Wu, Changsheng Xie, and Jianzong Wang, Huazhong University of Science and Technology and Wuhan National Laboratory for Optoeletronics; Shu Yin and Xiao Qin, Auburn University

In this work, Zhuo Liu talked about how they modeled disk array power consumption under certain workloads. This technique made use of DiskSim, the disk simulator, and tried to determine the amount of power consumed by the disk array when the workload is subjected to different RAID algorithms. The energy calculator estimated the power based on the different modes (e.g., spin up/down) of the disk and transitioning costs from one mode to the other.

- ***Investigating Locality Reformations for Cluster Virtualization***
  Ferrol Aderholdt, Benjamin Eckart, and Xubin He, Tennessee Technological University; Stephen L. Scott, Oak Ridge National Laboratory

Benjamin Eckart proposed the idea of improving performance of workloads running on a cluster of virtual machines by exploiting the property of locality of data. He proposed that if we moved the virtual machines closer or onto the nodes that contained the actual data, it would help improve performance. Currently, they pin the I/O bound applications running on a VM to a dedicated core in a cluster node. This improves the locality, as it avoids cache misses, resulting in better performance.

- ***VM Aware Journaling: Improving Journaling File System Performance in Virtualization Environments***
  Ting-Chang Huang, National Chiao Tung University, Taiwan; Da-Wei Chang, National Cheng Kung University, Taiwan

Ting-Chang Huang proposed improving the performance of journaling file systems running in a virtualized environment. The main idea is that, unlike traditional journaling approaches, which write journal data to the on-storage journal area, VM-aware journaling retains the data in the memory of the virtual machine and stores the information for locating the journal data (called the journal information) in the Virtual Machine Monitor (VMM). As a consequence, journal writes to storage are eliminated. Since only the metadata for locating the journal data is maintained in the VMM memory, it does not increase memory pressure.

- ***Security Aware Partitioning for Efficient File Systems Search***
  Aleatha Parker-Wood, Christina Strong, Ethan L. Miller, and Darrell D.E. Long, University of California, Santa Cruz

Aleatha Parker-Wood explained that partitioning is a system designed for indexing a file system. They have designed a security-aware partitioning algorithm, where if someone can see a file in a partition they can access every file in that partition. Their algorithm can add or eliminate the partitions for search depending upon the user's permissions, thereby obviating the need for expensive filtering operations. To evaluate their algorithm, she proposed building a mathematical model of what a query might look like and how their algorithm would perform under different types of queries. They are looking at different dimensions for evaluation: intra-partition similarity, inter-partition similarity, and partition size.

- ***InfoGarden: A Casual-Game Approach to Digital Archive Management***
  Carlos Maltzahn, Michael Mateas, and Jim Whitehead, University of California, Santa Cruz

Carlos Maltzahn presented an interesting way to tackle the tedious task of digital archiving through gaming, which has been used to increase productivity in the past (e.g., Chao's PSDoom). The main idea of InfoGarden is that it will make document tagging a fun activity through a gaming approach, thereby converting a neglected archive (garden) into a well-maintained one. InfoGarden considers all the documents without a tag as weeds. Once a person starts tagging his documents, the weeds get converted into plants with one or more fruits. As the garden of documents gets cleared up with more plants, the score increases.

- ***RAID4S: Adding SSDs to RAID Arrays***
  Rosie Wacha and Scott A. Brandt, University of California, Santa Cruz; John Bent, Los Alamos National Laboratory; Carlos Maltzahn, University of California, Santa Cruz

Rosie Wacha proposed a technique for improving the performance of RAID 4, which is usually bottlenecked by a common parity disk. She suggested a hybrid RAID 4 that consists of normal hard drives to store data chunks, while faster SSDs would store the parity information. Since SSDs are much faster than hard drives, they could potentially overcome the single-parity disk-performance bottleneck.

- ***Fuse for Windows***
  Mark Cariddi, Tony Mason, Scott Noone, and Peter Viscarola, OSR—Open Systems Resources

Mark Cariddi presented a FUSE implementation on Windows that enables user mode filesystem development on this platform. Mark and his team implemented a user-level service that communicates with the kernel-level library to achieve this goal. The current status is that FUSE is successfully ported to Windows. They have not implemented chown and extended attributes. Some of the difficulties they

face result from semantics mismatches between Windows and Linux file systems.

- ***Revisiting I/O Middleware for the Cloud***
  Karthik Kambatla, Naresh Rapolu, Jalaja Padma, Patrick Eugster, and Ananth Grama, Purdue University

Karthik Kambatla explained the problem of how different cloud applications vary in their requirements, such as consistency, availability, or bandwidth, leading to underutilization of resources available to the existing storage systems. He targeted a few cloud applications that need key-value store and proposed to incorporate the key-value (KV) store into the file system. The KV file format consisted of data blocks sorted on the basis of keys. The file also consisted of a data index, with key and data block number pairs, denoting where the information about a certain key was stored. All updates to a file were appended in a log-structured manner.

- ***Determining SLO Violations at Compile Time***
  Kristal Curtis, Peter Bodik, Michael Armbrust, Armando Fox, Michael Franklin, Michael Jordan, and David Patterson, University of California, Berkeley

Kristal Curtis presented this research focused on answering a few questions raised by developers of interactive Web applications: (1) what is the impact of peak workload on a query's latency and will it be within the service level objectives (SLO)? (2) is the new query meeting its SLOs? In this work, Kristal's group tried to estimate the latency of a query by first breaking it into different query operators. They estimated the latency of each query operator based on its latency distribution available from past queries. Later, they combined the individual query operator's latency, giving the total query latency. Based on the total latency, developers can determine whether their queries are meeting the SLOs and take appropriate steps. Since this analysis can be done at compile time, it can save a lot of testing and performance-analysis time.

- ***LazyBase: Freshness, Performance, and Scale***
  Craig A.N. Soules, Kimberly Keeton, and Charles B. Morrey III, Hewlett-Packard Laboratories

Craig Soules said that enterprise data management applications exhibit variations in query performance and result freshness goals. Some applications, such as Web search, require interactive performance but can operate on stale data. Others might require much up-to-date data but not have any stringent performance criteria. LazyBase is a system that allows users to trade off query performance and result freshness in order to satisfy the full range of user goals. LazyBase breaks up data ingestion into a pipeline of operations to minimize ingest time, and uses models of processing and query performance to execute user queries. Craig said that this system is ready but they are still investigating things such as: (1) how do applications specify freshness requirements? (2) how does LazyBase convert freshness to query, security, privacy, etc.?

- ***Study on Performance of Energy-efficient High-speed Tiered-Storage System***
  Hirotoshi Akaike, Hitachi Ltd; Kazuhisa Fujimoto, Naoya Okada, Kenji Miura, and Hiroaki Muraoka, Tohoku University

Kazuhisa Fujimoto proposed eHiTs, an energy-efficient, high-speed tiered-storage system that minimizes performance loss. eHiTs' first tier consists of high-speed online storage, with low-powered nearline storage as the second tier. The main idea behind eHiTs is to conserve energy by minimizing online storage capacity and powering off the HDD enclosures of the nearline storage when not needed. When an HPC job is submitted to eHiTs, it copies the required files (as specified in the job script) from the nearline to online storage, and powers off the nearline disks until the results are ready. The results and the original data are copied back after the results are available and the nearline disks are put to sleep again.

KEYNOTE ADDRESS

- ***Enterprise Analytics on Demand***
  Oliver Ratzesberger, eBay, Inc.

  Summarized by Daniel Rosenthal (danielr@cs.ucsc.edu)

Oliver Ratzesberger gave Thursday's keynote address, lending unique insight into the challenges of providing analytics at the scale of a large organization. Today, eBay runs multiple highly available, high-throughput datacenters, each of which has storage capacity exceeding 10PB. Employees are allowed to run advanced SQL queries, which support sophisticated analyses such as time-series computations, as well as custom C and Java code. This analytics infrastructure is provided indiscriminately to all employees at eBay, enabling analysis of any data to which they have access permission. This is a view of what eBay's infrastructure is in its current form; however, it was not always this way. Ratzesberger gave a brief history of the evolution of analytics at eBay and how it overcame various challenges. He also noted that many other companies, particularly smaller ones, face the same challenges today that eBay had to overcome.

Analytics began at eBay with various departments requesting access to customer or user data. Marketing might have been interested in customer purchasing behavior, whereas the Web design team might have been interested in customer click-through data. Each department would then purchase its own database, housing and maintaining that database with the rest of the departmental IT infrastructure. During his presentation, Ratzesberger described these per-department databases as "data marts." Setting up a data mart took time and was not amenable to change once deployed, and each data mart was paid for by the respective department requiring analytics. Eventually, as data marts became more and more pervasive throughout eBay, they began to contain duplicate information and would need to be synchronized with one another periodically to maintain

currency and consistency. All the while, eBay was running centralized computing infrastructure as part of its normal operations.

Eventually, around 2003–2004, one marketing data mart reached a size of 15TB, and eBay decided to try and consolidate that data mart with its centralized infrastructure. Ratzesberger recounts that the marketing data mart was eventually merged into the centralized infrastructure, and they were astounded to find out that the merge resulted in only 350GB of net new data on the centralized infrastructure. eBay had previously thought there might be 30% waste in the system, but there turned out to be orders of magnitude waste. Furthermore, the marketing data mart was returning different answers to queries than the centralized infrastructure was, because the two were not synchronized. This resulted in wrong answers to some queries that the marketing department had been running. In addition to technical issues, data marts don't show up on a single budget item; rather, they are cost-distributed, and the company never realizes how much it is spending on them.

eBay eventually consolidated all of its data marts into its centralized infrastructure using virtualization, but virtualization at a much higher level than is typically associated with the word. eBay introduced virtual data marts, which can easily be created and deleted on demand, and provided analytics as a service (AaaS) into its compute and storage infrastructure. Users are provided with a dashboard that allows creation of data marts up to 100GB in size without requiring special permissions. The virtual data mart has an associated expiration date, defaulting to a few months, but it can be renewed indefinitely. With virtual data marts, queries dynamically increase or decrease their computational resource usage depending on system load and job priority, thus attaining a major benefit of virtualization. Another benefit of virtualization is that it allows for completely automated management of data marts. This reduced the number of full-time employees at eBay required for data mart management from over 40 down to 2–4.

In closing, Ratzesberger reiterated that analytics should be dynamic and allow agile prototyping, and should allow users to fail fast when trying out new ideas. Most metrics have a hype cycle, with ROI peaking when first discovered, but eventually going into sustainment; the undiscovered metrics have the highest potential ROI. Analytics as a service reduces time to market and eliminates stray physical data marts located throughout a company.

Rik Farrow from USENIX and another audience member both asked about eBay's high utilization. Farrow pointed out that running at 100% utilization causes intermittent thrashing (since live loads are bursty), while the other audience member cited a result from queuing theory that states that at high utilization, latency grows without bound. Ratzesberger replied that, while high memory utilization causes thrashing, high CPU utilization does not. Setting SLAs for gating efficiency (runtime over runtime plus queuing time) can be used to prioritize tasks and appropriately distribute latency. Another audience member asked about how damage to data is prevented, since jobs share data. Ratzesberger replied that virtual data marts are fenced off from each other, and production tables are (mostly) read-only. The same audience member then asked if most queries scan all data, since so little indexing is used. Ratzesberger responded affirmatively, but noted that indexes are not the only way to improve performance. Data is distributed using a virtual hash, and the system supports multi-dimensional partitioning of data (e.g., time and customer segment as two dimensions). Many MPP systems, such as Teradata used by eBay, support hash-based joins, which also helps avoid sequential scans.

