news@UK

The Newsletter of UKUUG, the UK's Unix and Open Systems Users Group

Published electronically at `http://www.ukuug.org/newsletter/`

Volume 17, Number 3 **ISSN 0965-9412** **September 2008**

Contents

News from the Secretariat

Jane Morrison

On 30th June UKUUG, in conjunction with O'Reilly, organised a successful tutorial on Moodle. This event was the first of many we hope to bring you in the future with UKUUG working alongside O'Reilly.

Josette Garcia has access to many authors who can also be of interest to our membership by providing tutorials etc.

OpenTech 2008 (organised by Council member Sam Smith), was an exceptional success with some 600 people attending. Two accounts of the event appear in this Newsletter.

The 2008 event was so successful that plans are going forward for a similar event in early July 2009.

The Linux event this year has been delayed to November and you should find enclosed in this Newsletter the provisional programme and booking information.

On 26th November we are organising the OpenBSD's PF tutorial by Peter N M Hansteen. This is a repeat (with updates) of the tutorial previously held earlier in the year in Birmingham. On-line booking is available via our web site.

Looking further ahead to next year enclosed you should find the Call For Papers for the Spring Conference being held in London from 24th to 26th March 2009.

Please put these event dates in your diary now.

The AGM this year will be held on at 18:15 on 25th September in the Cruciform Building at University College London. Details have recently been sent to all members, and can also be seen on the web site. As usual the AGM will be followed by a technical talk.

The next Newsletter will be the December issue and the copy date is: 21st November. Please send submissions to:
`newsletter@ukuug.org`

Book discounts

O'Reilly books can be ordered through the UKUUG office at a discount of 32.5% on the published price, or (even better) directly through `www.oreilly.com` using the UK shopping cart and the promotional code OR111, which will give a discount of 35% and free postage and packing for UKUUG members.

UKUUG is also a UK distributor of books from the GNU Press which are also available at a discount for both members and non-members. For further information see
`http://www.ukuug.org/books/gnu/`

We also offer a special price on Philip Hazel's Exim book: see
`http://www.ukuug.org/books/exim/`

Members can also obtain 25% off books from Pearson Education (including Addison Wesley and Prentice Hall) and 30% off books from Wiley: for more details contact me at the office or see
`http://www.ukuug.org/books/`

Chairman's report

Alain WIlliams

Report on legal action with the BSI (British Standards Institute)

You will recall that we sought a Judicial Review of BSI's actions in voting "yes" to the fast-tracking of the Microsoft sponsored DIS29500 (OOXML) proposed standard in April 2008.

Shortly after I wrote my last report for the newsletter the court (Mr Justice David Lloyd Jones) rejected UKUUG's application for Judicial Review and observed: "*The application does not disclose any arguable breech of the procedures of BSI or of rules of procedural fairness.*", and "*In any event the application is academic in light of the adoption of the new standard by ISO.*"

UKUUG appealed against that decision since we believed that the judge had not understood the arguments, his decision was based purely on reading the papers.

Since then there has been no progress on the legal front, but we now have a date when this will return before the judge, but this time UKUUG and BSI will be able to present their cases orally. This will be 21st October.

Most of the action has been outside of the UK at the level of ISO (International Standards Organisation). Formal objections were submitted by South Africa, Brazil, Venezuela and India. These were rejected by ISO on 15th August; note that this was done with a corrected version of DIS29500 not being available, it is still marked 'deleted' on the ISO web site.

I gather that BSI supported the further processing of all four of the appeals on the basis that they all specifically referred to either the handling of the alleged contradictions, the non-publication of the text, or both, which BSI felt merited further investigation and discussion.

UKUUG does not understand how anyone is supposed to implement a standard when the document that describes it is not available. It is clear that the big loser in all of this as been the ISO itself, its reputation is severely damaged.

We will keep you informed of new developments at:
http://www.ukuug.org/ooxml/

We remain grateful to Richard Melville who continues to represent UKUUG on the relevant BSI committee.

Other matters

Many UKUUG members act as Unix consultants either full time or occasionally. To help you we created a special page on our web site where you can be listed, give a web link and describe your services in 100 words. Few of you have done so, contact us to have your details added. If you need a consultant, please visit this page and check out fellow UKUUG members. Please visit:
http://www.ukuug.org/consultants/

This is my last report as UKUUG chairman since I have now served the maximum of 6 years as a council member. I shall be standing down at the AGM in September. I have no intention of disappearing and will still be seen at conferences and such.

LISA '08 announcement

We have received the following notification of the LISA '08 event which will take place in San Diego, California, USA in November.

On behalf of all of the LISA '08 organizers, I'd like to invite you to join us in San Diego, CA, November 9-14, 2008, for the 22nd Large Installation System Administration Conference:
http://www.usenix.org/lisa08/proga

For the past 20 years LISA has been the focal point for the global community of system and network administrators. This year LISA continues that tradition, featuring innovative tools and techniques essential for your professional and technical development.

Take advantage of the popular 6 days of training. Select from over 50 tutorials taught by highly expert instructors, including:

- Mark Burgess on Integrating Cfengine into Organizational Service Management
- Tom Christiansen on Advanced Perl
- David N Blank-Edelman on Over the Edge System Administration

Plus, new in 2008, we're offering tracks on virtualization and on Solaris. These two 6-day series include classes such as:

- Peter Baer Galvin and Marc Staveley on Solaris 10 Administration

- James Mauro on Solaris Dynamic Tracing (DTrace)
- Richard McDougall on VMware ESX Performance and Tuning

The full training program can be found at
`http://www.usenix.org/events/lisa08/training/`

In addition to the training, 3 days of technical sessions include top-notch refereed papers, informative invited talks, expert ‘Guru Is In’ sessions, and a poster sesson.
`http://www.usenix.org/events/lisa08/tech/`

Our 20+ invited talks feature our most impressive slate of speakers to date. They include:

- Keynote Address: “Implementing Intellipedia Within a ‘Need to Know’ Culture” by Sean Dennehy, Chief of Intellipedia Development, Directorate of Intelligence, U.S. Central Intelligence Agency
- Plenary Session: “Reconceptualizing Security” by Bruce Schneier, Chief Security Technology Officer, BT
- Plenary Session: “The State of Electronic Voting, 2008” by David Wagner, University of California, Berkeley

LISA is the premier forum for presenting new research in system administration. We selected papers showcasing state-of-the-art work on topics including configuration management, parallel systems deployment, virtualization, and security.

Bring your perplexing technical questions to the experts at LISA’s ‘Guru Is In’ sessions.

Explore the latest commercial innovations at the Vendor Exhibition.

Benefit from opportunities for peer interaction (a.k.a. the “Hallway Track”).

Take advantage of the live streaming opportunities.