David Chambliss from IBM asked why Ratzesberger avoided the word "cloud" during the presentation. Ratzesberger commented that cloud computing is inadequate for I/O-intensive workloads on petabytes of data, where processing must be moved to the data rather than vice versa. Another audience member asked about potential uses for SSDs. Ratzesberger replied that, since current busses are not designed to handle the 600–800MB/s sustained data rate of SSDs, replacing hard drives with SSDs simply shifts the bottleneck to the bus. Amandeep Khurana from the University of California, Santa Cruz, asked how the underlying data is stored (e.g., in relational databases). Ratzesberger replied that more than 90% of the data is in relational databases, but with hash-based tables that are very simple in structure and can contain semi-structured data (e.g., XML, JSON, and name-value pairs). Only very frequently accessed data is laid out into columns. Ratzesberger also commented that the majority of data analysis is done in SQL and that user-defined functions are also allowed. Randal Burns from Johns Hopkins University asked about using relational interfaces to scan engines or object storage as a cost-saving opportunity over relational databases. Ratzesberger replied that, while eBay does use very expensive storage in the form of 15,000 RPM drives, the storage is also very performant, and it is difficult for lower-cost hardware to achieve the same performance in an equally cost-effective manner. Vidya Sakar from Sun Microsystems asked how eBay provides data reliability. Ratzesberger replied that eBay uses hardware RAID levels as well as software replication. For example, some data blocks are stored in two different cliques of processing nodes, preserving data even in the event of an entire clique failure.

FLASH: SAVIOR OF THE UNIVERSE?

Summarized by Lianghong Xu (lianghon@andrew.cmu.edu)

- ***DFS: A File System for Virtualized Flash Storage***
  William K. Josephson and Lars A. Bongo, Princeton University; David Flynn, Fusion-io; Kai Li, Princeton University

William Josephson presented the design and implementation of a flash-based file system, the Direct File System

(DFS). The primary feature of DFS is that it uses a new abstraction level between file system and the underlying device, which provides a large virtualized address space and reduces the redundancy of block allocation in traditional systems. Traditional file systems are designed for disks. A lot of work has been done for filesystem layout to deal with expensive disk positioning time, which isn't an issue in flash. Therefore, the good random I/O of flash isn't reflected in these systems. Also, traditional file systems designed for disks use a very complex block allocator, which seems to be redundant with flash, because flash devices themselves already embed this functionality inside the flash translation layer (FTL). DFS removes this extra level of indirection and increases the transparency of the system. Previous flash file systems were initially designed for embedded applications instead of high-performance applications and are not generally suitable for use with the current generation of high-performance flash devices.

The key design idea of DFS is to switch from the traditional block storage layer to a virtualized flash storage layer that sits in the device driver and embeds features like remapping, wear leveling, and reliability, while not disturbing the user interface. One drawback of this design is that the device driver may consume much CPU and memory resources. There are four requirements for DFS: a large virtualized address space, crash recoverability, the ability to atomically update one or more blocks, and an interface to the garbage collector for deallocating a range of blocks. According to Jefferson, these are not very strong assumptions and the trend already exists in some flash systems. In DFS, each filesystem object is assigned a contiguous range of logical block addresses. Large files and small files are distinguished and are assigned different allocation chunks. This approach enjoys its simplicity but may suffer from the waste of virtual address space.

One person asked how much memory would be consumed by the device driver in DFS. David Flynn answered that this is actually highly dependent on the write extent size. Someone from Sun raised the issue of considering ZFS as an option in their research. Jefferson answered that the Fusion-io device required driver support, and it didn't support Sun at the time they started the project. A person from NetApp pointed out that coalescing two allocators into one may lose some information and asked about the possibility of making this work better by using an object disk. Jefferson said he wasn't sure about whether this could make this work better but he thought an object-interface might be the right way to go in the long term.

- ***Extending SSD Lifetimes with Disk-Based Write Caches***
  Gokul Soundararajan, University of Toronto; Vijayan Prabhakaran, Mahesh Balakrishnan, and Ted Wobber, Microsoft Research Silicon Valley

Gokul Soundararajan showed how to use a disk as write cache for a solid-state device (SSD) to extend its lifetime. SSD has desirable features such as fast reads, low power consumption, and high reliability, but it suffers from limited write times because of the notorious "write amplification" problem. As a result, the practical write-lifetime for SSD is much less than ideal. In their approach to extending SSD lifetimes, the authors chose to add a level of indirection—adding a disk as the write cache for SSD. While it may seem odd at first glance, this decision was made out after comparison with other alternatives and serious consideration. Disks can provide comparable sequential write speed to SSDs and much higher capacity per dollar, while other choices are not suitable, for various reasons.

The authors built a Griffin hybrid device to mitigate the large volume of writes to SSDs. The Griffin hybrid device is composed of three parts: an SSD as the final destination of all the persistent data, a disk (log-structured) to cache write blocks, and a hybrid device controller which maintains the mapping of block numbers to log offsets. The basic idea is to sequentially cache all block writes to the disk log instead of writing them directly to the SSD. Then, at an appropriate time, the blocks in the disk are migrated to the flash. For every read request, the mapping table in the hybrid device controller performs a lookup to decide whether the request should go to the disk or the SSD.

In order to enhance the performance of the basic algorithm, the authors examined various I/O traces in the real world, including the I/O workload from desktops, servers, and Linux machines. The result shows that a large fraction of overwrites appear in the workload and that overwrites happen quickly while reads occur after a long interval. Based on these observations, the authors made several trade-offs between the write savings and read penalty introduced by the use of disk. Accordingly, several schemes are proposed as potential enhancement, such as selective caching and hybrid migration trigger. The evaluation shows that disk-based write cache improves SSD lifetime by a factor of two and reduces average I/O latency by 56%.

Someone from Stony Brook University asked why not use the disk as a journal and simply write all the data to the flash to reduce the overhead of migration. Soundararajan argued that this would require understanding the mechanisms of file systems' buffer cache; their device sits in the block I/O level and doesn't distinguish between eviction writes and consistency writes. Someone from NetApp asked if it is possible to use the trace data to estimate the real lifetime of current drives. Soundararajan answered that it is hard to evaluate because some of the data is collected from hard drive traces. A person from Northeastern University raised the concern about the memory overhead required to do the migration. Soundararajan said their approach only migrates 16 or 32MB of data at a time and thus doesn't incur much memory consumption.

- ***Write Endurance in Flash Drives: Measurements and Analysis***
  Simona Boboila and Peter Desnoyers, Northeastern University

Simona Boboila presented her study on write endurance of USB flash drives using reverse engineering, timing analysis, and whole-device testing. The migration of USB flash drives from mobile devices to desktop devices raised people's concern about USB flash drives' write endurance. The choice of targeting USB flash drives in this work is largely due to their simplicity, while device disassembling, destructive testing, and reverse engineering are more difficult to do for more sophisticated devices.

Device lifespan can be predicted from chip-level endurance and the internal algorithms implemented in the flash translation level (FTL). The internal algorithms can be obtained using techniques in reverse engineering. Specifically, the authors studied block update mechanisms with different complexity in three devices: a generic device, a House device, and a Memorex device. The lifespans of these devices are predicted as a function of chip-level endurance and the internal algorithms, which are pretty close to the measured result. Another technique, timing analysis, can be employed to determine whether the device is approaching its end of life. Specifically, the authors reveal that at 25,000 operations before the end, all operations slow down to 40ms. Based on the observation that the same update block is used when writes are issued to the same data block, the authors also propose a scheduling scheme to reduce garbage collection overhead and improve performance. The authors expect their results to apply to most removable devices and low-end SSDs with little free space and RAM.

Someone asked if the implementation of the scheduling mechanism should be integrated into file systems or the flash itself. Boboila said this needs further consideration but flash-based file systems are probably a good choice. Another person was curious about the extensibility of these techniques to more high-end and complex SSDs. Boboila answered that their approaches are good for low-end devices because they use simple internal algorithms and have few free blocks, but they may not be suitable for high-end devices, due to the complexity of the algorithms implemented. Another person asked if the authors had looked into the power consumption of these USB drives. Simona said no.

I/O, I/O, TO PARALLEL I/O WE GO

Summarized by Daniel Rosenthal (danielr@cs.ucsc.edu)

- ***Accelerating Parallel Analysis of Scientific Simulation Data via Zazen***
  Tiankai Tu, Charles A. Rendleman, Patrick J. Miller, Federico Sacerdoti, and Ron O. Dror, D.E. Shaw Research; David E. Shaw, D.E. Shaw Research and Columbia University

Tiankai Tu presented a technique for caching scientific simulation data on analysis nodes as it is generated in order to speed up the overall process of data analysis. The authors focus on analysis of molecular dynamics (MD) simulations, which occur over micro- or millisecond time scales and generate tens or hundreds of millions of discrete-time output frames. Each simulation consists of two potentially overlapping phases: simulation, which outputs generated data to a central file system, and analysis, which reads and analyzes simulation output data from the central file system. MD simulation output frames have a strong inter-frame dependence, causing data analysis to be tightly coupled with data retrieval. This coupling, in combination with the use of a central file system, creates an I/O bottleneck. The authors overcome this bottleneck by, at simulation time, proactively pushing frames from simulation nodes into both the central file system and the local caches of analysis nodes as the frames are generated. Then, at analysis time, the analysis nodes simply fetch data in parallel from their local caches. Any frames not in an analysis node's local cache can be fetched on-demand from the central file server.

The authors implemented their solution in a distributed consensus protocol called Zazen. The Zazen protocol ensures that no frame is analyzed more than once. Zazen requires each analysis node to generate a bitmap of its locally cached frames. These bitmaps are then exchanged between analysis nodes using an all-to-all reduction algorithm. In the course of exchanging these bitmaps, each node determines which frames it is responsible for analyzing. One key feature of Zazen is that it does not require any central servers for coordination. The authors showed that Zazen reduces file read time by nearly two orders of magnitude compared to reading data over NFS from central filesystem servers. Tu concluded by noting that Zazen only works for a certain class of time-dependent simulations, but the authors believe Zazen may be applicable to analysis of any data with a total ordering.

Craig Soules from HP Labs asked if these techniques could be applied to HDFS. Tu replied that this is an orthogonal design choice because HDFS has a central metadata server, whereas Zazen does not use centralized coordination.

- ***Efficient Object Storage Journaling in a Distributed Parallel File System***
  Sarp Oral, Feiyi Wang, David Dillow, Galen Shipman, and Ross Miller, National Center for Computational Sciences at Oak Ridge National Laboratory; Oleg Drokin, Lustre Center of Excellence at Oak Ridge National Laboratory and Sun Microsystems Inc.

Sarp Oral presented a software technique for improving journaling performance in a supercomputing environment by providing asynchronous commits of metadata. This technique is in production use on the Spider storage system, which provides storage to the Jaguar XT5 supercomputer (currently number one on the TOP500 list of supercomputers). Spider runs the Lustre file system. The authors' work was motivated by an observed factor of four difference in raw-block throughput and filesystem throughput on Spider, caused by seeks attributed to filesystem journaling. Lustre

nodes run ldiskfs locally, which is a journaling file system similar to ext3. The default journaling mode for ldiskfs is ordered mode, in which only metadata blocks are journaled, but all data blocks are written to their final locations on disk before any associated metadata blocks are committed to the journal. The size of the journal is limited to one-quarter of the disk capacity, and clients with outstanding RPC requests block on the server whenever the journal commits. Blocking clients with outstanding RPC requests prevents those clients from issuing further requests until their current request completes, creating a point of serialization.

The authors' solution is to synchronously write data but asynchronously journal metadata. This lowers the number of seeks that are required for journal commits and provides the possibility of coalescing metadata updates. The authors show this solution to be superior to hardware solutions and demonstrate that it allows ldiskfs to achieve 93.1% of the raw-block throughput, as opposed to 24.9% of the raw-block throughput measured with synchronous journaling.

Craig Soules from HP Labs asked if performance could be improved further by delaying journal commits even longer. Oral responded by noting that the journals reach their capacity very quickly due to the volume of data being generated, so delaying further might be of little benefit, since the journal must eventually be flushed anyway. Additionally, a longer delay requires replaying more data when recovering from a failure.

- ***Panache: A Parallel File System Cache for Global File Access***
  Marc Eshel, Roger Haskin, Dean Hildebrand, Manoj Naik, Frank Schmuck, and Renu Tewari, IBM Almaden Research

Renu Tewari presented a technique for cluster-to-cluster caching in a wide-area file system. Tewari opened by stating that often data sources (e.g., radar and satellites), datacenters, and clients are geographically disparate, resulting in unreliable network connections with high latency but reasonable bandwidth. The authors attempt to leverage these characteristics to provide global data access at local speeds through the use of caching. Their solution is Panache, which is a caching system built on top of GPFS. It assumes a storage model consisting of a "home" cluster, which holds the authoritative copy of a file system, and one or more cache clusters, which service clients (applications or users) requiring access to files stored on the home cluster. Panache caches files when they are first read in order to reduce the latency of future accesses. It also uses delayed write-back to reduce write latency and supports disconnected operation in the event of network failures (latent update conflicts are handled as described in Coda). Panache performs all data transfer operations in parallel across multiple nodes using pNFS, making full use of available inter-cluster bandwidth.

Sometimes dependencies arise between metadata operations, which can be problematic when using delayed write-back at a cache cluster. For example, a directory might be created, and a file subsequently created within that directory. To maintain consistency, the cache cluster must flush these operations back to the home cluster in the proper order. However, since Panache operates in parallel, there might be multiple queues of pending updates waiting to be flushed to the home cluster, potentially causing dependent operations to be permuted. Panache handles this by tracking all dependencies and enforcing a write-back order consistent with the dependencies.

Benny Halevy from Panasas asked whether file handles are stored locally or remotely, and how stale file handles are detected. Tewari responded that file handles are stored remotely, and so standard NFS error codes will be returned if a stale file handle is used.

MAKING MANAGEMENT MORE MANAGEABLE

Summarized by Sriram Subramanian (srirams@cs.wisc.edu)

- ***BASIL: Automated IO Load Balancing Across Storage Devices***
  Ajay Gulati, Chethan Kumar, and Irfan Ahmad, VMware, Inc.; Karan Kumar, Carnegie Mellon University

Ajay Gulati presented BASIL, a system for automated I/O load balancing across a group of storage devices. Storage is one of the most expensive components of any datacenter, and management of these resources presents a tremendous opportunity to avoid unnecessary investment in storage when careful utilization of resources could solve the problem. Live migration of VM hosts has been a popular technique to alleviate CPU and memory overloading. This technique doesn't directly help manage a similar situation in storage arrays. Storage bottlenecks have to be manually identified. This problem is very complex,involving identification of the problem-causing storage device, of the virtual disks that are causing the problem, and, most importantly, of the destination machine (but without causing the same problem in that machine).

The two key aspects of storage management are I/O load balancing and virtual-disk placement. The novel aspects of this work include using latency as the main metric for characterizing workload. Latency and throughput are both reflective of the workload being run—throughput tends to saturate with increasing load, but latency doesn't. Latency is more sensitive to load and tends to have linear properties, making modeling easier. For example, latency tends to linearly increase with metrics such as outstanding I/O, I/O size, and read-write ratio. Randomness present in the I/O provides an interesting point in the study—linearity breaks down with fully sequential workloads. These factors are then empirically combined to produce a workload model. Device modeling has its own set of challenges, as the performance of the underlying devices tends to vary widely and all the lower-level information is abstracted away from the hosts. The models developed here allow both load

balancing and initial placement through use of a normalized load metric. The evaluation of BASIL showed around 25% improvement in IOPS over random migration and 53% improvement in IOPS over random initial placement. It also showed promising results in studies that involved experts who were asked to perform the same operations as BASIL using the same information that was available to BASIL.

Kaladhar Voruganti (NetApp) asked if BASIL would work against the block-level migration policies of storage vendors. Gulati felt that this wouldn't be an issue as long as the granularity of these operations were vastly different—block-level migration tends to be much shorter, on the order of minutes, but BASIL works in the granularity of days or weeks. If the granularity is very similar, then turning off one of these would be better. Someone asked about write flushes in practice and how BASIL models would react to write flushes. Gulati mentioned that they haven't seen too many in practice. However, frequent write flushes should get reflected in the device model. Also, as was stated before, the model beaks down for write-heavy workloads. It was part of their future work to extend their model to work better with write workloads.

- ***Discovery of Application Workloads from Network File Traces***
  Neeraja J. Yadwadkar, Chiranjib Bhattacharyya, and K. Gopinath, Indian Institute of Science; Thirumale Niranjan and Sai Susarla, NetApp Advanced Technology Group

The paper described the work on identifying application workloads from NFS traces. Knowledge of the application can be useful in provenance mining, anomaly detection, enabling autonomous systems, etc. The goals of this work include (1) identifying workloads, (2) identifying transition workloads in a trace sequence, (3) identifying incomplete or partial traces, (4) identifying concurrent applications at the same client, and (5) identifying variability of operations—the same cp command with a minor variation in the options can produce a vastly different trace. This means that a straightforward comparison of traces won't suffice.

Yadwadkar went on to explain the significance of global and local alignment and how they pertain to the goals of this work. Global alignment is the process of arranging the sequences so as to maximize the similarity between the sequences. Local alignment allows sub-sequence matches, thereby making the transition study possible. The key insight here is that there exists a similarity between the computational and the biological problem of matching gene sequences, and profile HMMs (hidden Markov models) have been successfully employed to achieve good results. The model was evaluated using traces from both commonly used UNIX commands (cp, tar, grep, etc.) and larger workloads such as TPC/C and Postmark. It was also shown that just about 20% of the trace is required to make a reasonable match.

Margo Seltzer, Harvard University, asked how the server could retain more information, thus providing vital information typically not available in the trace. Irfan Ahmed (VMware) wondered about extending the work to identify the queries that make up the workload in TPC/C and also to use a similar technique to block traces. Yadwadkar said both suggestions would be looked into as part of future work.

- ***Provenance for the Cloud***
  Kiran-Kumar Muniswamy-Reddy, Peter Macko, and Margo Seltzer, Harvard School of Engineering and Applied Sciences

Provenance is the metadata that describes the operations that were performed on the data. They are typically described as a DAG that shows the dependency relationship between various entities that process that piece of data. Muniswamy-Reddy presented protocols for storing provenance for cloud-based applications. Cloud stores have gained significant traction since the introduction of Amazon S3 and Microsoft Azure. Cloud stores are used in a wide variety of domains: for backup, to share scientific data and Web application data, and also for Web pages themselves for hosted environments. The nature of cloud stores makes the process of storing provenance non-trivial, and the latency and cost constraints make the trade-offs very challenging. Provenance is important as it allows users to validate data sets, identify how data spread through the system, and improve the quality of data search. Cloud storage providers can also trade off between storage and computation by generating rarely used data on demand as opposed to storing it all the time if they had the provenance available to them. The work builds upon the Provenance Aware Storage System (PASS), which could originally handle local and network file systems.

The key properties that make provenance truly useful are (1) provenance data coupling—provenance should accurately describe the data; (2) multi-object ordering, which extends the coupling property all through the provenance chain; (3) data-independent persistence—provenance should be retained even after the data it describes has been deleted; and (4) efficient query—provenance needs to be accessed efficiently. Muniswamy-Reddy went on to describe three protocols of increasing strength and complexity that guarantee varying levels of these properties described above. These protocols ensure persistence and causal ordering, but data coupling and efficient queries are not available in all of them. These protocols are (1) stand-alone cloud store, (2) cloud store and cloud database (to allow efficient queries), and (3) cloud store with cloud database and messaging to ensure data coupling and efficient queries. The evaluation of the system compared the baseline time of Amazon S3fs with these systems using the Blast micro-benchmark, nightly backup benchmark, and provenance challenge benchmark. The most interesting result was that Protocol 3 delivered the best performance. Thus good performance and strong provenance are not always opposing functions.

Session chair Kaladhar Voruganti (NetApp) asked how a clean slate approach would re-architect the provenance system. Muniswamy-Reddy thought that a new approach would require better APIs from the cloud vendor as well as transactional support. He also felt that storing DAGs in cloud-db may not be ideal, and so a storage system that is better suited to storing graphs would be worth exploring.

CONCENTRATION: THE DEDUPLICATION GAME

Summarized by Dutch Meyer (dmeyer@cs.ubc.ca)

- ***I/O Deduplication: Utilizing Content Similarity to Improve I/O Performance***
  Ricardo Koller and Raju Rangaswami, Florida International University

Ricardo Koller introduced I/O Deduplication, a way to improve storage efficiency by using content similarity to improve I/O performance. It comprises three techniques: content-based caching, dynamic replica retrieval, and selective duplication. This approach was shown to improve performance 28–47% across a set of three workloads.

Each of their techniques was built around the observation that it is possible, and in some cases common, to have data at different sector addresses that share the same content. While this is the same observation that leads to traditional deduplication efforts, I/O Deduplication differs in that it focuses on optimizing the cache and I/O performance when accessing duplicate data.

In content-based caching, the authors observe that identical data may be included multiple times in the buffer cache. To address this, they created a secondary cache underneath the page cache that services blocks of data based on content. In dynamic replica retrieval, the authors note that when data is requested for a read, it could be served from any location that holds identical data. By predicting the location of the disk head based on previous I/O operations, the authors argue, one can select alternate addresses that minimize read I/O latency. Finally, in selective duplication, data is intentionally replicated across the disk to improve locality.

In closing, Koller detailed their impressive performance results and pointed out areas of future work. He proposed integrating I/O deduplication with the page cache and the I/O scheduler. He also proposed extending the technique to work with multiple disks and variable-sized blocks.

Mark Lillibridge from HP Labs noted that the whisker plots showing performance improvement had overlapping confidence intervals and asked if any conclusions could really be drawn about performance. Koller pointed to the average performance, which was improved in every instance, though not by more than the confidence interval. Margo Seltzer of Harvard University wondered how much memory would be required to achieve the same performance benefit by simply expanding the size of the buffer cache. Finally, Bill Bolosky of Microsoft Research stressed that since MD5 is insecure, one likely needs a more costly algorithm to ensure that users receive the correct data in the presence of an attacker. He asked how SHA-1 would change the performance of the system. Koller wasn't certain, but acknowledged that using the more costly SHA-1 algorithm would have a performance impact.

- ***HydraFS: A High-Throughput File System for the HYDRAstor Content-Addressable Storage System***
  Cristian Ungureanu, NEC Laboratories America; Benjamin Atkin, Google; Akshat Aranya, Salil Gokhale, and Stephen Rago, NEC Laboratories America; Grzegorz Całkowski, VMware; Cezary Dubnicki, 9LivesData, LLC; Aniruddha Bohra, Akamai

Cristian Ungureanu detailed NEC's file system, which addresses the need for scale-out failure-resistant storage that features global deduplication, high throughput, and ease of management. Their system, HydraFS, is built as a layer on top of HYDRAstor, the content-addressable (CA) store presented at FAST '09. Content-addressing schemes, while popular, do not expose an API that is compatible with conventional applications and file systems. HydraFS addresses this issue by providing a traditional filesystem interface that can sit atop a CA data store, while maintaining filesystem consistency and optimizing throughput.

When designing the system, Ungureanu and his team faced several challenges. In a content-addressed system, modifying a block changes both the data and its address. This necessitates updating on-disk references to the modified block, and doing so in a way that leaves the file system in a consistent state. Content addressing also implies a higher degree of request latency, because chunking, hashing, erasure coding, and compression each add delay to request processing. The presentation touched on many interesting aspects of the system, starting with the overall architecture and continuing into the details of the I/O path and the admissions policies that ensure that enough resources are available for each request.

As in other file systems, the write path begins with a buffer that accumulates write requests and is flushed on sync requests. These requests are passed to a chunker, which decides where block boundaries should be placed. Like many other systems, these boundaries are based on data content, so inserting into the middle of a file will not change chunking decisions far from that location. The file server places the file system's metadata updates in a log of modification records. The log is read by the commit server, which periodically translates log entries into the appropriate filesystem tree updates. Each time it does this, a new on-disk consistency point is created. At each new consistency point, the file server has the opportunity to clean its in-memory cache by removing entries that have since been updated. This process is carefully constructed to avoid large-scale or long-held locks, and need not consult the filesystem tree to perform its function. On the read path, prefetching and a buffer cache are used to mitigate the performance impacts of the deep filesystem metadata tree. Both prefetching and

eviction policy are designed to favor filesystem metadata, as metadata cache misses are particularly costly.