For complete program information and to register, see
`http://www.usenix.org/lisa08/proga`

OSCON 2008

Peter H Salus

In July I flew to Portland, Oregon, to be one of the keynote speakers at O’Reilly’s Open Source event, OSCON . . . along with over 3000 other attendees. It was quite an event. (As with all events, there was too much going on simultaneously to get to most stuff.)

I did get to hear Mark Shuttleworth on Tuesday night and a truly excellent presentation on “Economics, Standards and IP” by Stephe Walli on Wednesday. Wednesday evening I spent some time with the Perl folks, chatting with Larry Wall and Tom Christensen. Thursday morning, I listened to an extremely enthusiastic Tim O’Reilly – who remarked “Data is the Intel inside” – and Keith Bergelt, the CEO of the Open Invention Network, who was extremely illuminating on the “challenges that Linux faces from a patent perspective”. (Oh, yeah, I spoke, too.)

All the usual suspects were there, of course. The motto I came away with was: **No secret software!**. Sounded good to me.

Lots of good talks, some interesting stuff at the vendor exhibits, many interesting people.

It’s really expensive for folks in the UK to get to conferences in the US. Luckily, O’Reilly has two this year (2008) in Europe: Rails in early September, which will be past by the time you read this; and the second Web 2.0 Expo Europe, taking place 21-23 October, in a new venue – the Berliner Congress Center, in central Berlin, DE.

I was very impressed by the efforts of O’Reilly’s conference staff, not least Shirley Baiiles, the Conference’s Speaker Manager, and Maureen Jennings, O’Reilly’s PR manager.

Among the awards, I was proud to see that another Canadian, Angela Byron (Drupal), received the Best Contributor award. Angela gave a superb paper at last year’s Ontario Linux Fest, in a session I was lucky to chair.

Now, if only O'Reilly were holding a conference in Canada . . .

Other cis-Atlantic (government agency) news

Much to my pleasure, PUBPAT – the Public Patent Foundation – succeeded in getting the US Patent and Trademark Office to reject the dynamic Web site patent of EpicRealm. In their filings, PUBPAT submitted prior art that the Patent Office was not aware of when reviewing the applications that led to the two patents and described in detail how the prior art invalidates the patents. The Patent Office has now found that the first patent indeed is invalid. I'm sure the second is, too.

"EpicRealm is yet another example of the growing trend of businesses whose sole purpose and activity is to sue others for patent infringement, but the fact that they are claiming rights over the vast majority of websites based on these patents that the Patent Office has now found have substantial issues relating to their validity only makes the matter that much more unsettling", said Dan Ravicher, PUBPAT's Executive Director. "Perhaps some day soon Congress will fix the patent system so that such exploitation cannot occur. In the interim, with respect to these specific patents, now that the Patent Office has agreed with us that the first of the patents is invalid, we expect them to find the same with respect to the other and then ultimately withdraw these patents from issuance."

Some of you may have heard that ComCast was blocking BitTorrent. On August 20, the US Federal Communications Commission issued a 34-page order telling ComCast to stop under threat of severe penalties. I've not read the mass of legalese yet, but there's a five-page commendatory letter by Larry Lessig which appears to be a fair presentation. And Larry always makes sense.
http://www.lessig.org/blog/

Comedy section

On 27 August, it was revealed that several laptops taken to the International Space Station were virus-infected. The virus was identified as `W32.Gammima.AG worm`. The "`W`" stands for Windows, the only OS this malware effects. Perhaps (someday) NASA et al will learn not to employ Windows.

Also on 27 August, the US FAA revealed that "a computer breakdown" had caused a delay in "hundreds" of flights. The glitch occurred in Atlanta and then at the "backup site" in Salt Lake City. Trivial research revealed that the FAA was running Windows. Here's part of a 2005 press release:

> *The FAA is implementing the Stratus servers, which use Intel Xeon 2.8 MHz large cache MP processors and support the Microsoft Windows operating system, at control centres in Atlanta and Salt Lake City. Uninterrupted availability of the NADIN 1 is important to all aspects of the aviation industry, as well as the nation's economy and, increasingly, as a tool to help protect national security.*

What more can I say?

27 August was a good day. Judge Howard Lloyd, in the US District Court in the Northern District of California, threw out adult entertainment company IO Group's 2006 copyright infringement case against Veoh. At the time, Veoh had some user-uploaded porn on its service that belonged to IO Group. Despite quick takedowns from DMCA notices, IO Group sued anyway. Judge Lloyd said, among other things, that as the total came to under 7% of Veoh's postings and that the takedowns meant that Veoh was secure under the DMCA's "safe harbor" provision. So all you porn posters can breathe easy.

And from the courts . . .

On 16 July, Justice Dale Kimball held that The SCO Group had, indeed, not passed on to Novell monies due. Over $2 million plus accrued interest. This was far less than I would have expected, but, in all likelihood it doesn't matter. The SCO Group doesn't have $2-3 million, nor is there much they can sell off to glean such an amount. More importantly, when this is officially communicated to the bankruptcy court in Delaware, the IBM suit may be released from durance vile.

At the same time, the court in Munich (DE) has held that sco.de violated a previous order and has fined them 10,000 Euros. And the arbitration proceedings in Switzerland should resume around the time you read this.

My opaque crystal ball refuses to reveal the eventual outcome, but I am fairly confident that The SCO Group will no longer exist at that point.

Oh, yes, don't worry about Novell or SUSE or IBM or AutoZone or Red Hat. I doubt whether they ever expected to gain substantial funds from this. But they certainly will have demonstrated that commercial blackmail is unproductive.

More self-advertising

By the time you read this, my *The Daemon, the Gnu and the Penguin: How Free and Open Software is Changing the World* should be available. ISBN 978-0-9790342-3-7. Try to find it through Amazon.

And, should you be in the US, I'll be speaking at the Ohio Linux Fest on 11 October.

Open Tech 2008 (1)

Paul Thompson

As a relatively new member of UKUUG, this was my first Open Tech event so I wasn't quite sure what to expect – but I wasn't disappointed. The day was packed with over 40 talks about technology, society and low-carbon living. With three concurrent tracks it was often difficult to decide on which session to forfeit, but I was determined to hear those around Green Business. Here are my recollections of a couple that I attended.

From E-Business to G-Business

Martin Chilcott compared and contrasted the growth of G-business today with the rise of E-business in the 1990's Internet boom.

He suggested that the chasm between early adopters of green technologies ("dark greens" and technology enthusiasts) and the early majority "light greens" is wider than that famously identified in Geoffrey Moore's "Crossing the Chasm". Martin argued that the lessons learnt from the adoption lifecycle of Internet technologies are applicable to the revised adoption lifecycle of G-business processes, particularly a recognition of costs and efficiencies.