In the future, the authors plan to enhance HydraFS2 to operate as a cluster-wide distributed file system. They also plan to incorporate solid state disks into their designs.

- ***Bimodal Content Defined Chunking for Backup Streams***
  Erik Kruus and Cristian Ungureanu, NEC Laboratories America; Cezary Dubnicki, 9LivesData, LLC

Erik Kruus of NEC began his presentation by asking that the audience join him in considering new ways to improve on his simple but effective technique for content-defined chunk selection in backup streams. In his talk he posed the question, "What other approaches are out there, just waiting for brilliant minds to figure them out?"

Chunk selection refers to the process of taking a large file and breaking it up into smaller units of storage. These boundaries are chosen with the hope that chunks with the same data can be found in other files and duplicate data can be eliminated. To improve the deduplication rate, chunk selection is often done by evaluating a sliding window over the data. This produces variably sized chunks, where the boundaries are determined by the data itself. This content-defined chunking process increases the chances that identical sub-sequences will be deduplicated, even if they appear at different offsets or in a different file.

Selecting large chunk sizes offers lower storage overheads and better I/O performance, but smaller chunk sizes offer higher deduplication rates. Kruus's observation is that between two backups many small changes are likely to occur. Around the areas that have been modified, small chunk sizes can best capture the differences. The rest of the data is unchanged, so is better represented with large chunk size. To facilitate this approach an interface was added that allows one to check for the existence of a particular chunk in the data store. When larger chunks are not already present, smaller chunks may be considered. Kruus showed the effects of incorporating compression and the merging and splitting of chunks in an attempt to reach the highest levels of deduplication possible. He also warned that as features are introduced they increase metadata overheads, which degrades the overall performance of the system. He cited Occam's razor in explaining that even though he had an impulse to try complicated algorithms, simpler approaches usually prevailed.

In the question period, Kaushik Veeraraghavan from the University of Michigan suggested taking quick initial action to speed requests along the I/O path, then revisiting simple chunking decisions later with more complicated algorithms. Randal Burns from Johns Hopkins University drew a parallel between this work and the differencing seen in network literature, and wondered why, in the former, some workloads seemed to perform poorly. Kruus replied that without a reasonably high duplication rate (cutting the original data down to a third or more) the approach didn't have enough opportunities to provide benefit.

THE POWER BUTTON

Summarized by Sriram Subramanian (srirams@cs.wisc.edu)

- ***Evaluating Performance and Energy in File System Server Workloads***
  Priya Sehgal, Vasily Tarasov, and Erez Zadok, Stony Brook University

Power consumption is one of the biggest operating expenses of a datacenter. For every $1 spent on hardware, 50 cents are spent on just power. This clearly shows the importance of power management, and the first step towards better understanding this is to benchmark file systems' power consumption and performance. To make such benchmarking systematic is the most important contribution of this paper. Two of the commonly used techniques to reduce power are (1) right sizing—reducing the number of active components results in significant power reductions (CPU DVFS is a popular technique); (b) work reduction—in essence, reducing the workload but doing the same amount of work.

The experimental methodology involved varying the workload, file system, and hardware. The workloads employed were Web server, database, file server, and mail server (all through FileBench). These workloads have different I/O sizes, directory depth, average total number of files, read-write ratio, etc. The file systems studied (ext2, ext3, XFS, and ReiserFS) also had varied allocation block sizes, journaling modes, extents, indexing structures, etc. Finally, two widely different hardware configurations were also studied and their performance-power consumption analyzed. One significant observation was that performance closely followed energy consumption in almost all the configurations. In the first machine configuration and under the mail server workload, ReiserFS performed the best; ext2 suffered from fsync bottleneck, and XFS from a lookup-related bottleneck. Another interesting observation was that ReiserFS with no tail-packing produces a 29% improvement in performance due to avoidance of tree rebalancing.

Sudhanva Gurumurthy, University of Virginia, asked what were the other parts of the system stack that can be reconfigured. Priya suggested DVFS as a probable option. Also the system admin can provide useful hints that can make this process better. Why is fsync not a bottleneck in machine 2? Machine 2 has a faster disk and a much larger disk cache. Jason Flinn, University of Michigan, asked whether performance is a good indicator of power consumption; can performance be used as a metric to reduce energy costs? Priya felt that the current experiments were carried out at peak loads, so reduced loads might give different results. Irfan Ahmed, VMware, asked if this research work can be used to create a comprehensive power model. Priya felt that it would require more in-depth study and would be part of their future work.

- ***SRCMap: Energy Proportional Storage Using Dynamic Consolidation***
  Akshat Verma, IBM Research, India; Ricardo Koller, Luis Useche, and Raju Rangaswami, Florida International University

Luis Useche presented SRCMap, which attempts to make storage energy proportional through the use of dynamic consolidation in order to deal with datacenters' growing energy requirements, which are increasing at 15–20% every year. Storage forms a significant portion of the energy requiremen,t at almost 10% to 25%. The problem with storage devices is that even at low load, they consume peak power. Other parts such as the CPU can be powered down, making them energy-proportional. Thus the solution proposed in this work is to consolidate workloads in a few storage devices to make them energy-proportional. The most important challenge here is to keep the cost of migrating logical volumes to a minimum while still achieving the benefits of consolidation.

The three key observations that drive their solutions are: (1) active data sets only form a small portion of the total storage, (2) there is significant variability in the I/O load, and (3) over 99% of the working set consists of either popular or recently accessed data. This implies that there is significant opportunity to offload volumes due to load variability, and the stability of working sets means that the operation of synchronizing replicas of a logical volume is relatively rare.

For each logical volume, SRCMap evaluates the working sets and replicates them on scratch space available in all physical volumes. Once this is done, the short-term workload is characterized and SRCMap figures out which subset of physical volumes and replicas needs to be active for the next few hours. The other disks are spun down, thus saving power. The characterization and replication are done only at initialization, while the consolidation phase is repeated every few hours so as to change the active replicas in response to variations in the observed workload.

SRCMap also provides fine granularity for replication of volumes, a relatively low space overhead in terms of the replicas, workload adaptation, and reliability. The replica placement policy is based on the stability of the working set, the average load, power efficiency of the primary physical volume, and the working set size. Experiments were run on top of eight independent volumes and the time interval for disk spin up/down was set to two hours. The evaluation clearly showed a reduction of power requirement of about 35.5%. Also, SRCMap was shown to be an energy-proportional system.

Someone asked how writes are handled reliably during volume replication. Useche said that the reliable replication is the responsibility of the virtualization manager, and writes to both the content of the working set being replicated and to ones not in the working set were handled while the volume was offloaded. Sankaran Sivathanu (Georgia Institute of Technology) asked about the differences between this work and PARAID. Useche suggested that PARAID was a solution for disk level, and SRCMap targeted volumes that were being managed by the volume manager, thus managing more than just the physical volumes. Also, PARAID and SRCMap are complementary and can be used together. Vidhya Bhusan from Virginia Tech wanted to know about the mechanism for determining the working sets. SRCMap has block-level traces and thus can easily maintain the working set for each of the logical volumes based on the access patterns. Ajay Gulati, VMware, wanted to know if they performed any analysis on SRCMap's ability to handle bursts. Useche acknowledged that I/O bursts could pose a problem and that it was a good direction for future work.

- ***Membrane: Operating System Support for Restartable File Systems***
  Swaminathan Sundararaman, Sriram Subramanian, Abhishek Rajimwale, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, and Michael M. Swift, University of Wisconsin—Madison

  Awarded Best Paper!

Sundararaman presented Membrane, which describes operating system support to make file systems restartable. Recent research has uncovered a lot of bugs in filesystem code, and this can not only cause significant problems to end users but also prevent adoption of newer file systems. A key insight into this problem is that filesystem developers are paranoid about failures, which is evident from the presence of lots of asserts in the code. But the absence of a generic and correct mechanism of recovery results in all these asserts triggering a kernel panic. Recovery of a crashed file system is hard, as there is a lot of state spread all through the kernel and tracking; cleaning and restoring state is a challenging problem.

The three main components of Membrane are fault anticipation, fault detection, and recovery. Fault anticipation refers to the mechanisms added to prepare the kernel for a crash. In Membrane, the anticipation machinery involves generic copy-on-write-based checkpointing and maintaining operation logs to aid recovery. The fault detection components harden the kernel interfaces by adding additional parameter checks. The recovery machinery involves a cleanup of state on crash, un-mount and remount of the file system, and replay of completed operation. A key aspect of this recovery is handling operations that were in flight at the time of the crash, using the skip-trust unwind protocol. The evaluation of Membrane showed the applicability of these mechanisms to three different file systems and the limited overhead imposed on the kernel by Membrane.

Geoff Kuenning of Harvey Mudd College felt that the whole infrastructure of Membrane was hacky and wanted to know why one would trust the Membrane machinery to handle the reliability of a file system. Sundararaman replied that Membrane should be considered as the last line of defense and that filesystem developers must continue improving the

systems. Rick Spillane from Stony Brook University wanted to know how Membrane interacts with a journaling file system. Each journal transaction acts as the checkpoint boundary to ensure through modifications done at the jbd layer. The second question had to do with COW pages creating memory pressure, but Sundararaman felt that pages that are marked COW are copied only when they are touched by other processes, so it is copy on demand and not copy always. Erez Zadok from Stony Brook University asked how Membrane ensures that only filesystem code pages are made non-executable. The file system is compiled as a loadable kernel module, thus letting Membrane mark its code pages easily. Sundararaman said that some compile time modifications could help solve this issue. Jason Flinn, University of Michigan, wanted to know about the frequency of bugs that were both fail-stop and transient. The bug reporting and fixing methodology of Linux kernel is ad hoc and done primarily through mailing lists, which makes the process of identifying real bugs harder.

First USENIX Workshop on Sustainable Information Technology (SustainIT '10)

San Jose, CA
February 22, 2010

WELCOME AND OPENING STATEMENTS

Ethan L. Miller, University of California, Santa Cruz

Summarized by Priya Sehgal <psehgal@cs.sunysb.edu>

Ethan Miller welcomed everyone to the first USENIX workshop on sustainable IT. He began by explaining how sustainability is an important issue, encompassing much more than just power consumption. It is more about electronic waste, conserving natural resources, etc. There were around 45 attendees for this workshop, with one-third from outside the US. Ethan was glad to announce that people from both academia and industry had participated, and he thanked the National Science Foundation (NSF), for the student scholarships, and the industry sponsors, IBM Research and VMware.

INVITED TALK

- ***On the Science of Power Management: Encouraging Sustainability R&D***
  Erez Zadok, Stony Brook University

  Summarized by Priya Sehgal (psehgal@cs.sunysb.edu)

Erez Zadok shared the seven key findings from the science of power management (SciPM), a workshop conducted by the NSF in 2009. In this workshop, people from industry, academia, and government identified, prioritized, and recommended promising research directions in the area of power management.

The first finding was to observe systems, i.e., simply measure and analyze what systems are doing, and disseminate the results. This would aid modeling of systems and optimizing them for power and performance. Second, develop useful and clear metrics (e.g., ops/sec, ops/watt). The challenge in metrics is how to account for long-term effects such as e-waste, carbon footprints, longer hardware lifetimes, etc. Third, develop models based on the most significant factors after one observes and develops metrics. Modeling is required at all levels from hardware to software, chip, system, and datacenter.

Optimization at various levels is another important finding. There are many point solutions, but the question is how useful it is to others. Computer systems are generally complex, with multi-dimensions such as reliability, performance, energy, and security. Optimizing for multi-dimensions is challenging, as optimizing one dimension could hurt another and vice versa. For example, rotating the disks faster could improve the performance of I/O but also cost a lot in terms of energy. Thus, there is a need for rigorous analytical techniques such as control theory.

Erez Zadok pointed out that there is very little education on power management within IT. Special graduate and undergraduate courses could help solve this problem, but, for now, it could be integrated with existing system courses. He gave the example of how security education took 15 long years to get into the mainstream of courses and stressed that we should not repeat this mistake in the case of energy literacy.