Various regulations requiring lower carbon use are forcing businesses to adopt "disruptive" green technologies, but unlike with Internet adoption, the major players are corporations capable of delivering whole solutions – and the innovators are having to co-operate with big business. Several examples were presented of where adoption is necessitating complex partnering between very different organisations (e.g. Cisco and Arup) and how traditional competitors are being coerced into collaboration, e.g. M&S and Tesco – "they'll work together to address these technological challenges, then get back to competing over baked beans".

The presentation was very informative, and provoked some interesting questions from the audience. A recurring thread was the suggestion that G-business projects require longer-term and more up-front funding than Internet-revolution projects. Martin responded that venture capitalists have recently become used to that model (vide Biotech projects).

Solar Hot Water and Open Source IT

Tim Small gave a whirlwind tour of the current state of solar hot water technologies and the financial and environmental benefits of deployment.

The general technical message was that heating water via electricity generated by photovoltaic cells is inefficient and expensive compared with heating water directly using evacuated tube collectors – despite the latter requiring additional pump and controller.

The presentation was mostly about domestic systems (of which Tim has built several from scratch), but he indicated how even bigger savings could be gained at larger installations. Examples cited included a housing association expecting to recoup the cost of replacing oil-fired boilers within three years (although 7 years is more likely for a typical family home) – however he also mentioned the controversial worst-case estimate by RICS of 208 years to recoup outlay. Tim cautioned that commercial organisations employing "double-glazing-style" sales tactics were charging householders several thousands of pounds for installations that could be done by a self-installer for a few hundred pounds, but the photographs of his airing cupboard (with more CPUs than a typical airing cupboard) might not be the best advert! Commercial companies might also be less open to investing in emerging [more efficient] technologies of which there are many (albeit buried in academic research journals) because it's expensive to re-train and their established methods "kinda work".

It was suggested that applying Open Source methodologies to the pump controller hardware and software could benefit them greatly. The controller's purpose is to ensure the pump circulates water (or heat-transfer fluid) between the collectors and the storage cylinder only when the temperature differential warrants. Some simple Open Source hardware and software designs are already available (Freeduino et al) but there's scope for improvements in functionality and sophistication. This part of the session was probably the most interesting to the audience, and raised the most discussion points, including one observation that by eliminating the controller entirely and simply powering the pump from electricity generated by PV cells, it would operate only when there's sufficient sunlight – which mostly coincides with when it's required.

Open Tech 2008 (2)

Paul Waring

On a wet and windy morning in July, I rose at the crack of dawn to get down to London for Open Tech 2008 – a one day event held at the students' union of University College London, sponsored by BT Osmosoft and organised by UKUUG and friends. On the starting line, and viewed through slightly bleary eyes, were a series of talks covering a diverse range of subjects, from Rembrandt to Cybernetics. Simon Wardley's successful attempt to deliver 150 slides in 15 minutes, punctuated by various humorous references, was particularly impressive and certainly tied in well with the topic of "Living with Chaos".

The second session was my turn to chair, with presentations on TiddlyWiki (a wiki which doesn't require a server, so documents can be shared on USB pens or sent by email), the Android mobile device platform and Social networks and FOAF (introduced as "making the Semantic Web actually work" – a challenge if ever there was one!). All three talks provided interesting insights to the projects under discussion, and the short time limit on each meant that no one strayed too far off topic. The familiar beast of "laptop not wanting to talk to projector" reared its head part-way through, but we still managed to leave plenty of time for a variety of questions and finish on time – particularly important as this was the slot before lunch.

The first afternoon session began with feedback on "one year on from the Power of Information Report", which revealed some fascinating uses of data which has been made available by the public sector. The announcement of a new data unlocking services from the Office of Public Sector Information to allow people to request that information held by public sector bodies is made available for re-use will hopefully spark some new and interesting applications – and indeed some projects are already being worked on as a result.

One thing in particular which I noticed about the day was the amount of networking which occurred over the course of the event. Whilst the most useful conversations at conferences are often those had over tea in the breaks – or a pint in the bar afterwards – OpenTech seemed to stimulate this more than I've seen at other similar events. In fact, I spent most of the afternoon chatting to various different people about all sorts of IT-related topics (and beyond), and learnt a lot of new things in the process.

Overall, the event was a great success (unlike the trains and replacement buses back to Manchester!) and at £5 the price was accessible to all. Plans are afoot to run OpenTech again next year, so keep an eye on the UKUUG website for details. In the meantime, photos and recordings of most of the talks from this year are available online at:

`http://www.ukuug.org/events/opentech2008/`

Linux kernel resource allocation in virtualized environments

Matthew Sacks

The behaviour of the Linux kernel and its resource allocation methods are an art and science, more the former than the latter. When working with Linux in a virtualized environment, the complexities of the Linux kernel's resource allocation algorithms increase. New performance issues may arise and proper functionality can come to a halt – especially on overutilized systems. The Linux kernel behaves differently on a virtualized platform in comparison to bare-metal. Why Linux behaves differently on a virtual platform and how to address performance and stability issues when it starts to malfunction or degrade in performance relate, but do not depend on, the environment. The solution presented here is not that of changing the environment, but, rather, that of making adjustments to the Linux kernel to coalesce with the hypervisor.

The Environment

The virtual environment comprised 6 VMware ESX 3.01 servers running approximately 11 virtual machines per server. Each ESX server had Red Hat Enterprise Linux 4 Update 4 machines running a wide array of application and Web servers. The environment was intended to simulate a high-volume, high-traffic production Web site by simulating load tests on the virtual servers. The phenomenon experienced was the Linux Kernel's OOM-Kill function, which would trigger and kill processes that were consuming the most resources. How the ESX server interacts with this Linux kernel in allocating resources is the starting point.

VMWare ESX Server resource allocation

> *NOTE: VMware's ESX platform is certainly not the be-all and end-all of virtualization. However, it is widely used and accepted. The same principles used in this example may apply to other virtual platforms. The tuning methods provided in this article are not intended as a replacement for good capacity planning.*

The VMware ESX server adds another layer of abstraction between the Linux server's physical and virtual memory and the real memory of the ESX server. The ESX server creates additional memory overhead in managing the virtual devices, CPUs, and memory of the virtual machine. It can be thought of as virtual memory that manages virtual memory: a new set of resources that must be managed on top of the guest operating system's own virtual memory management algorithms. Resources can run thin quickly and resource allocation issues tend to increase faster on a virtual server than on a bare-metal server.

For example, consider an ESX server with 1 GB of memory, running two virtual machines with 256 MB of "physical" memory allocated for each virtual server. The amount of free resources available is approximately 170 MB. The service console uses approximately 272 MB, the VMkenel uses somewhat less than that, and, depending on how many virtual CPUs and devices are added to each virtual server, the memory overhead increases.

ESX uses a proprietary memory ballooning algorithm to adjust and allocate memory to virtual servers. ESX loads a driver into the virtual server which modifies the virtual server's page-claiming features. It increases or decreases memory pressure depending on the available physical resources of the ESX server, causing the guest to invoke its own memory management algorithms. When memory is low the virtual server's OS decides which pages to reclaim and may swap them to its virtual swap.

However, sometimes pages cannot be reclaimed fast enough or memory usage grows faster than can be committed to swap; then the Linux OS kills processes and "Out of Memory" errors appear in the syslog.

The Linux Out of Memory Killer

The `out_of_memory()` function is invoked by `alloc_pages()` when free memory is very low and the Page Frame Resource Allocation Algorithm has not succeeded in reclaiming any page frames. The function invokes `select_bad_process()` to select a victim and them invokes the `oom_kill_process()` to kill the process that is utilising the most resources. Typically, the `select_bad_process()` function chooses the process that is not a critical system process and is consuming the largest number of page frames. This is why, when running a resource-intensive application or Web server on a virtual environment, the application or Web server may begin crashing frequently. Check the logs for the "Out of Memory" errors to see if the `oom_kill_process()` function is being called by the Linux kernel. The `oom_kill_process()` function comes into play because of how Linux is allocating memory into lower memory zones.

Memory Zones and the Dirty Ratio

By default, the Linux kernel allows addressing of memory in the lower zone called `ZONE_DMA`. This zone contains page frames of memory below 16 MB. In high workloads, once the `ZONE_NORMAL` (normal memory zone) and `ZONE_HIGHMEM` have been exhausted by an application, it will begin to allocate memory from `ZONE_DMA` and the requestor of the application will pin them, thereby denying access to these zones by other critical system processes. The `lower_zone_protection` kernel parameter determines how aggressive the kernel is in defending the lower memory allocation zone.

The dirty ratio is a value expressed in percentage of system memory at which limit processes generating dirty buffers will write data to disk rather than relying on the pdflush daemons to perform this function. The pdflush kernel thread scans the page cache looking for dirty pages (pages that the kernel has set to be swapped to disk) and then ensures that no page remains dirty for too long. `ZONE_DMA` can be protected from being utilized by applications and the dirty ratio can be adjusted by tuning the Linux kernel.

Tuning the Linux Kernel for Virtualization

The `/etc/sysctl.conf` file allows modification of select kernel settings without recompilation of the kernel. The `/etc/sysctl.conf` file is used to adjust behaviors of the Linux kernel to address issues with resource allocation. A set of virtual memory tunable parameters is available for tuning from within this file. Two tunable virtual memory parameters in particular will solve the “Out of Memory” problems and most other problems with memory allocation in a virtual Linux server. To protect the lower zones of memory from being utilized by the applications on a virtual Linux server, edit the `/etc/sysctl.conf` file to include the following parameter: `lower_zone_protection=100`

To increase the ratio by which the pdflush kernel thread scans the page cache to look for dirty pages to 5 percent of the system memory, edit `/etc/sysctl.conf` to include the parameter: `dirty_ratio=5`

Reboot the system or type the command `sysctl -p` so that the new kernel settings will take effect. Now most memory resource allocation issues should be resolved in a virtualized Linux environment. Tuning these few settings provides a small insight into how tuning the Linux kernel can solve performance-related problems in a virtualized environment. As a result of the tuning changes, “Out of Memory” errors are reduced dramatically in scope and frequency, and virtual memory is utilized more effectively.

Conclusion

There are numerous algorithms at work with VMware’s ESX server and within the Linux kernel itself. In a low-volume environment the standard configurations and settings may be sufficient. In a high-volume, high-performance environment where load tests are constantly making requests against application and Web servers, the defaults are typically insufficient. To squeeze the maximum amount of performance out of a system, an understanding of the underlying algorithms and behaviors of the ESX server is essential before tuning the guest operating system’s kernel. The end result is maximal performance on an otherwise overutilized or poorly performing virtual environment. The key is to understand which algorithms need to be changed and to set them to the right values. This is where kernel tuning becomes more of an art than a science.

Acknowledgments

I want to acknowledge the Systems Administration Team at Edmunds.com, Safdar Husain, David Wolfe, David Morgan, Nikki Sonpar, and Eric Theis, who all contributed in some way to this article.

References

[1] D. Bovet and M. Cesati, Understanding the Linux Kernel (Sebastopol, CA: O’Reilly & Associates, 2005).

[2] B. Matthews and N. Murray, “Virtual Memory Behavior in Red Hat Linux A.S. 2.1”, Red Hat white paper, Raleigh, NC, 2001.

[3] N. Horman, “Understanding Virtual Memory in Red Hat Enterprise Linux 4”, Red Hat white paper, Raleigh, NC, 2005.

*Originally published in **;login: The USENIX Magazine**, vol. 33, no. 2(Berkeley, CA: USENIX Association, 2007). Reprinted by kind permission.*

Fear and Loathing in the Routing System

Joe Abley

Anycast is a strange animal. In some circles the merest mention of the word can leave you drenched in bile; in others it’s an over-used buzzword which triggers involuntary rolling of the eyes. It’s a technique, or perhaps a tool, or maybe a revolting subversion of all that is good in the world. It is “here to stay.” It is by turns “useful” and “harmful”; it “improves service stability,” “protects against denial-of-service attacks,” and “is fundamentally incompatible with any service that uses TCP.”

That a dry and, frankly, relatively trivial routing trick could engender this degree of emotional outpouring will be unsurprising to those who have worked in systems or network engineering roles for longer than about six minutes. The violently divergent opinions are an indication that context matters with anycast more than might be immediately apparent, and since anycast presents a very general solution to a large and varied set of potential problems, this is perhaps to be expected.

The trick to understanding anycast is to concentrate less on the “how” and far more on the “why” and “when.” But before we get to that, let’s start with a brief primer. Those who are feeling a need to roll their eyes already can go and wait outside. I’ll call you when this bit is done.

Nuts and Bolts

Think of a network service which is bound to a greater extent than you’d quite like to an IP address rather than a name. DNS and NTP servers are good examples, if you’re struggling to paint the mental image. Renumbering servers is an irritating process at the best of times, but if your clients almost always make reference to those servers using hard-coded IP addresses instead of names, the pain is far greater.

Before the average administrator has acquired even a small handful of battle scars from dealing with such services, it’s fairly common for the services to be detached from the physical servers that house them. If you can point NTP traffic for 204.152.184.72 at any server you feel like, moving the corresponding service around as individual servers come and go becomes trivially easy. The IP address in this case becomes an identifier, like a DNS name, detached from the address of the server that happens to be running the server processes on this particular afternoon.

With this separation between service address and server address, a smooth transition of this NTP service from server A to server B within the same network is possible with minimal downtime to clients. The steps are:

> 1) Make sure the service running on both servers is identical. In the case of an NTP service, that means that both machines are running appropriate NTP software and that their clocks are properly synchronised.
>
> 2) Add a route to send traffic with destination address 204.152.184.72 toward server B.
>
> 3) Remove the route that is sending traffic toward server A.