Cross-disciplinary workshops and scientific community interactions will boost research in the energy and sustainability domains. We need to think beyond just computing and datacenters. There is a lot of development potential for intelligent software and hardware techniques for use in smart buildings, smart power grids, automated transportation and the like.

During the last part of the talk, Erez Zadok discussed some of the research work going on in his lab, the Filesystems and Storage Lab (FSL). Erez and his students have been trying to unravel the intricate interactions among hardware, software, and workloads. The first survey was to address the question of whether compression helps save energy. Some of the results showed an improvement in energy and performance by 10–40%, while in others it was hurt by as much as a factor of 10 to 100. Thus, the impact of compression on both performance and energy is governed by the type of workload and such characteristics as read-write ratios and file or data type. According to other research, Erez and his students found that filesystem performance and energy depend on hardware and software configurations as well as workloads. Varying filesystem configurations could positively impact power/performance from 6–8% up to a 9-fold improvement. The third research project carried out in FSL was to study the mix of NFSv4 clients and servers and their effects on performance. They performed various

benchmarks on clients and servers belonging to different platforms such as Linux, FreeBSD, and Solaris and found a performance variation of around 2–3 times.

INVITED TALK

- ***Reduced and Alternative Energy for Cloud and Telephony Applications***
  James Hughes, Huawei Technologies

  Summarized by Vasily Tarasov (tarasov@vasily.name)

James Hughes explained that in telcos, most energy per subscriber (57%) is consumed by the base stations, while the datacenters consume only 6%. Although the mobile phone's embodied CO_2 emissions are twice as high as a base station's, a base station's operation CO_2 emissions are 3.5 times higher than a cell phone's. Overall, the ratio of base station to cell phone emissions is approximately 6:5, so optimizations are possible at both sides.

Ethan Miller asked what part of a cell phone consumes the most energy. Somebody from the audience confirmed that nowadays it is the screen.

In the datacenters, 33% of the energy is consumed by the chiller, 30% by the IT equipment, and, surprisingly, 18% by the UPSes. The speaker speculated that it might be possible to eliminate UPSes somehow, but somebody from the audience said that the UPS cleans the power, so if we remove it, we need to build equipment capable of working on dirty power, which might mean higher embodied energy.

Taking into account that cooling consumes a lot of energy, it makes sense to place datacenters in colder areas. In addition, to save on energy transition and distribution (7.2% loss in the US), it is efficient to co-locate datacenters and energy generators. A potential problem in this case is increased network latency. A fibre channel link between San Francisco and Chicago (2,250 miles) has 24ms optical latency. However, the measured latency is 130ms, so there is room for improvement.

In terms of server power consumption distribution, CPU, memory, and PSU consume 30%, 20%, and 18% of energy, respectively. The average server is only 10% utilized, and taking into account that idle energy is very high, it makes sense to consolidate severs. It might require more RAM, though: 25% of RAM power consumption is static; the other 75% is mostly based on access.

Hughes considers Fast Array of Wimpy Nodes (FAWN) as a promising direction, as well as Ceph, a distributed file system. However, the scalability of these technologies needs to be practically proved. Another interesting area is recycling. The average server lives only three years. So, how can we reduce embodied CO_2 emission of a server? And can we reuse waste efficiently?

Erez Zadok asked if one can increase the usage cycle by various software techniques. Hughes answered that software definitely constitutes the largest area for improvement.

INVITED TALK

- ***The Green Cloud: How Cloud Computing Can Reduce Datacenter Power Consumption***
  Anne Holler, VMware

  Summarized by Priya Sehgal (psehgal@cs.sunysb.edu)

According to a report submitted by the EPA to Congress in 2007, energy consumption by US datacenters alone amounted to around 1.5% of the total national energy consumption in 2006, and it is expected to double by 2011. Anne Holler's talk envisioned how cloud computing could be exploited to achieve energy conservation in datacenters. Towards the end, though, she presented a few counter-arguments for how cloud computing could even increase datacenter power consumption.

According to NIST's definition of cloud computing, it is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models. The five characteristics include on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service. The three service models are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), while the four deployment models include private, public, community, and hybrid cloud. Her talk was focused on IaaS, since it is often used in datacenter (DC) virtualization.

Holler presented facts about how VM consolidation saves power. According to some previous reports from the EPA, physical hosts are usually 5–15% utilized but still consume around 60–90% of their peak power. Based on a report from VMware, VM consolidation can achieve energy savings of around 80%. Resource Management (RM) plays a vital role in efficient VM consolidation such that it maintains specified QoS and high overall throughput. Some of the elements of effective DC virtualization RM are dynamic work-conserving allocation instead of static partitioning, rich resource control set (e.g., reservations, shares, limits), RM support across a cluster of nodes with the use of live migration (e.g., VMware VMotion).

The rest of the talk focused on how the cloud model could further increase the power savings in a virtualized DC environment. As the cloud model fosters VM consolidation and provides incentives to reduce operating expenses (OpEx), it looks attractive in terms of power savings. In terms of VM consolidation, cloud provides an aggregation point for

workloads that would otherwise be executed on separate DCs. Thus, more pooling of workloads can better utilize multi-core hosts that tend to be more energy-efficient than narrower ones.

In terms of incentives, Holler pointed out that administrators of a private dedicated (non-cloud) DC often are held accountable for high resource availability rather than for power consumption OpEx. So there is a low incentive to reduce power use but significant incentive to over-provision for peak usage. Similarly, users of a private DC are often entitled to waste resources based on their capital expenditure (CapEx) and are not billed for resources their VMs consume. This leads to low incentive to reduce VM resource consumption at steady and peak loads. Cloud providers want low OpEx that can be achieved by high utilization of powered-on computer resources and keeping a few low-cost computers as spares for peak demands. Cloud providers also try to achieve low CapEx through limiting the DC capacity. Thus, cloud providers are more aggressive in reducing OpEx and CapEx than a dedicated DC administrator.

Someone from the audience asked whether Holler was making some assumptions regarding the SLAs in cloud and she answered that in cloud one expects some SLA, depending upon the workload, but they can still take some risks that a dedicated DC would not.

One way cloud providers reduce OpEx is by powering off the hosts when demand is low and waking them up when demand increases (e.g., VMware DRS with DPM enabled), and powering off the less efficient hosts. Another important technique includes using demand prediction and proactive RM: knowing, for example, that there is a steep rise in demand at 8 a.m., while at 10 p.m. it is quiet. Discounting off-peak usage (e.g., running a low-priority MapReduce job at night, when the rates are low) is another viable option. Also, for multi-site providers, moving the workloads to sites providing the cheapest power (VMotion) could help curtail OpEx.

Cloud users also want low OpEx (i.e., a basis to trade CapEx for OpEx). Resource pooling removes CapEx resource entitlement, the measured service model highlights OpEx costs, while rapid elasticity avoids OpEx for peak usage until absolutely necessary. In order to reduce the OpEx, users can characterize computing resources needed for applications and set the QoS SLA appropriately. Users can reduce resource usage of application workloads by right-sizing the VM (e.g., matching the number of virtual CPUs to workload parallelism) and reduce usage when workload is light/idle (e.g., use tick-less OS). Someone from the audience pointed out that tools that could help automatically reduce the OpEx would be very useful, and Holler agreed. There are some challenges in designing these tools—for example, when should the guests go to low power state?

Erez Zadok asked how security in the cloud would impact power consumption. Holler replied that the usefulness of the cloud might decrease in terms of power reduction if we add the dimension of security. She also said that this point could act as a counter-argument for clouds saving energy.

DON'T THROW IT AWAY

Summarized by Vasily Tarasov (tarasov@vasily.name)

- ***Estimating Environmental Costs***
  Kiara Corrigan, Amip Shah, and Chandrakant Patel, Hewlett Packard Laboratories

Kiara Corrigan presented their effort to estimate an IT infrastructure cost in the event of increased electricity prices or additional environmental taxes. Producing a computer involves a long supply chain, and every link in the chain is affected when prices and taxes go up. It is not easy to predict the new price of the final product in such a case, but the authors do so by making use of the Economic Input-Output (EIO) model in conjunction with the Life-Cycle Assessment (LCA) method. The main object in the EIO model is an inter-industry transaction matrix that exhibits the flows of the goods between different industrial sectors.

Using the EIO-LCA model, the authors consider three scenarios: (1) constant electricity prices, no carbon tax; (2) 3% increase in price (industrial rate), $10 carbon tax; (3) 3% increase in price (residential rate), $50 carbon tax.

Estimating the costs of an enterprise IT portfolio—consisting of thousands of laptops, desktops, servers, etc.—their model shows the potential increase in TCO varying from 1% to 12% depending on the scenario.

Someone asked if the model takes disposal costs into account. Corrigan answered that waste is one of the sectors, so the model is capable of accounting for disposal costs. How was the model validated? It was not; at the moment, its purpose is to compare prices between different portfolios but not to project precise costs.

- ***The Green Switch: Designing for Sustainability in Mobile Computing***
  Galit Zadok and Riikka Puustinen

Galit Zadok and Riikka Puustinen presented their methodology for designing energy-efficient mobile devices. Specifically, they concentrated on cell phones during the use phase.

There are 4.6 billion cell phone users in the world at the moment, and the projected number for 2013 is over 6 billion. More smartphones are also appearing: by 2015 all the handsets sold will be smart. An average user replaces a cell phone every 18 months. These parallel trends indicate that the impact of the cell phone industry on the environment is rapidly growing and can go out of control if no appropriate actions are undertaken.

There are certain efforts to reduce the environmental impact of cell phones at the manufacturing and disposal phases. However, less is done at the customer use phase, which, for the iPhone 3GS, for example, accounts for 49% of its greenhouse gas emissions. The Green Switch methodology suggested by the authors allows evaluating at the design stage whether a product or a service fulfills both human and ecological needs. The methodology can be distilled to a checklist that contains human and green appeals. Human appeals evaluate if the product designed is beneficial for the user, convenient to use, has good value, and is socially acceptable. The only green appeal at the moment is the reduction in energy use. As you can see, there are more human-centric appeals than eco-centric ones. This is because sustainable design solutions can have a significant positive impact on the environment only if these solutions are widespread and thus adopted by the mass market.

As an example of the Green Switch methodology, the authors presented the Green Mode concept for mobile phones. The idea is to separate the applications that are frequently used by a specific user from the rarely used ones. When a cell phone runs in the Green Mode, only frequently used services are available. This saves energy by not running background processes of non-used services. When the user wants to have access to all cell phone features, she can switch the phone to the so-called Fat Mode.

Someone asked if the authors have looked at the possibilities of reusing components from old phones in the new phones. The answer was no. The reason why this practice is not widespread nowadays is because manufacturers use proprietary designs and prefer to keep this information secret.

More information about the Green Switch methodology can be found at www.thegreenswitch.org.

- ***Towards Integrated Datacenter Energy Management: An IBM Research Strategic Initiative***
  Jody Glider, IBM Almaden Research Center

  Summarized by Priya Sehgal (psehgal@cs.sunysb.edu)

Jody Glider walked us through some of the initiatives taken by IBM to improve datacenter efficiency. IBM announced Project Big Green in May 2007, which was intended to create a family of energy-saving solutions spanning the entire stack, from hardware to software to facilities based on the large set of monitoring and management tools. In 2009, IBM launched the Data Center Energy Management strategic initiative (DCEM SI), whose goal was to identify and exploit synergies across 15 research laboratories worldwide, a kind of Project Big Green in microcosm.

The mission of DCEM SI is to deliver energy monitoring and management products to the market by working with IBM product and service partners, to develop and showcase these solutions in living datacenters, and to quantify the resulting financial/energy savings. Some of the DCEM SI projects include energy-proportional storage, energy-aware dynamic server consolidation, monitoring and visualization, novel approaches to cooling, and processor microarchitecture/memory design, etc. Out of the many DCEM SI projects, Glider presented the Mobile Measurement Technology project. He also talked about research in storage energy management.

The IBM Mobile Measurement Technology (MMT) project consists of three steps to help save power in a datacenter: (1) capture high resolution temperature, air flow, infrastructure, and layout data; (2) model the datacenter and identify improvement opportunities through optimized algorithms; and (3) manage best practices (realize air transport energy savings, thermodynamic energy savings, etc.) for reduced energy consumption. This project achieved energy savings of around 177kW with the measurements included, and the ROI was around 1–2 months. Since the datacenter can change over time, MMT has evolved from a static to a dynamic solution, which makes use of a base model and deployment of real-time sensors in the datacenter.

Before talking about research in storage energy management, Glider briefly discussed the facts about storage energy consumption. According to a report from StorageIO, storage accounts for up to 37–40% of the energy consumption of all IT components. Energy usage in the storage domain is expected to get worse as storage unit sales outpace server unit sales in the next five years. With the slowing of the per-drive increase in storage capacity and with increasing storage subsystem performance demands, there will be a need for more physical drives. Some of the strategic directions for storage energy efficiency are monitoring, modeling, control, and optimization. Better monitoring will drive control and optimization. One can optimize for better performance vs. more energy savings.

Glider talked briefly about the storage power modeling and estimation work going on in Haifa; people who are interested can contact Kalman Meth (meth@il.ibm.com). In this work Kalman Meth and his team tried to model and estimate the power consumed by a storage subsystem as a function of a particular workload and storage configuration. The total power of the disk is broken down into two main components: fixed power and dynamic power. Fixed power is the power taken up by the rotation of the spindle and the electronics on the disk. Dynamic power is the power required to perform all the I/O, i.e., to perform head seeks and data transfers. The mechanical components draw electricity from aa 12V electrical channel, while electrical components use a 5V electrical channel. Glider showed an interesting graph which showed power consumed by the 5V (electronic) and the 12V, when running various levels of 4K random read workloads on an enterprise disk.

Although the 5V part dominated the overall power consumption and remained constant for all kinds of workloads, the dynamic part was affected by the disk activity and is important in enterprise systems. The interesting thing to note in the case of dynamic power was that initially, as the number of I/O requests increased per second, the dynamic

power increased too. But after a certain number of I/O requests per second, dynamic power remained constant. This is because, with a larger number of concurrent I/O requests, the disk controller can effectively reorder the requests to shorten seek times. Power estimation consisted of constructing "Power Tables" formed by a set of pairs denoting seek activity <#seek,power> and data transfer <MBPS, power>. These power tables are later used in linear interpolation to estimate the disk power under a specific workload. This work was validated against the SPC-1 benchmark and it had an estimation error of 2.5%.

Glider presented a table of various energy optimization techniques in the storage subsystem—consolidation, tiering, opportunistic spin down, MAID, adaptive seek speeds, deduplication, etc.—and the potential energy savings achieved from each. Ethan Miller asked whether one could get extra energy savings by combining two or more of these optimization techniques or would it hurt. Glider answered that it depends on the type of the system or workload. For example, a critical airline application would take the least number of such optimizations in order to reduce the response time. On the other hand, an archiving application could make use of one or more such optimizations. Among the combination of techniques, consolidation and tiering can go hand-in-hand, while spin-down can be used where response time is not important.

In terms of tiering, Glider presented various combinations of HDD and SSD and their repercussions on cost, energy, and performance. He briefly explained the "tiering via flash cache" project. The flash cache is a block layer sandwiched between the file system and RAID layers. It caches all disk accesses onto SSD, allowing spin-down of a RAID rank that has been idle for *N* seconds. They obtained good results for a few of the MSR traces; file server trace saved 7% more energy than SAS, while firewall/Web proxy trace consumed 11% less energy than SAS. "Dynamic tiering and consolidation extent migration" (DTAC), another variant of the optimization technique, performs extent-based dynamic placement into tiers of storage after matching the performance requirement with the most appropriate tier. DTAC then consolidates the data and turns off the drives not needed. The results of DTAC looked promising in terms of both performance improvement and energy reduction compared to pure SAS configurations. (Further information on DTAC can obtained from hpucha@us.ibm.com.)

Future work includes pushing storage systems towards more variable energy cost components than fixed ones. Demand response planning is an important direction to focus on. Coordinated energy consumption optimization, i.e., ensuring that optimization in one area does not defeat optimizations elsewhere, and unified modeling and analysis to obtain optimal performance/energy are other interesting research areas that IBM is concentrating on.

Someone from the audience asked if one could replace RAM (volatile memory) with SSD and see its effect on performance and energy. Glider replied that it is not so easy and straightforward to speculate on performance and energy behaviors after replacing RAM with SSD, as the two pieces of hardware have different characteristics and cannot be used interchangeably.

PURE ENERGY

Summarized by Priya Sehgal (psehgal@cs.sunysb.edu)

- ***Power-aware Proactive Storage-tiering Management for High-speed Tiered-storage Systems***
  Kazuhisa Fujimoto, Research Institute of Electrical Communication, Tohoku University; Hirotoshi Akaike, Systems Development Laboratory, Hitachi Ltd.; Naoya Okada, Kenji Miura, and Hiroaki Muraoka, Research Institute of Electrical Communication, Tohoku University

Kazuhisa Fujimoto proposed an energy-efficient and high-speed tiered-storage system that minimizes performance loss, called eHiTs. eHiTs consists of a tiered storage system with high-speed online storage as the first tier and low-powered nearline storage as the second tier. The main idea behind eHiTs is to conserve energy by minimizing online storage capacity and powering off the HDD enclosures of the nearline storage when not needed.

eHiTs was evaluated against an HPC application. The HPC system consists of a supercomputer with an HPC management server. In eHiTs, the online storage tiered with low-powered, high-capacity nearline storage and network-attached storage offers file access to the supercomputer. All the files are always stored on the nearline storage on creation. Jobs executed in the HPC are controlled by the scheduler inside the HPC management server. Each job's script specifies the list of input files and the directory location of the result files. Based on job submission and execution time, eHiTs copies the user volume or the data needed by the job from the nearline to online storage at an appropriate predicted time. After the copy operation, the nearline HDD enclosure is powered off. After the job completes execution, the results are copied to nearline storage. Also, the former volume that was copied to online is copied back and remounted to the user directory. This nearline storage can then be powered off again.

The results obtained from their testbed with 64TB capacity showed that the system was able to conserve up to 16% of energy consumed by an ordinary tiered-storage system with the same capacity.

- ***Towards Energy Proportional Cloud for Data Processing Frameworks***
  Hyeong S. Kim, Dong In Shin, Young Jin Yu, Hyeonsang Eom, and Heon Y. Yeom, Seoul National University

Hyeong Kim investigated the feasibility of using power-save mode (PSM) in cloud computing. He basically tried to answer two questions for data processing frameworks: (1) the feasibility of low-power computers instead of commod-

ity servers, and (2) the practical challenges in enabling PSM. To answer these questions, they evaluated Apache Hadoop on four different classes of machines. They also proposed AnSwer (Augmentation and Substitution), an energy-saving method to reduce energy consumption by augmenting the power-hungry servers with low-powered ones in correct proportion.

The machines evaluated by Kim included two server class machines, called Svr1 and Svr2, and two low-powered machines using Atom CPUs, called Low1 and Low2. He reported the results of running two Apache Hadoop workloads, mainly sort and gridmix, in units of normalized running time and performance/watt. Although Low1 and Low2 exhibited a performance degradation by a factor of 2–3 compared to Svr1 and Svr2, they turned out to be more power-efficient by a factor of 14 to 113. This motivated him to propose AnSwer, which makes use of these low-powered machines.

According to Kim, suspending partial servers leads to inevitable problems such as data loss and performance degradation. Using AnSwer, a rack of low-powered servers can be used for reliability. Whenever a high-powered server is suspended, its replica can be placed at this low-powered remote rack. This ensures reliability at reduced cost. Thus AnSwer uses two techniques, augmentation and substitution. Augmentation reduces the data transfer caused when auxiliary nodes replace existing servers, while substitution reduces the impact on data processing by replacing high-end servers with low-powered ones. Some of the practical challenges which he did not address completely are part of future work: which nodes to choose for partial suspension, and where to migrate lost replicas.

INVITED TALK

- ***Storage Class Memory: A Low-power Storage Opportunity***
  Richard Freitas, IBM Almaden Research Center

  Summarized by Vasily Tarasov (tarasov@vasily.name)

Richard Freitas told us about the current status in Storage Class Memory (SCM) technology. SCM is a new class of storage/memory devices that blurs the difference between memory and storage. It is non-volatile, has short access time, is cheap, and does not have moving parts. There are several dozen technologies competing to be the "best" SCM. Flash memory is a widespread example, but Richard believes that in the future the majority of SCM will be represented by the Phase Change Memory—PCM.

Areal density of conventional disk drives keeps growing by approximately 40% a year. However, access time is proportional to the linear density, approximately the square root of areal density. Consequently, disk access time reduces by only 15% a year. Due to physical and cost limitations it is hardly possible that we will have disk drives rotating faster than 15,000 RPM. Consequently, the gap between CPU performance and disk access time continues to widen. Additionally, space and power become larger concerns. To satisfy 2 GIOP/sec one needs 5 million HDDs, which occupy 16,500 sq. ft. and consume 22 megawatts of energy!

SCM is a possible solution for the problems above, and flash memory is the most widespread SCM technology nowadays. It is based on the classical MOS transistor with a redesigned transistor gate. This allows placing or removing charge from the transistor, which corresponds to 0 and 1 values of a bit. Flash provides much lower access times, but has well-known endurance problems. Additionally, the smaller the flash cell, the higher the areal density, which is desirable. But already in certain flash-based products you can almost "name" every electron residing near the gate.

PCM is similar to the technology used nowadays in DVD disks. As it turns out, if you put a piece of DVD-like material between two electrodes, heat it, and then cool it down, depending on how you do it the resistance of the material will be different. Using the difference in resistance one can encode and decode zeros and ones or even multiple values in one PCM cell (which is similar to Flash MLC). The potential problem for PCM is the high current required to heat the cells. Freitas' crystal ball suggests that as early as 2016, the prices on PCM might be comparable to the prices on flash memory. Once SCM memory is available there will be a lot of ways to redesign current computer architecture appropriately.

If it's so fast, should we put it near to CPU, that is, in front of the I/O controller, or should we keep it behind the I/O controller? The presenter thinks both will be appropriate. There might be classes of SCM: very fast and expensive ones that should fit near the DRAM modules, and slower and cheaper SCM that can go behind the I/O controller, next to the disk drives.

IMPROVING THE (DATACENTER) ENVIRONMENT

Summarized by Priya Sehgal (psehgal@cs.sunysb.edu)

- ***Effects of Datacenter Vibration on Compute System Performance***
  Julian Turner, Q Associates

Turner presented an interesting paper in which he tried to determine whether ambient datacenter vibrations affect storage system I/O and performance. This was basically prompted by a YouTube video called the "yell test" performed by Brendan Gregg of Sun Microsystems, in which he demonstrated the adverse impact on I/O when he yelled at a running storage system. Another goal of Turner's presentation was to determine if any performance degradation found could be reduced or eliminated using specially designed anti-vibration rack by Green Platform Corporation (GPC).

In order to evaluate this behavior, Turner ran a number of benchmarks, mainly micro and macro benchmarks from FileBench, on a Sun 7110 array with sixteen 300GB, 10K

RPM SAS disks against two environments. The first environment was a specially constructed sound room with ambient noise less than 40dB and with no source of vibration within two miles. The second environment was a Tier 1 raised floor datacenter. The second environment was characteristic of an enterprise datacenter, with vibrations from compute nodes, A/C equipment, UPS, etc. Also, in the second setup, he ran experiments on two types of racks: a metal CPI rack and a GPC anti-vibration rack (AVR). The Sun Analytics tool was used to capture and report different characteristics of the disks, an important one being disk I/O broken down by latency.