Ta-da! Transition complete. Clients didn’t notice. No need for a maintenance window. Knowing smiles and thoughtful nodding all round.

To understand how this has any relevance to the subject at hand, let’s insert another step into this process:

> 2.5) Become distracted by a particularly inflammatory slashdot comment, spend the rest of the day grumbling about the lamentable state of the server budget for Q4, and leave the office at 11 p.m. as usual, forgetting all about step 3.

The curious result here is that the end result might very well be the same: Clients didn’t notice. There is no real need for a maintenance window. What’s more, we can now remove either one of those static routes and turn off the corresponding server, and clients still won’t notice. We have distributed the NTP service across two origin servers using anycast. And we didn’t even break a sweat!

Why does this work? Well, a query packet sent to a destination address arrives at a server which is configured to accept and process that query, and the server answers. Each server is configured to reply, and the source address used each time is the service address. The fact that there is more than one server available doesn’t actually matter. To the client (and, in fact, to each server), it looks like there is only one server. The query-response behaviour is exactly as it was without anycast on the client and on the server. The only difference is that the routing system has more than one choice about toward which server to send the request packet.

(To those in the audience who are getting a little agitated about my use of a stateless, single-packet exchange as an example here, there is no need to fret. I’ll be pointing out the flies in the ointment very soon.)

The ability to remove a dependency on a single server for a service is very attractive to most system administrators, since once the coupling between service and server has been loosened, intrusive server maintenance without notice (and within normal working hours) suddenly becomes a distinct possibility. Adding extra server capacity during times of high service traffic without downtime is a useful capability, as is the ability to add additional servers.

For these kinds of transitions to be automatic, the interaction between the routing system and the servers needs to be dynamic: that is, a server needs to be able to tell the routing system when it is ready to receive traffic destined for a particular service, and correspondingly it also needs to be able to tell the routing system when that traffic should stop. This signalling can be made to work directly between a server and a router using standard routing protocols, as described in ISC-TN-2004-1 [1] (also presented at USENIX '04 [2]). This approach can also be combined with load balancers (sometimes called "layer-4 switches") if the idea of servers participating in routing protocols directly is distasteful for local policy reasons.

This technique can be used to build a cluster of servers in a single location to provide a particular service, or to distribute a service across servers that are widely distributed throughout your network, or both. With a little extra attention paid to addressing, it can also be used to distribute a single service around the Internet, as described in ISC-TN-2003-1 [3].

Anycast Marketing

Some of the benefits to the system administrator of distributing a service using anycast have already been mentioned. However, making the lives of system administrators easier rarely tops anybody's quarterly objectives, much as you might wish otherwise. If anycast doesn't make the service better in some way, there's little opportunity to balance the cost of doing it.

So what are the tangible synergies? What benefits can we whiteboard proactively, moving forward? Where are the bullet points? Do you like my tie? It's new!

Distributing a service around a network has the potential to improve service availability, since the redundancy inherent in using multiple origin servers affords some protection from server failure. For a service that has bad failure characteristics (e.g., a service that many other systems depend on) this might be justification enough to get things moving.

Moving the origin server closer to the community of clients that use it has the potential to improve response times and to keep traffic off expensive wide-area links. There might also be opportunities to keep a service running in a part of your network that is afflicted by failures in wide-area links in a way that wouldn't otherwise be possible.

For services deployed over the Internet, as well as nobody knowing whether you're a dog, there's the additional annoyance and cost of receiving all kinds of junk traffic that you didn't ask for. Depending on how big a target you have painted on your forehead, the unwanted packets might be a constant drone of backscatter, or they might be a searing beam of laser-like pain that makes you cry like a baby. Either way, it's traffic that you'd ideally like to sink as close to the source as possible, ideally over paths that are as cheap as possible. Anycast might well be your friend.

Flies in the Ointment

The architectural problem with anycast for use as a general-purpose service distribution mechanism results from the flagrant abuse of packet delivery semantics and addressing that the technique involves. It's a hack, and as with any hack, it's important to understand where the boundaries of normal operation are being stretched.

Most protocol exchanges between clients and servers on the Internet involve more than one packet being sent in each direction, and most also involve state being retained between subsequent packets on the server side. Take a normal TCP session establishment handshake, for example:

- Client sends a SYN to a server.
- Server receives the SYN and replies with a SYN-ACK.
- Client receives the SYN-ACK and replies with an ACK.
- Server receives the ACK, and the TCP session state on both client and server is "ESTABLISHED."

This exchange relies on the fact that "server" is the same host throughout the exchange. If that assumption turns out to be wrong, then this happens:

- Client sends a SYN to server A.

- Server A receives the SYN and replies with a SYN-ACK.
- Client receives the SYN-ACK and replies to the service address with an ACK.
- Server B receives the ACK and discards it, because it has no corresponding session in “SYN-RECEIVED.”

At the end of this exchange, the client is stuck in “SYN-SENT,” server A is stuck in “SYN-RECEIVED,” and server B has no session state at all. Clearly this does not satisfy the original goal of making things more robust; in fact, under even modest query load from perfectly legitimate clients, the view from the servers is remarkably similar to that of an incoming SYN flood.

It’s reasonable to wonder what would cause packets to be split between servers in this way, because if that behaviour can be prevented perhaps the original benefits of distributed services that gave us all those warm fuzzies can be realised without inadvertently causing our own clients to attack us. The answer lies in the slightly mysterious realm of routing.

The IP routing tables most familiar to system administrators are likely to be relatively brief and happily uncontaminated with complication. A single default route might well suffice for many hosts, for example; the minimal size of that routing table is a reflection of the trivial network topology in which the server is directly involved. If there’s only one option for where to send a packet, that’s the option you take. Easy.

Routers, however, are frequently deployed in much more complicated networks, and the decision about where to send any particular packet is correspondingly more involved. In particular, a router might find itself in a part of the network where there is more than one viable next hop toward which to send a packet; even with additional attributes attached to individual routes, allowing routers to prioritise one routing table entry over another, there remains the distinct possibility that a destination address might be reached equally well by following any one of several candidate routes. This situation calls for Equal-Cost Multi-Path (ECMP) routing.

Without anycast in the picture, so long as the packets ultimately arrive at the same destination, ECMP is probably no cause for lost sleep. If the destination address is anycast, however, there’s the possibility that different candidate routes will lead to different servers, and therein lies the rub.

Horses for Courses

So, is anycast a suitable approach to making services more reliable? Well, yes and no. Maybe. Maybe not, too. Oh, it’s all so vague! I crave certainty! And caffeine-rich beverages!

The core difficulty that leads to all this weak hand-waving is that it’s very difficult to offer a general answer when the topology of even your own network depends on the perspective from which it is viewed. When you start considering internetworks such as, well, the Internet, the problem of formulating a useful general answer stops being simply hard and instead becomes intractable.

From an architectural perspective, the general answer is that for general purpose services and protocols, anycast doesn’t work. Although this is mathematically correct (in the sense that the general case must apply to all possible scenarios), it flies in the face of practical observations and hence doesn’t really get us anywhere. Anycast is used today in applications ranging from the single-packet exchanges of the DNS protocol to multi-hour, streaming audio and video. So it does work, even though in the general case it can’t possibly.

The fast path to sanity is to forget about neat, simple answers to general questions and concentrate instead on specifics. Just because anycast cannot claim to be generally applicable doesn’t mean it doesn’t have valid applications.

First, consider the low-hanging fruit. A service that involves a single-packet, stateless transaction is most likely ideally suited to distribution using anycast. Any amount of oscillation in the routing system between origin servers is irrelevant, because the protocol simply doesn’t care which server processes each request, so long as it can get an answer.

The most straightforward example of a service that fits these criteria is DNS service using UDP transport. Since the overwhelming majority of DNS traffic on the Internet is carried over UDP, it’s perhaps unsurprising to see anycast widely used by so many DNS server administrators.

As we move on to consider more complicated protocols – in particular, protocols that require state to be kept between successive packets – let’s make our lives easy and restrict our imaginings to very simple networks whose behaviour is well understood. If our goal is to ensure that successive packets within the same client-server exchange are carried between the same client and the same origin server for the duration of the transaction, there are some tools we can employ.

We can arrange for our network topology to be simple, such that multiple candidate paths to the same destination don't exist. The extent to which this is possible might well depend on more services than just yours, but then the topology also depends to a large extent on the angle you view it from. It's time to spend some time under the table, squinting at the wiring closet. (But perhaps wait until everybody else has gone home, first.)

We can choose ECMP algorithms on routers that have behaviour consistent with what we're looking for. Cisco routers, for example, with CEF (Cisco Express Forwarding) turned on, will hash pertinent details of a packet's header and divide the answer space by the number of candidate routes available. Other vendors' routers have similar capabilities. If the computed hash is in the first half of the space, you choose the left-hand route; if the answer is in the other half, you choose the right-hand route. So long as the hash is computed over enough header variables (e.g., source address and port, destination address and port) the route chosen ought to be consistent for any particular conversation ("flow," in router-ese).

When it comes to deploying services using anycast across other people's networks (e.g., between far-flung corners of the Internet), there is little certainty in architecture, topology, or network design and we need instead to concentrate our thinking in terms of probability: We need to assess benefit in the context of risk.

> *Internet, n: "the largest equivalence class in the reflexive transitive symmetric closure of the relationship 'can be reached by an IP packet from' " (Seth Breidbart).*

The world contains many hosts that consider themselves connected to the Internet. However, that "Internet" is different, in general, for every host – it's a simple truism that not all the nodes in the world that believe themselves to be part of "the" Internet can exchange packets with each other, and that's even without our considering the impact of packet filters and network address translation. The Internet is a giant, seething ball of misconfigured packet filters, routing loops, and black holes, and it's important to acknowledge this so that the risks of service deployment using anycast can be put into appropriate context.

A service that involves stateful, multi-packet exchanges between clients and servers on the Internet, deployed in a single location without anycast, will be unavailable for a certain proportion of hosts at any time. You can sometimes see signs of this in Web server and mail logs in the case of asymmetric failures (e.g., sessions that are initiated but never get established); other failure modes might relate to control failures (e.g., the unwise blanket denial of ICMP packets in firewalls which so often breaks Path MTU Discovery). In other cases the unavailability might have less mysterious origins, such as a failed circuit to a transit provider which leaves an ISP's clients only able to reach resources via peer networks.

Distributing the same service using anycast can eliminate or mitigate some of these problems, while introducing others. Access to a local anycast node via a peer might allow service to be maintained to an ISP with a transit failure, for example, but might also make the service vulnerable to rapid changes in the global routing system, which results in packets from a single client switching nodes, with corresponding loss of server-side state. At layer-9, anycast deployment of service might increase costs in server management, data center rental, shipping, and service monitoring, but it might also dramatically reduce Internet access charges by shifting the content closer to the consumer. As with most real-life decisions, everything is a little grey, and one size does not fit all.

Go West, Young Man

So, suppose you're the administrator of a service on the Internet. Your technical staff have decided that anycast could make their lives easier, or perhaps the pointy-haired guy on the ninth floor heard on the golf course that anycast is new and good and wants to know when it will be rolled out so he can enjoy his own puffery the next time he's struggling to maintain par on the eighth hole. What to do?

First, there's some guidance that was produced in the IETF by a group of contributors who have real experience in running anycast services. That the text of RFC 4786 [4] made it through the slings and arrows of outrageous run-on threads and appeals through the IETF process ought to count for something, in my opinion (although as a co-author my opinion is certainly biased).

Second, run a trial. No amount of theorizing can compete with real-world experience. If you want to know whether a Web server hosting images can be safely distributed around a particular network, try it out and see what happens. Find some poor victim of the slashdot effect and offer to host her page on your server, and watch your logs. Grep your `netstat -an` and look for stalled TCP sessions that might indicate a problem.

Third, think about what problems anycast could introduce, and consider ways to minimize the impact on the service or to provide a fall-back to allow the problems to be worked around. If your service involves HTTP, consider using a redirect on the anycast-distributed server that directs clients at a non-anycast URL at a specific node. Similar options exist with some streaming media servers. If you can make the transaction between clients and the anycast service as brief as possible, you might insulate against periodic routing instability that would be more likely to interrupt longer sessions.

Fourth, consider that there are some combinations of service, protocol, and network topology that will never be good environments for anycast to work. Anycast is no magic wand; to paraphrase the WOPR [5], sometimes the only way to win is not to play.

References

[1] J. Abley, "A Software Approach to Distributing Request for DNS Service Using GNU Zebra, ISC BIND9 and FreeBSD", ISC-TN-2004-1, March 2004:
`http://www.isc.org/pubs/tn/isc-tn-2004-1.html`

[2] USENIX Annual Technical Conference (USENIX '04) report, ;login:, October 2004, page 52.

[3] J. Abley, "Hierarchical Anycast for Global Service Distribution", ISC-TN2003-1, March 2003:
`http://www.isc.org/pubs/tn/isc-tn-2003-1.html`

[4] J. Abley and K. Lindqvist, "Operation of Anycast Services", RFC 4786, December 2006. [5] "War Operation Plan Response":
`http://en.wikipedia.org/wiki/WarGames`

Originally published in ***;login: The USENIX Magazine****, vol. 33, no. 1 (Berkeley, CA: USENIX Association, 2007). Reprinted by kind permission.*

Hardware review: Fujitsu Siemens Esprimo P2411

Martin Houston

I was looking for a cheap PC to replace an ailing 4 year old Dell belonging to a friend, and I spotted this on ebuyer:

> *Fujitsu Siemens Esprimo P2411 Sempron 3800, 1GB, Burner, 80GB HDD, Linux.*

In fact the ONLY PC they seem to sell that does not run Vista (apart from the Asus EEEpcs of course). The 'Vistaless' price was rather attractive too – under £ 140.

The rest of the spec is pretty low but a 2GHz Sempron ain't no slouch when freed from the penal servitude of running Vista.

1GB of memory is hardly to be sniffed at either.

I went ahead and ordered one and was stunned by the quality of what I got for the measly amount of money. The 'Built in Germany' PC exudes quality. NVIDIA 3D graphics, a nice fast DVD writer, and quiet too.

The only slight niggle was that the hard disk arrived empty. Linux was sitting on a DVD ROM in the packaging with a rather nice quality USB keyboard and mouse. The other problem was that the Linux was a rather old Fedora Core. Not that much of a problem until I realised that the DVD writer was a new fangled SATA one. Original Fedora core is just too old to know about SATA DVD drives. Such a silly mistake one wonders if some skulduggery was at work to get a Fujitsu-Siemens employee to make it :)

However the machine is a gem if you put a Linux DVD in it from the past couple of years. I tried FC9 32bit and that installed like a dream. Then I realised that this Sempron is really a 64 bit Athlon! The 64 bit build of Suse 11.0 also went on equally easily.

The only thing that this machine lacks compared to one 2 or 3 times the price is that the Sempron is not capable of Xen being able to run 'native' virtual machines (i.e. Windows as a guest) – that was easily solved by a quick download of VMware player. VMware may be a little slower but doing things in software allows it to work with cheaper hardware.

Its an ideal machine to get people to try Linux out on. For one thing they are not paying the Vista tax so there is no temptation to think "this has cost me a lot, I may as well try to get my moneys worth". It's modern and practical entry level, in that it is fast enough to be a joy to use until you get sophisticated enough to want to hammer the CPU for hours on end.

Java Pocket Guide

Robert Liguori and Patricia Liguori

O'Reilly Media
ISBN: 978-0-596-51419-8
178pp.
£ 8.99
Published: 5th March 2008

reviewed by Paul Waring

Having spent the last two months trying to make other peoples' Java code talk to each other, with Tomcat thrown into the mix for good measure, any guide which claims not only to cover the essentials of Java but also fit into my pocket – albeit rather a large one – has definitely got my attention. With references for all the basic types, quick descriptions of abstract and static methods and other fundamental concepts, this is the sort of book which you dive into for a specific answer – almost like a scaled down version of the popular Java Cookbook - rather than reading it from cover to cover.

Where I can see this guide being particularly useful though is when you need your mind jogging as to how to perform a task which seems simple and routine but which requires creating half a dozen different objects. A perfect example of this is writing to or reading from a file on disk – I can never remember which of the 20+ input and output classes I should be using, and what order I have to create objects in just to open a file stream. A quick scan of the relevant pages in the guide reveals the answer, which is far less hassle than working through all the relevant online documentation.

The only minor issue I found with this text was that the margins are quite small, which means reading the text near to the binding is sometimes a little tricky. I'm also not sure that I agree with the inclusion of a section on naming conventions, as these are largely a matter of personal preference and house style – though Sun does define a set of standards for code in the core Java distributions. These are small issues though, which can be worked around by breaking the spine (likely to happen through heavy use anyway) and skipping the coding standards section.

Overall, this book is a useful text to have to hand for anyone doing a significant amount of Java programming, and one which already has a space on my desk rather than the bookshelf.

CakePHP Application Development

Ahsanul Bari and Anupom Syam

Packt Publishing
ISBN: 978-1-847-19389-6
332pp.
£ 24.99
Published: 30th June 2008

reviewed by Paul Waring

For those who haven't heard of CakePHP, it is an open source (MIT Licence) framework for building PHP applications. Frameworks appear to be, for better or worse, in vogue at the moment, and they promise to take care of all the boring repetitive code which is required for most applications, such as form generation, validation of user input and session handling. As someone who has written this sort of code time and time again, I'd be happy for any piece of software to take this tedious load away from me. However, the documentation for these frameworks is often best described as an API listing rather than an introduction to using the code, so hopefully this book will fill that gap.

Like most texts on frameworks, this book begins with a brief explanation of the Model View Controller (MVC) pattern. This is kept sufficiently short, and is described in a way which is clearly aimed at existing PHP developers, rather than generic software engineers – a major plus point for me. A short chapter follows on how to install Cake, which is once again to the point and thankfully doesn't go into swathes of detail on how to configuring every possible web server which you might wish to run the software under (though tips on setting up Apache for Cake are included).

Following the quickest introduction and installation chapters I've seen for some time, the book launches straight into developing your first application. After a clear and concise explanation of Cake naming conventions – which is part of what enables the framework to automatically generate the bulk of your code – a simple 'todo' list is up and running. The rest of the chapter runs through the basic operations – create, update and delete records – with data verification thrown in for good measure.

The next few chapters begin to delve deeper into Cake, but most of the sections are backed up by clear code examples which gradually build on what has gone before, and you can slowly see a complete book store management system begin to appear. Chapter 8 provides a solid overview of using the console to automatically generate Cake applications, saving even more typing. The final three chapters take you through the development of a more complicated application, which introduces user authentication, AJAX and other minor improvements such as search and RSS feeds.

Overall, this book is the best introduction to a PHP framework that I've seen so far. The only minor niggle I have is that it seems to stop abruptly at the end of chapter 12 – there is no concluding chapter with ideas for future projects, more in-depth resources to look at etc. – but heading to the CakePHP website (`www.cakephp.org`) will take you to the next step. If you want to get started with PHP frameworks, this is one of the books to look at – and a percentage of the price even goes to the CakePHP project.

The Gift: How the Creative Spirit Transforms the World

Lewis Hyde
Canongate Books
ISBN: 978-1-841-95993-1
352pp.
£ 8.99
Published: 6th September 2007

reviewed by Roger Whittaker

This book is hard to classify: the author himself devotes the book's foreward to explaining why it doesn't easily fit into any of the usual categories. After admitting the difficulty for his publisher of describing the book in a short blurb, the last sentence in the foreward is:

> *And if the salesman want to pitch it as 'Bad-boy critic deploys magic charm against vampire economy', that's all right with me.*

The book dates from 1979, but has recently been republished in the UK. Its subject matter is not Free Software. It includes discussion of (among other things): Celtic folklore, the anthropology of the native North American peoples, the history and theology of usury before and after the Reformation, the ethics of organ donation, the poetry of Walt Whitman and Ezra Pound, the early history of patents and a great deal more.

Only in the "afterword" provided for this British edition does the author discuss in passing matters guaranteed to be of direct interest to readers of this newsletter: the current period for which copyright is protected in the US, publishing of scientific papers under permissive licences, and the briefest possible passing mention of open source.

All that being said, "The Gift" is deeply relevant to our concerns. Eric Raymond and others have characterised the Open Source and Free Software communities as "gift economies" as opposed to "market economies". Hyde describes what that concept means from an anthropological point of view, and also makes a case for his view that serious artistic expression is by its nature a gift rather than a commodity.

In a gift economy, the person who gives most is the one with the greatest status, not the one who has the most. Similarly, in a gift economy, the worst sin is to consume or hide away the gift that has been received: gifts must continually be passed on.

The parallels with the Free and Open Source software world are very clear, and the kind of “sacred obligation” entailed by gifts in traditional societies is mirrored in the very strong feelings in our communities about how the gifts that we have received should be used and transmitted.

Although the replacement of tribal and local economies by a global one has converted many of our relationships into purely commercial ones, the examples that Hyde gives of organ donation, artistic expression and scientific publishing show how there is a very strong and persistent feeling that some things can only be transmitted and received in the form of gifts. Such things change the nature of the relationship between the parties involved. New forms of communication (the Internet) and new ways of insisting on the obligations that a gift traditionally carries with it (the GPL) have re-enabled older and wiser ways of organising communities. The book helps to put that process into a wider historical and cultural context.

Intellectual Property and Open Source

Van Lindberg

O’Reilly Media
ISBN: 978-0-596-51796-0
390pp.
£ 21.99
Published: 29th July 2008

reviewed by Roger Whittaker

Almost my first thought on seeing this book was that out of the five words in the title, Richard Stallman would approve only the third and shortest one: the word “and”. It is quite clear from the preface, however, that the author is fully aware of the awkward issues that surround terminology in this area: his “note about terminology” refers to the FSF’s “phrases that are worth avoiding” and also addresses directly the question of “free software” versus “open source”. He explains that part of his choice of wording by saying

> *Where applicable, I will use the correct term to describe how they are both socially and legally different. Nevertheless, because open source software is a strict superset of free software, I will generally use the more inclusive term when discussing legal elements common to both.*

Stallman’s primary objection to the use of the term “Intellectual Property” is that it is often used in such a way as to blur and confuse the differences between the very different concepts of copyright, trademarks, patents and trade secrets. Van Lindberg can certainly plead not guilty to that one: the book defines its terms extremely clearly, and discusses the various concepts separately and in depth, while also describing the way they interact.

In the first chapter he defines different types of good: rivalrous and non-rivalrous goods, excludable and no-excludable goods, private goods, public goods, common-pool goods and club goods, and goes on to examine the legal concept of property. I found these short sections very interesting and enlightening, because they made me realise that I had never really analysed the underlying concepts in any depth.

The same kind of clarity is applied to all the concepts described in the book. The author is clearly a person with a close knowledge both of the law and of the world of software. He uses interesting and sometimes surprising analogies to illustrate legal concepts and practices. For example, he observes that patent applications use indentation to clarify structure in a manner similar to coding conventions. Elsewhere he uses a Simpsons story line to help explain the concept of trade secrets, draws a parallel between credit unions and open source, compares Red Hat’s patent policy and India’s nuclear strategy, and sees a parallel between contract law and a distributed source code management system.

Many of the most controversial and notorious cases of recent years appear as examples: for instance a discussion of the “GIF patent”. In the section on trademarks, there is a clear description of why AOL really had no choice but to pursue what was then called GAIM over infringement of the trademark it held on the name “AIM”.

The book does not attempt to cover the legal issues internationally: the descriptions of the legal situation are all concerned with US law. This means that the sections on patents and copyright both describe different laws from those which apply in the UK and Europe. However, the world of software is an international one, and the one jurisdiction that matters in that world is that of the United States (which has been assiduously attempting to force its intellectual property laws on the rest of the world for some time now). As it is in the US that many of the most important and significant battles for free and open source software are being played out, understanding those battles is largely a matter of understanding the American legal position. So this book is useful and appropriate outside the US, but readers in this country will want to compare with some other source of information about the situation here. The copyright sections were the area where I felt the need for a British parallel text most strongly.

As would be expected there is considerable discussion of the various free and open source licences, and in particular the current state of the law as it applies to the GPL. Appendices include the text of the more important licences, and there are tables showing the interactions and compatibility between them.

Despite my slight reservation about the "US-only" descriptions of the relevant laws, this is a valuable book which should be on the shelf of anyone who is interested in the issues it covers.

Contributors

Joe Abley is the Director of Operations at Afilias Canada, a DNS registry company, and a technical volunteer at Internet Systems Consortium. He likes his coffee short, strong, and black and is profoundly wary of outdoor temperatures that exceed 20°C.

Martin Houston looks after systems for Man Investments, and is a long-standing UKUUG member. He was the original founder of the UKUUG Linux SIG and was also involved in setting up the UK C users group (now ACCU).

Jane Morrison is Company Secretary and Administrator for UKUUG, and manages the UKUUG office at the Manor House in Buntingford. She has been involved with UKUUG administration since 1987. In addition to UKUUG, Jane is Company Secretary for a trade association (Fibreoptic Industry Association) that she also runs from the Manor House office.

Matthew Sacks works as a systems administrator at Edmunds.com. His focus is network, systems, and application performance tuning.

Peter H Salus has been (inter alia) the Executive Director of the USENIX Association and Vice President of the Free Software Foundation. He is the author of *A Quarter Century of Unix* (1994) and other books.

Paul Thompson has supported many diverse technologies and operating systems over the years but is now firmly hooked on SUSE Linux which he supports for Novell at a City bank. . . but he still dabbles with Solaris in his spare time.

Paul Waring is a PhD student at the University of Manchester and the newest member of the UKUUG council. He is also responsible for organising the UKUUG Spring conference in 2009.

Alain Williams is UKUUG Chairman, and proprietor of Parliament Hill Computers Ltd, an independent consultancy.

Roger Whittaker works for Novell Technical Services at Bracknell and is the UKUUG Newsletter Editor.

Contacts

Alain Williams
Council Chairman
Watford
Tel: 07876 680256

Sam Smith
UKUUG Treasurer; Website
Manchester

John M Collins
Council member
Welwyn Garden City

Phil Hands
Council member
London

John Pinner
Council member
Sutton Coldfield

Howard Thomson
Council member
Ashford, Middlesex

Paul Waring
Council member
Manchester

Jane Morrison
UKUUG Secretariat
PO Box 37
Buntingford
Herts
SG9 9UQ
Tel: 01763 273475
Fax: 01763 273255
office@ukuug.org

Sunil Das
UKUUG Liaison Officer
Suffolk

Roger Whittaker
Newsletter Editor
London