Turner observed that the vibrations did impact random read and writes significantly. Performance numbers on the metal rack in environment 2 were worse than those on AVR and the ideal environment, 1. Performance improvements for random reads ranged from 56% to 246%, while for random writes it ranged from 34% to 88%. Streaming sequential reads and writes experienced a much smaller improvement than its random counterparts. Another important observation made by Turner was about the "latent performance effect," i.e., when a system was moved from, say, a metal rack to an AVR or vice versa the performance numbers remained roughly equivalent to the previous system for some time. Not taking this latency effect into account would lead to significant errors in the benchmarks. In his view, the old state of the system (e.g., cache hits) causes these latent performance effects.

Finally, Turner raised some research questions: (1) how much vibration can be allowed in a datacenter without any performance degradation? (2) how can these vibrations be mitigated? (3) should a datacenter really care about it? Yelling measures 130dB+ and is definitely detrimental, whereas a datacenter operates at 80dB to 95dB, which could still affect performance adversely.

Someone from the audience asked whether they measured the amplitude of the vibrations. Turner replied that they could measure the frequency but not the amplitude.

- ***CFD-Based Operational Thermal Efficiency Improvement of a Production Datacenter***
  Umesh Singh, Amarendra K Singh, Parvez S, and Anand Sivasubramaniam, Tata Consultancy Services Ltd., India

Umesh Singh talked about the application of a computational fluid dynamics model (CFD) in a production datacenter for improving its operational efficiency. The model he described was based on conjugate heat transfer and fluid flow at a datacenter. The authors carried out CFD analysis for a mid-sized (~3000 sq. ft.) datacenter. The model was used for providing design and operational guidelines to improve the energy efficiency of the servers and also to help in ramping up the partially filled datacenter. The guidelines relate to layout and airflow modifications required for an efficient cooling system while avoiding hot-spot formation.

As the first part of the modeling procedure, information regarding the geometry layout of the equipment, relevant thermal loads, and operational conditions was collected. Using the geometrical details, a 3D layout of the datacenter was created. This model was meshed with the hexahedral elements, consisting of hangers, beams, etc. The mesh files were loaded into CFX, software to model air-flow and heat transfer, with necessary boundary conditions incorporated. The computational solution for this problem setup was generated through CFX-Solver and results for different parametric studies were reported.

This model was validated with temperature measurements. Some of the recommendations obtained from this model which helped improving the energy efficiency included optimum placement of tiles in cold aisles and increased supply temperatures from CRAC units. These recommendations resulted in 20% energy savings in a datacenter that was ramping up.

PANEL SESSION

- ***The Present and Future of Sustainability R&D***
  Moderators: Ethan L. Miller, University of California, Santa Cruz; Erez Zadok, Stony Brook University

  Panelists: Kirk Cameron, Virginia Tech; Douglas H. Fisher, National Science Foundation; Dushyanth Narayanan, Microsoft; Amip Shah, HP Labs; Matt E. Tolentino, Intel

  Summarized by Vasily Tarasov (tarasov@vasily.name)

Dushyanth Narayanan (Microsoft) voiced a provocative opinion about the pointlessness of sustainability in IT. If one averages the amount of energy used per person in the world, 40% goes to cars, 30% to jet flights, and only 5% to electronic appliances, which include not only computers, but vacuum cleaners, microwave ovens, etc.! So it is unlikely that there are big sustainability targets in IT. But it makes sense to use IT for optimizing energy use in other areas. Somebody disagreed: optimizations need to be done in every area to make people think "green," and IT has one of the widest proliferations.

Amip Shah from HP Labs pointed out in his speech that sustainability becomes a large-scale problem. Though a lot of studies are concentrated on devices, "the device is not the whole system." If you look at any modern device, there are multiple industrial levels below that allowed creating it, and inter-industry interconnections are very large and complex. Due to these connections, it is possible that improvements in one sector that consumes only 2% of energy (IT) will affect all other industry sectors drastically. Shah concluded that we need to use large system models (some of them already exist, e.g., in economics) to see the whole picture.

Kirk Cameron from Virginia Tech emphasized that different applications exercise different systems differently, e.g., use systems' components unevenly. Consequently, power optimization must be specific to the use scenarios. He showed

as an example his tool, MicroMiser, wich allows saving energy on Linux/Windows machines.

Matt Tolentino talked about what Intel does to decrease their footprint on the environment. First, modern CPUs consume 10^-7 less power than decades ago, and they are 10^5 times faster. Intel introduces eco-friendly package materials, reduces the carbon footprint of their factories, cleans wafers using recycled water. Interestingly, Intel is the largest green power consumer. The main question for Intel now is how to bring Intel's green technologies to other industries: how to effectively use it in Smartgrid, build better turbines, etc.

Douglas Fisher from NSF presented a lot of programs and funding opportunities that are for sustainable IT research.

Ethan Miller asked about the possibility of replacing computer components as a measure for increasing IT infrastructure life cycle. In fact, nowadays, people just throw computers away, instead of upgrading them. Amip Shah agreed that this is a big problem.

Somebody asked about solar energy perspectives. Dushan answered that for solar energy to be widespread, a distribution network is needed. Additionally, there is no silver bullet: some devices won't work on solar energy.

Douglas mentioned the use of reduced functionality as a way to save energy. For example, thin clients might be a good way to go.

Tolentino noticed that economic factors are more important than sustainability. Dushyanth answered that revenue optimizations sometimes include environmental factors, especially when the society becomes more environmentally responsible.

2nd USENIX Workshop on the Theory and Practice of Provenance (TaPP '10)

San Jose, CA
February 22, 2010

INVITED TALK

- ***Naming, Identity, and Provenance***
  Jim Waldo, Distinguished Engineer, Sun Microsystems Laboratories

  Summarized by Aditya Parameswaran (adityagp@cs.stanford.edu)

Jim Waldo's talk centered on identity as a philosophical notion and how it relates to how we think about provenance and data.

Waldo discussed what "identity" means. Identity is hard to describe (how do we say that a is the "same" as b?), understand (if a is the same as b, what does that mean?), and teach. In addition, identity may be discovered through references. For example, two items that were called by different names might be discovered to be the "same."

Our notion of what is the "same" is also confusing. To illustrate this, Waldo described the paradox of the Ship of Theseus: There is an original Ship of Theseus, whose parts are stripped off and used to build a new identical ship. At the same time, the old ship is embellished with new parts to replace the old parts. At this point, it is hard to say which of these two ships is the "same" as the original Ship of Theseus.

Even in computer science, there is ambiguity in "identity." For instance, there are two notions of identity in programming languages: referential identity (== in Java) and structural identity (.equals() in Java). The name of a variable is equivalent to the reference, while structural identity actually compares the content.

Waldo spoke of the connections between the problems of reasoning about identities and modal logic. He suggested that reasoning about various "versions" of an object can be done by means of possible worlds, with the notion of a "Designator," i.e., a canonical version of an object, whose properties may change over various possible worlds.

In science, there is a need for reproducibility in experiments, in order to get the "same" result. However, reproduction is hard, since in the worst case, one might need a snapshot of the entire universe to reproduce the same environment for the experiment. Thus the challenge is to maintain the "right stuff" in order to be able to reproduce experiments to a rough approximation. Traditionally, we do this in computer science by maintaining data via version control. However, it is still unclear what we need to save (as provenance) to ensure reproducibility.

Waldo then cautioned us that identity is a deep unsolved philosophical problem which has been around for centuries, and thus it is likely that it will not be solved in the near future. However, he suggested that for special cases, understanding and solving the problem of identity should be possible.

SECURITY AND EXPERIENCE

- ***Trusted Computing and Provenance: Better Together (long paper)***
  John Lyle and Andrew Martin, Oxford University Computing Laboratory
- ***Towards a Secure and Efficient System for End-to-End Provenance (short paper)***
  Patrick McDaniel, Kevin Butler, and Stephen McLaughlin, Pennsylvania State University; Radu Sion and Erez Zadok, Stony Brook University; Marianne Winslett, University of Illinois at Urbana-Champaign
- ***Towards Query Interoperability: PASSing PLUS (long paper)***
  Uri J. Braun and Margo I. Seltzer, Harvard School of Engineering and Applied Sciences; Adriane Chapman, Barbara Blaustein, M. David Allen, and Len Seligman, The MITRE Corporation

- ***Provenance Artifact Identification in the Atmospheric Composition Processing System (ACPS) (short paper)***
  Curt Tilmes, NASA Goddard Space Flight Center and University of Maryland, Baltimore County; Yelena Yesha and Milton Halem, University of Maryland, Baltimore County

No reports are available for this session.

INVITED TALK

- ***Provenance for the Nationwide Health Information Network***
  Latanya Sweeney, Distinguished Career Professor of Computer Science, Technology, and Policy, CMU, Director of the CMU Privacy Laboratory, and Visiting Scholar at the Harvard Center for Research on Computation and Society

Summarized by Robin Smogor (pyrodon@gmail.com)

Recently, national funding is pushing the creation of a nationwide health information network. Currently hospitals and care facilities are not sharing information even locally, due to privacy concerns, except for billing or claims which are forwarded to national databases in the various insurance companies. Building a system to provide the attributes desired by policymakers and health care providers involves tracking many different kinds of provenance, and solutions that naively use one kind will often cause issues in other kinds. The provenance community needs to rise to the design challenge soon in order for a solid, good network to become adopted as hospitals move towards sharing information nationwide.

Someone pointed out that when digging into complex provenance a little, we sometimes want a cumulative answer and also proof but are not allowed to share proof. For example, we want to count the number of newly diagnosed cases of HIV in an area, but providers can't share identifiable characteristics that are needed for deduplication. Sweeney agreed and said that this is one of the big problems we need to solve. Even negative information can reveal information (no new cases in a hospital gives information about the site). Sweeney also mentioned that part of the 2009 stimulus bill in the US called for nationwide electronic medical records by January 2011. That deadline will likely be postponed but will eventually happen, and if we don't propose a better solution to the committee distributing the money and the various projects working towards nationwide EMR, they will default to using social security numbers as the unique patient identifiers, which would not be good. Someone else commented that we want national sharing of case details in some form, especially for rare diseases. You really want clinicians to have access to good information so they can treat things they have never seen.

Someone else asked about an architecture that focused on the patient, having each control their own flash drive. Sweeney answered that providers consider it their information, not the patient's. Providers don't want to give you full access to your own medical records because it might "confuse you." There is also the liability aspect—they don't trust the patient to protect their own data. Another person pointed out that all labs are not created equal. The local lab may use less accurate machines or methods than the regional lab. Sweeney responded that the value is in the report, not the processes or materials. That is, a PCP can't read an x-ray any better than we can. The value is in the radiologist's report and trust is in the radiologist, not in the lab tech who took it.

Someone else asked about policy, pointing out that medical records are in C42 format, which does not include provenance. Since provenance is not covered by the standard, how can we get it included? Sweeney answered that clinical information is mostly in plain text, not a database format, for anthropological reasons.

Finally, an attendee wondered about modeling the health network on credit reporting, with three approved competing businesses. Sweeney said she liked the idea of health reporting agencies, but it's not getting support right now, even though there's a nice proof of concept. The government is paying from the bottom up, and medical software manufacturers are doing a big turf sweep, tying up California.

SYSTEMS AND USES OF PROVENANCE

Summarized by Peter Macko (pmacko@eecs.harvard.edu)

- ***Panda: A System for Provenance and Data (short paper)***
  Robert Ikeda and Jennifer Widom, Stanford University

Panda is a work-in-progress project developing a complete, general-purpose solution for capturing, storing, and querying provenance. The project focuses on workflow-based systems and captures both provenance and data, which enables it to support a rich set of features. For example, a user would be able to pick one of the inputs and trace it through the computation, or select a piece of the output and trace it backwards. The system would also be able to propagate a change in the input by recomputing only the parts of the workflow affected by the change. Similarly, the system would be able to check whether a given result is still valid after correcting an input and then use this forward propagation method to refresh its value.

One of the goals of Panda is to seamlessly support relational operators with known, well-defined semantics as well as fully opaque operators. The system would further support query-driven provenance collection (record only the provenance that you need to answer pre-specified queries), lazy provenance computation and storage (compute provenance of selected parts of a workflow only when needed), multiple granularities of provenance, and approximate provenance (allow the system to record provenance imprecisely in order to save space).

A member of the audience asked the speaker how the provenance is captured—whether the system places wrappers around the workflow operators or executes them inside

a provenance-aware interpreter. The speaker explained that they do not use a specialized Python interpreter; the individual workflow operators export their own provenance back to the system.

- ***Towards Practical Incremental Recomputation for Scientists: An Implementation for the Python Language (long paper)***
  Philip J. Guo and Dawson Engler, Stanford University

Scientific computations run typically on the order of minutes to hours. This makes the development cycle unacceptably long: after a developer corrects a few lines of code at the end of the program, he or she has to rerun the entire computation. Most developers thus break their programs into small pieces, which read and write intermediate results. While this reduces the development cycle, it greatly increases the complexity of the code, is time-consuming, and introduces many new bugs.

The paper describes a system that addresses this problem by providing a modified Python interpreter that automatically memorizes saves) results of functions. The paper focuses specifically on programs written in Python, but its approach generalizes to any interpreted general-purpose imperative language. The system detects code changes both in the actual memorized function and in all functions it calls. The interpreter also keeps track of which functions read which files and on the state of the global variables that the function reads for any given memorized result. Furthermore, the system is careful not to memorize the results of impure functions, which mutate non-local values, write to files, or call non-deterministic functions.

The described approach uses only dynamic analysis, so some members of the audience were wondering about the possibility of using static analysis. The speaker explained that using dynamic analysis is conceptually more straightforward, but it is possible to use static analysis as an optimization. Furthermore, static analysis is difficult in interpreted languages with no explicit types, such as Python. Another member of the audience asked whether the system influenced the way its users develop their programs. The authors did not come far enough to provide their system to its intended real users, but ideally, the users would structure their programs using more self-contained functions.

Why did they choose to use Python for their work? One of the main reasons was the authors' personal familiarity with this language, but this technique should work with any other high-level dynamic language, such as Matlab. When does the system purge its memorization cache? They remove entries from the memorization table whenever the system detects a new version of the code. What about the space overhead and about dealing with changes in libraries? There is anecdotal evidence that the space overhead depends on the size of the intermediate data and that the changes in libraries would be detected by the interpreter's code change-detection mechanism. Finally, in the response to a related question, the speaker explained that the system does not yet handle function calls outside Python.

- ***Using Provenance to Extract Semantic File Attributes (short paper)***
  Daniel Margo and Robin Smogor, Harvard University

The authors present a method for automatically extracting meaningful semantic attributes of files from their provenance, or more precisely, the context in which they are used. For example, if an application always reads a file in its directory, it is most likely its component, but if the application sometimes writes a file outside its directory, it is probably a document.

The described tool captures provenance from PASS, collapses versions of the same objects into single nodes, and then proceeds with feature extraction. The program produces multiple ancestor and descendant graphs for each file with different features, such as with collapsed nodes with the same name or path, or with just file or process objects. The program then extracts simple per-file statistics, such as node and edge counts in the neighborhood of each file. The authors also experimented with graph clustering and other sophisticated methods of feature extraction, but they did not produce good results.

The authors next combined the extracted features with relevant meta-data of existing files collected using the stat command, and then constructed a decision tree. They evaluated their approach by predicting file extensions (because that makes it easy to establish ground truth) and achieved 86% accuracy.

Which features did the decision tree split on? It typically split on the depth of the provenance graph, because this is an indication of how often the file is accessed. A good research direction is to consider the shape of the graph. Do semantic attributes reflect what the document contains? This is still a research question; another question is whether the usage of the file reflects what the user thinks about the file. So far, the project has shown that the way a file is used predicts its type, and it is an open question how far it is possible to push this.

Is their method an alternative for content-based extractors? It is beneficial to extract as many rich semantic attributes as possible, so this method should be used in conjunction with traditional content-based extractors. Furthermore, this method still allows you to extract attributes from files that were already deleted. Finally, someone suggested that the authors should consider expanding their work to include feature extraction methods from graph indexing and querying literature.

MODELS: NEW AND DIFFERENT WAYS OF THINKING ABOUT AND REASONING ABOUT PROVENANCE

Summarized by Abhijeet Mohapatra (abhijeet@stanford.edu)

- ***A Graph Model of Data and Workflow Provenance (long paper)***
  Umut Acar, Max-Planck Institute for Software Systems; Peter Buneman and James Cheney, University of Edinburgh; Jan Van den Bussche and Natalia Kwasnikowska, Hasselt University; Stijn Vansummeren, Université Libre de Bruxelles

James Cheney presented a graphical model that captures the common formalism for workflow and database provenance. Cheney's work addresses the fact that workflow systems are seldom accompanied by formal specifications of the desired provenance semantics. Hence, it is difficult to integrate database and workflow provenance or compare provenance generated by different systems.

Cheney proposed a model based on provenance graphs that document the evaluation of a DFL program. Such graphs contain values as well as evaluations. Ignoring the value structure in the provenance graph would produce the order of evaluation of processing nodes.

Cheney described their implementation of the proposed graphical model in Haskell. He then discussed how different provenance queries could be expressed over provenance graphs. These queries were a mixture of Datalog and annotation propagation queries. Most of the queries related to where and why provenance in databases. Finally, he outlined some unsolved problems that relate to modeling updates to provenance graphs and identifying classes of provenance queries that exhibit symmetry in querying the provenance graph "forward" vs. "backward."

- ***A Conceptual Model and Predicate Language for Data Selection and Projection Based on Provenance (long paper)***
  David W. Archer and Lois M.L. Delcambre, Portland State University

David Archer presented a predicate language that supports a broad class of provenance queries having applications in data curation. Current provenance models have two major shortcomings. First, they are either fine-grained or coarse-grained. Second, annotation management in such systems is messy. Thus, there is a need to develop a language that helps end users pose queries that select data by its provenance information.

Archer described a conceptual model for capturing provenance that separated provenance tracking and its manipulation by end-users. He then proposed a predicate language to record provenance for SELECT and PROJECT operators using "path qualifiers."

Archer later evaluated the proposed model against Trio and PASS's provenance models comparing the expressivity of provenance queries.

- ***On the Use of Abstract Workflows to Capture Scientific Process Provenance (long paper)***
  Paulo Pinheiro da Silva, Leonardo Salayandia, Nicholas Del Rio, and Ann Q. Gates, University of Texas at El Paso

Paulo Pinheiro da Silva presented a model to capture and reuse how provenance in scientific processes. He was prompted by the fact that scientists often track provenance without using methods specifically designed to record provenance, and this makes it hard to reuse the recorded provenance. In his proposed model he used Process Markup Language (PML) to encode distributed provenance.

Da Silva began his talk by describing the languages and tools commonly used to capture provenance of scientific processes. He later described how provenance could be captured for automated as well as manual processes. He also outlined a data annotation scheme to support provenance queries.

At the end of the talk, da Silva noted that the proposed approach to capture provenance might not be scalable.

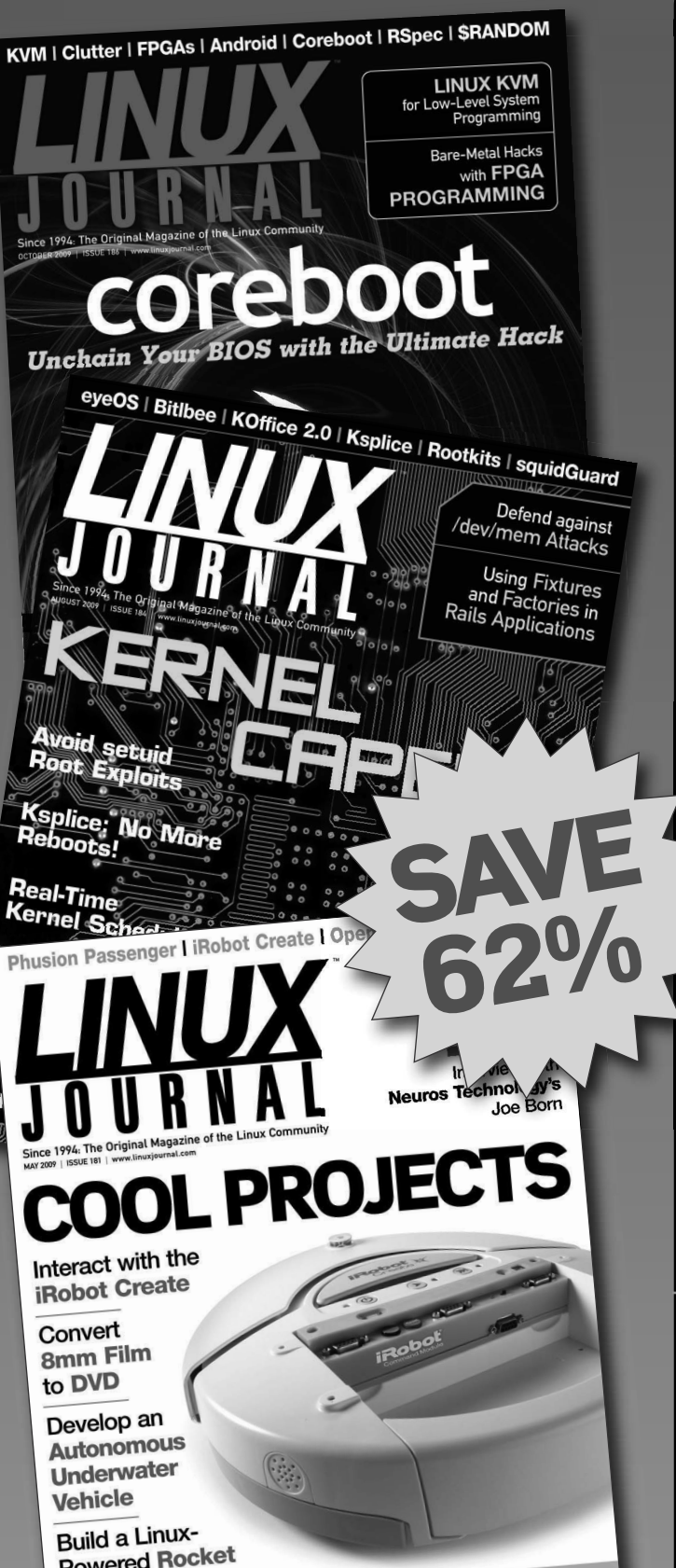
- ***Provenance-based Belief (short paper)***
  Adriane Chapman, Barbara Blaustein, and Chris Elsaesser, The MITRE Corporation

Adriane Chapman presented a mechanism to express trust in data sources without actually accessing them. This is intended to help answer provenance queries based on a certain level of trust. She commented that provenance graphs can be viewed as a causal structure which can be used to compute belief of an output from assessments of input data and derivations.

Chapman talked of integrating Bayesian causal reasoning and provenance. She described how belief of outputs could be computed by generating conditional probability tables for the output tuple's intermediate derivations. She commented that the provenance store could be used to identify sharing between sources. Modeling provenance with a causal model would enable propagation of beliefs based on shared and independent sources.

Chapman ended the talk by saying that her group is currently implementing the causal model to capture provenance into a real system for evaluation purposes.

If You Use Linux, You Should Be Reading LINUX JOURNAL™



- In-depth information providing a full 360-degree look at featured topics relating to Linux
- Tools, tips and tricks you will use today as well as relevant information for the future
- Advice and inspiration for getting the most out of your Linux system
- Instructional how-tos will save you time and money

SAVE 62%

Get *Linux Journal* delivered to your door monthly for 1 year for only $29.50! Plus, you will receive a free gift with your subscription.

SUBSCRIBE NOW AT:

WWW.LINUXJOURNAL.COM/SUBSCRIBE

Offer valid in US only. Newsstand price per issue is $5.99 USD; Canada/Mexico annual price is $39.50 USD; International annual price is $69.50. Free gift valued at $5.99. Prepaid in US funds. First issue will arrive in 4-6 weeks. Sign up for, renew, or manage your subscription on-line, www.linuxjournal.com/subscribe.

USENIX
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send Address Changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES