news@UK

The Newsletter of UKUUG, the UK's Unix and Open Systems Users Group
Published electronically at `http://www.ukuug.org/newsletter/`

Volume 15, Number 2 **ISSN 0965-9412** **June 2006**

Contents

News from the Secretariat

Jane Morrison

It has been another busy time for UKUUG. Since the end of March we have been working hard on the organisation of the Linux 2006 Conference which will be held this year in Brighton at the University of Sussex.

The conference takes place over three days between Friday 30th June and Sunday 2nd July. On the day before the conference proper (Thursday 29th June) there will be a choice of tutorials: two half day tutorials (on MySQL optimisation and SystemTap) and a full day tutorial on building RPM packages. Members should already have received the information booklet and booking form. Up-to-date details are at:
`http://www.ukuug.org/events/linux2006/`

Delegate bookings are arriving each day. If you want to take advantage of the University Halls B&B option please make your booking as soon as possible.

On 20th April in conjunction with Apple we organised a one day meeting in London 'OS X for Intel'. This free event was kindly sponsored by Apple and was well attended.

The UKUUG Annual General Meeting which will be held in September. At the time of going to press, the exact date has not been fixed: we are hoping to combine the AGM with an interesting evening event in London. Further details will be sent out in due course.

The next Winter/Spring Conference is planned for March 2007. The venue has not been decided yet, but we are currently looking at possibilities in Manchester.

The copy date for the next newsletter is 25th August.

Chairman's Report

Ray Miller

UKUUG's annual Large Installation Systems Administration (LISA) conference was held earlier this year in Durham. The event was a great success, attracting almost 100 delegates from all parts of the UK and a fair contingent from mainland Europe. This year, in addition to the main LISA stream, we hosted a BSD MiniCon with invited speakers from the BSD community. This attracted a number of new faces, and we hope to build on this success to put together events that appeal to users and professionals from all corners of the Unix world.

We are pleased to announce the winner of the prize for best paper. This goes to Simon Wilkinson of the School of Informatics, University of Edinburgh, for his paper "Kerberizing Our Network". Simon wins £125 of O'Reilly books. The runners-up prize goes to Robert Watson of the FreeBSD project, who wins £75 of O'Reilly books for his paper "TrustedBSD OpenBSM". We are grateful to O'Reilly UK for sponsoring these prizes. Tutorial and workshop notes, along with papers and slides submitted by speakers, are available on the CD accompanying this newsletter.

Feedback from delegates was generally very positive, with Gerald Carter's Samba tutorial proving particularly popular. On the downside, we received a lot of complaints about the lack of wireless access in the conference venue. Although this was not advertised as part of the conference package, it is clear that wireless internet access is now taken for granted by delegates. We will be sure to bear this in mind when deciding the venue for our next conference.

I look forward to meeting more of you at future UKUUG events.

The Newsletter

Roger Whittaker

This issue of the newsletter is the largest that we have produced for some time.

We want to include content that is of interest to our members and which they are unlikely to have come across elsewhere. Original articles from members are always very welcome as well as suggestions of relevant items published elsewhere which could be reprinted.

Note that the prices quoted for books in the newsletter reviews are the full prices: UKUUG members are entitled to a 27.5% discount on O'Reilly books (30% for on-line orders) as well as similar substantial discounts on books published by Pearson Education and by Wiley. We are currently in discussion with other publishers about providing relevant books for review.

Any articles or other suggestions should be sent to:
`newsletter@ukuug.org`

From the UKUUG Diary

The UKUUG maintains a web diary of future events of interest at
`http://www.ukuug.org/diary/`

The following events are a small selection of those currently listed.

Institutional Web Management Workshop 2006

14th June 2006: Bath

The event will provide an opportunity for those involved in the provision of institutional Web services to hear about institutional case studies, national initiatives and emerging technologies and to actively participate in a number of parallel sessions. This series is organised by UKOLN to support members of institutional Web management teams within the UK academic communities.
`http://www.ukoln.ac.uk/web-focus/events/workshops/webmaster-2006/`

HotAC1

16th June 2006: Dublin

The First Workshop on Hot Topics in Autonomic Computing – conquering the growing complexity of large-scale systems. (Sponsored by IEEE Computer Society and USENIX).
`http://www.aqualab.cs.northwestern.edu/HotACI/`

GUADEC 2006: The GNOME Conference

24th to 30th June 2006: Vilanova i la Geltrú, Catalonia, Spain

The 7th annual GNOME User and Developer European Conference (GUADEC) will bring developers, GNOME Foundation leaders, individuals, businesses and governments, as well as Free Software and Open Source software users together in Vilanova i la Geltrú (Catalonia – Spain).
`http://guadec.org/GUADEC2006`

Linux 2006

29th June to 2nd July 2006: Brighton

The annual UKUUG Linux Technical Conference.
`http://www.ukuug.org/events/linux2006/`

Exim Course 2006

18th July to 21th July 2006: Cambridge

Exim is a mail transfer agent (MTA) developed by Dr Philip Hazel at the University of Cam-

bridge for use on Unix systems connected to the Internet. It runs on most versions of Unix and is freely available under the terms of the GNU General Public Licence. Exim is in production use at many sites around the world, including some large ISPs moving hundreds of thousands of messages per day. Version 4 was released in February 2002. This course is aimed both at those mail administrators who are already using Exim and also those who may be thinking about it. The course is based on Exim 4, the current release of Exim. A general knowledge of how Internet mail works will be assumed, but a one-hour "warm-up" introduction will be given before lunch on the first day for those who feel the need for it.
`http://www-tus.csx.cam.ac.uk/courses/exim/`

LUGRadio Live 2006

22nd and 23rd July 2006: Wolverhampton

LUGRadio Live is an annual event driven by, and for the Open Source community. The event includes a range of speakers, exhibitors and other attractions, all housed within a unique event with a unique atmosphere. Last year's event in June 2005 was a huge success, and this year LUGRadio Live 2006 will be nothing you have seen before.
`http://www.lugradio.org/live/2006/index.php/Main_Page`

YAPC Europe 2006

30th August to 1st September 2006: Birmingham

"The Accessibility of Perl" is hoped to inspire the submission of talks and tutorials in many areas that can be covered by accessibility. There are many areas of disability that Perl is involved with, but the idea of accessibility can be taken further. Larry Wall is often quoted as saying Perl makes the hard things easy, and the impossible possible. What project are you working on, that is making life easier or a dream a reality?
`http://www.birmingham2006.com/`

EuroOSCON 2006
18th to 21st September 2006: Brussels, Belgium
The O'Reilly European Open Source Convention is where coders, sys admins, entrepreneurs, and business people working in free and open source software gather to share ideas, discover code, and find solutions. At EuroOSCON 2005, nearly 500 delegates took part in sessions and tutorials across eleven technology tracks, learning about the newest features and versions from creators and experts. We anticipate that EuroOSCON 2006 will be even more successful – the place for the open source community to meet up, debate, make deals, and connect face to face with other open source enthusiasts from across the continent and around the world.
`http://conferences.oreillynet.com/euos2006/`

UKUUG Spring Conference

Dru Lavigne

Firewalling with OpenBSD's PF packet filter: Peter Hansteen

The day started with a choice of workshop. I've been promising for some time to write an article on PF so I was looking forward to this talk to help round out my experience with this firewall. Peter started with an overview of PF's features:

- both the module itself and the administration utilities are part of the kernel for performance reasons; PF is now in the base system for OpenBSD, FreeBSD, NetBSD, and DragonFly BSD

- PF can filter by protocol, port, packet type, address, and operating system

- altq, which can be used for load balancing and traffic shaping has been integrated into PF
- the configuration file is human readable and supports macros and tables to simplify rules
- PF supports NAT and redirection to proxies and spamd
- top bottom rulebase logic where last match wins; can use quick keyword to stop parsing at that match

Peter also mentioned some utilities I hadn't come across yet and which are now on my list of things to check out when I get time:

- ftpsesame: an fTP proxy that will create an anchor rule for you by analyzing FTP connections
- pftpx: the next generation FTP proxy
- pftop: for displaying in real time the state table and pf statistics

Also, the firewall can operate at Layer 2 (i.e. totally invisible to outside world) as the bridging feature can still support packet filtering, NAT and redirection. Keep in mind that you will need serial or local access to the firewall system as it won't be accessible from the network.

Peter has promised to add more material on his site giving examples of using anchors. He gave many useful examples for capabilities which are found in the manpages but haven't quite found their way yet into tutorials. These included adding labels to rules for collecting per-rule statistics, handling unwanted traffic such as worms, prioritizing ACKs to improve transfer speed over asymmetric links, and configuring firewall redundancy.

Then he demonstrated how to turn a wireless driver into a WAP using hostap mode with ifconfig and then adding rules to `/etc/pf.conf`. He also mentioned an excellent online course on wireless security which is well worth checking out.

This was the first time I had heard of `authpf` which can be used to create an authentication gateway. He also demonstrated an efficient rule for filtering dictionary attacks against SSH which has the added benefit of greatly reducing logs. I already use overload but will try replacing flush with flush global.

Peter also gave some rule examples for the three types of alternate queueing with altq:

- cbq: class based or percentage of MB or GB
- priq: priority based
- hfsc: hierarchical

He ended his talk with some haiku. All of the rule examples I mentioned and more can be found at Peter's site.

Keynote – almost

As I was packing Monday morning and going through my checklist, I suddenly remembered that I had been asked to do a short keynote in addition to the BSD Certification talk. I spent the next hour gathering interesting bits on "general system administration" and turning them into a short and visual OpenOffice Impress presentation.

When we arrived at registration this morning, the speakers all went to test their laptops with the projectors. All of the non-IBM laptops talked nicely with the University's projectors. The IBM laptops (mine and Peter's) did not, despite changed CMOS settings, Xorg settings, IBM function keys, powering off/on, unplugging cables and whatever other mojo we could think up. The promised wireless network turned out to only support Windows systems, making it that much harder to simply bop over the presentations to another laptop.

This added up to me not doing my keynote and missing the Google keynote seeing if I could coax interaction with the projector in the other lecture room in time for the Certification talk. Fortunately, the conference proceedings included screenshots of my slides so I ended up doing the talk the old fashioned way.

Talk on BSD Certification

The talk was fairly well attended by the mostly BSD contingent of the Unix conference attendees and I'll upload the slides once I'm back in Canada. It was interesting to find that most of the audience had not heard of the effort until the conference but now that they had they were genuinely interested. I also received some good feedback on resources in Britain, ideas for testing methodologies and answered some questions on psychometrics.

I missed the next talk as I became engaged in conversation regarding BSD Certification. Richard from MOST discussed the state of Open Source in the UK and what his organisation provides. He also introduced me to INGOTS which started by introducing Open Source to elementary school students and is now creating programs to teach Open Source office skills to adults.

I also managed to miss tea and the final talk as there had been a mixup with our room and we had to packup our gear and move it to another part of residence.

Conference Dinner

After the conference, we all headed downhill for the 3 course sitdown dinner at the Swallow Three Tuns Hotel. I had a chance to speaker with Peter and we discussed the need for the publication of a "pf cookbook". At the table, I chatted with Josette Garcia from the O'Reilly booth, Sunit Gopal from the Google booth, a sysadmin from Oxford and another chap who had enjoyed his first holiday to Canada last year.

If nothing else, you eat very well (much too well, to be honest) at conferences. Fortunately, everything in Durham is up a very steep hill (in both directions I'm sure) which hopefully will make up somewhat for the eating.

Large Database Administration with PostgreSQL: Simon Riggs

Simon Riggs of the PostgreSQL Project started with an overview of PostgreSQL's features:

- fast becoming the de facto database for enterprise level open source solutions as it copes well with databases of 100s of GB in size
- full constraint and referential integrity features
- function based and partial indexes
- multiple server side languages and multiple client interfaces
- advanced optimiser, intuitive index use, publicly available performance tests
- used at Oxford University and APLAWS

Followed by what's new in 8.x:

- native Windows port
- SAVEPOINTs and ROLEs

Regarding risk:

- very low threat-to-fix times: typical is a day or two
- Coverity report showed the number of bugs per lines of code which was 10x better than Mysql which was 10x better than Oracle
- PITR allows full transactional recovery
- slony and pgpool can provide high availability
- philosophy is that is is easier to code something that “just works” than to answer the same questions on mailing lists
- one of the largest database communities with thriving mailing lists

Sponsors and Supporters:

- Red Hat, SRA, Fujitsu, Unisys, Sun, Afilias, Sony Online, Pervasive, EnterpriseDB, Greenplum, OSDL

The rest of the talk dealt with specific tuning and design tips for various database scenarios. I’m hoping Simon’s slides eventually make their way online as this was the most useful part of the talk and would make a handy optimization reference.
Coming in 8.2:

- +300% faster on large sort performance
- +30% sequential scan performance
- +39% data loading
- +100% full text search
- further scalability gains for 8 - 16 CPUs
- bitmap indexes
- procedural language debugger
- improved partitioning
- standby server
- index organised tables

How you can help:

- publish case studies
- test the features you want in the next release
- ask for help so we know what people want
- publish performance reports and annoyances

Proactive Wireless Networks with OpenBSD: Reyk Floeter

This was an enlightening talk from a OpenBSD developer who gave good insight into the negotiations and difficulties that happen behind the scenes in order to get the specs needed to program drivers for Open Source operating systems.

Reyk started by discussing the goals of the OpenBSD project: to provide good code with a free license (GPL only as an exception as in gcc), and to focus on portability, correctness and proactive security.

He then discussed traditional wireless support which includes the wi, atw, and an drivers. These typically used a flash-based firmware with active handling of 802.11 by the firmware.

However, 3rd generation wireless products are problematic due to massive complexity of chipset design, many features provided by software, regulation issues in the 5 GHZ range, vendor politics, FCC's push to regulate possible manipulation of SDRs (software Designed Radios) making many vendors unwilling to provide source.

OpenBSD's wireless goals are to open the firmware or get a new and free license; get hardware specs without signing any NDAs. They feel it was a wrong reaction of some other Open Source projects to integrate non-free and binary-only drivers, distributing non-free firmware files and signing NDAs with vendors while still claiming to have open source or free software.

Progress includes new drivers: atu, iwi, ipw, (u)ral, rtw, ar5, ath. Read man iwi for the email address of the person at Intel who refuses to release specs. These drivers are in the queue for OpenBSD 4.0: zd, anw.

Reyk also gave a personal story on a specific vendor's threats for providing an OpenSource driver. He responded that it is legal in Germany to reverse engineer a driver to create your own code to support European devices when vendor does not release specs.

Changes for 3.9:

- hostapd(8)
- implementation of 802.11f
- Inter Access Point Protocol which speeds up roaming between APs
- decentralised wireless solution v.s. CAPWAP/LWAP which is more secure and simple

He then spent some time discussing hostapd which provides event rules using a well designed configuration language; hostapd.conf has a similar syntax to pf.conf. It provides rogue accesspoint detection by creating a table defining the MAC addresses of your APs. It provides protection against wireless DoS attacks by using rate keyword (similar to pf's max-conn). It is also integrated with Prelude, a hybrid IDS framework; their LML sensor supports hostapd.

Still todo: 802.11 fingerprinting and 802.3ad LACP support

Reyk then demonstrated how trunk(4) provides failover between a wireless and wired interfaces. He started playing an mp3 (using a command line player without a receive buffer) over the network he had attached to through his Ethernet NIC. He then unplugged the Ethernet NIC–the music paused for about a second as the bridge learned the MAC address of the wireless NIC and then the music resumed. To configure, simply:

```
ifconfig trunk0 trunkproto failover trunkport em0 trunkport ath0
dhclient trunk0
```

OpenSSH 4.3 has new VPN tunneling which can create Layer 2/3 tunnels over SSH2 without the need for additional software. It supports ad-hoc VPN tunnels (check for tunnel in ssh_config).

OpenBSD now provides an improved ipsec.conf: can setup vpn in 2 config lines using flow keyword.

He has started working on WPA2 as heavily requested; it will be a clean and simple implementation from scratch.

Automating Xen Virtual Machine Deployments: Kris Buytaert

In this talk, Kris discussed why he integrated SystemImager with Xen to make a customised and automated imaging system that just worked, regardless of the distro and the package management system. As he discussed the difficulties of moving from distro-specific solutions to a generic solution I was reminded that such gymnastics simply aren't an issue in BSD. His mention of using a cvs system to store the images is a good idea and a mental note stowed away for the next time I'm in a scenario where I need to manage a large amount of system images.

He also mentioned 2 interesting sites which are worth checking out:

- Infrastructures.org
- SISuite

eXtreme Programming, FreeBSD a Case Study: Paul Richards

I've written other blog entries interviewing developers who use the Agile development process. Not being a developer myself, I found Paul's talk a very informative overview of how Agile differs from traditional development processes. Here are the points I recorded regarding the Agile philosophy:

- Agile development project methodology is designed to deliver on time and within budget while accepting that requirements change
- "user stories" are not techinical specs but user descriptions of a piece of deliverable functionality; they facilitate time estimating in "ideal" time
- in Agile roles, developers estimate effort required to do the work and customers determine the priorities based on business requirements
- developers choose their own tasks and time estimates for each iteration or mini-release
- "project velocity" determines how much can be done with each iteration and is calculated using the estimates from the previous iteration and the actual work completed; it requires constant interaction with customer to see if scope, cost or time estimates need to be changed
- "task based" philosophy moves people around which prevents islands of knowledge and keeps developers fresh and interested; a daily standup meeting in dev room at whiteboard keeps everyone on track; mutual respect means no leaders in XP (eXtreme Programming) team
- "system metaphor" helps to communicate ideas and guides naming conventions (e.g. filesystem layout is like folders in filing cabinet)
- CRC (Class, Responsibility, Collaboration) similar to UML class diagrams; physical paper is satisfying to complete in conjunction with online Planner
- "spike solutions" are used when you don't know how long it will take to implement an idea; it's not an initial implementation or a prototype; aim is to improve iteration estimates
- delivering small solutions regularly helps to deal with customer's changing their minds; results in simpler and less costly solutions

- if project gets canned, you still have a working something as each iteration produces a piece of working functionality
- before any programming on the project itself starts, the unit test is written first to define functionality; expand unit tests as problems are found; re-factoring mercilessly is safe
- pair programming is good as process of talking about a problem often reveals solution; assumes team bonding
- integrate early, often, sequentially (commit work as it is completed); always develop against the current version
- coding standards prevent chaos
- run unit tests against each commit so know which commit has problem
- worry about optimisation when the project is finished and actually works
- no overtime allowed!

Paul wrapped up the talk nicely comparing the FreeBSD project to Agile philosophy. While some programming practices are different due to the global and mostly volunteer nature of the project, one can still see that the best practices for commercial software projects which XP formalises draw upon the practices used by the FreeBSD project. (I would go further and say each of the BSD projects and, earlier, the CSRG environment at Berkeley)

Some features mentioned that were unique to FreeBSD:

- 5.0 was a classic example of being scope based rather than time based
- the “Danish Axe” is useful for removing unused features from the src tree
- there is no official unit testing but make world before a release and those who run current and report bugs provide a close approximation

Robert Watson commented that Coverity now provides tools for unit testing. The FreeBSD Project has automated this to run every 24 hours and to input its findings into the GNATS database.

Security Through Obscurity, A Review of FreeBSD’s Lesser Known Security Capabilities: David Malone

I happened to have breakfast with David, a FreeBSD developer and sysadmin from Ireland. Even though his talk didn’t mention anything new to me (you’ll find several how-tos for many of these in the Security chapter of BSD Hacks), I still found his talking style engaging. I’ve included notes for those that are new to FreeBSD’s security features:

Older features include file flags and securelevels (designed to limit damage of malicious root). Newer features include MAC, seeotheruid, BSDextended, portacl and GEOM/GBDE.

David did a quick review of file flags (chflags, ls -lo).

To stop hardlink tricks with flags, take a look at sysctl -l security.bsd.hardlink_check_uid; note that this may not work over NFS.

He then provided an overview of the MAC framework which asks module(s) if an operation is permitted. It can implement Biba, MLS, SELinux, as well as more simple policies. Some modules are loadable and some are not; modules which use labels aren’t loadable. The framework provides for over 60 MAC checks.

He demonstrated `mac_seeotheruids`.

Then he showed a configuration for `mac_portacl` which allows you to create ACLs on which users can bind to what port. Note that an allow won't override other kernel restrictions. In his example, he gave the syntax to give the www user access to ports 80 and 443. This allows you to start apache as www (not root); make sure all log files are writable by www.

BSDextended allows more complex rules than those provided by ugo permissions. The rules you create are global like a firewall ACL not individual like a file ACL. In other words, despite file permissions you can restrict users from accessing files. This can be used for sandboxing. Assuming MAC support is in the kernel: kldload mac_bsdextended, then use ugidfwto create the rules. David has added extensions (to be committed soon) to ugidfw.

David then gave an example of setting up an encrypted disk using gbde.

Adding `gbde_devices="AUTO"` in `/etc/rc.conf` will mount filesystem at bootup and prompt you for passphrase. If you set `gbde_swap_enable="YES"` for an automatic encrypted swap partition, it will generate an automatic encrypted random key at bootup which does not require a passphrase.

Keynote – finally

With some work we managed to scp my keynote slides onto a FreeBSD server on the network. I used OpenOffice2 to first convert the Impress format into a PDF as the PC system connected to the projector was running Windows.

I started the afternoon lightning talks with the 15 minute keynote on "general system administration". I'll give the URL to the PDF once I have a chance to upload it. I also need to work on my timing for the next time I try to deliver a humourous keynote.

Certified Open – The Provenance for Skills and Deliverables

Alan wasn't able to make it to Durham for the talk so Basil Cousins from OpenForum Europe gave this lighting talk on behalf of the Certified Open project.

During my research for BSD Certification, I have come across many of the Open Source and training initiatives available in the UK. I also had a chance to listen to members from OpenForum Europe at last year's OSCON and am actively involved with GOSLING in Canada. For this and many more reasons, I wish this had been a full blown talk (or even a half day workshop) rather than just a 20 minute overview.

Some of the points Basil raised:

- OpenForum Europe is now 4 years old
- difficulties are not technical but legal and marketing
- key to success is to raise perception of OSS in the eyes of the users and the Members of Parliament; we need to add value to the services we provide
- the key message of Open Certified is to define the dangers of lockin and to avoid it for full interoperability by publishing interoperability criteria
- it will provide a self-assesment system for certifying products, services, skills, and business practices
- originated from the Open Source Academy
- March 24 (the day of the talk) was when Certified Open was officially launched
- self-assessment with governance to support the criteria
- levels of openness established (gold, silver, bronze)
- simple to administer and use

- comprehensive appeals process and early warning system
- trademarked process run by non-profit Certified Open

Value propositions for the initiative:

- hiring managers receive endorsement of experience of job applicants
- they are looking for SMEs (Subject Matter Experts)
- will provide a SkillsTracker Process (RedHat helped with design of database)
- certifications are part of obtaining a skills record
- will become a driver for Open Source training and certification
- procurement for evaluation tenders
- current goal is to get as many projects self-assessed and on the website within the next few months in preparation for the next stage of the project
- he also encouraged everyone to download and read the current publications

Trading Connectivity and Convention for Address Space: Andrew Macpherson

This quick 10 minute talk was targetted for companies with small address blocks as it won't be of use to single-address broadband users or dynamic IP users.

Basically, RFC 1219 defines reverse binary counting for allocating address space while reducing waste. Unfortunately, it is often not used by those who provision address space.

Consider that if you are assigned 8 addresses, 3 are unusable (broadcast, subnet, gateway). He then explained a trick on how to gain those addresses back at the cost of just losing access to other datacenter customers (which you don't need anyway). I didn't quite grasp the math and it looks like the slides haven't made it online yet – either that or I mis-spelled the URL in my notes.

nmon - Nigel's Performance Monitor: Nigel Griffiths

Nigel definitely was the most enthusiastic speaker at the conference and looked like someone who really enjoys his job and hacking solutions to solve problems.

He demonstrated `nmon` which is a performance monitor for AIX and Linux.

From the same site, you can also download `nmon2rrd`, Stephen Atkins' automated utility for sending graphs of capturing data to Excel.

Once the talks were over, Ray Miller of the UKUUG gave some quick closing notes thanking the speakers and organisers of the conference. Then most of the attendees started the uphill trek to the residence to pack their things and head back home on the next train.

Evening Talk on GPL version 3 given by Georg Greve

Sam Smith

On May 29th, UKUUG was proud to present an evening talk in Manchester by the President of FSF Europe, Georg Greve, talking about GPL version 3.

The presentation itself was non-legal, giving a deep insight into the what and why of GPL version 3, without going into any of the legalese of the language itself. It covered the policies and thinking underpinning discussions, and the aims that the changes were intending to achieve. It's important to note that, at the moment, there is no such thing as GPL v3. The license itself does not yet exist, there is only a single draft version (at time of writing). Through 2006, there will be at least one more draft put to the community for review, as well as meetings across the world before the new license is released in 2007. As a result, FSF does not advise anyone to use the draft license, but is only soliciting as many comments and opinions as it can get to make GPL v3 last as long as the current version. There are comment forms and discussions ongoing, and the FSF is extremely keen on getting people's input on the license, and its changes, rather than using it at the moment. The thoughts that one person has that no one else has yet thought of could be vital, and this process is seen as key to making GPL v3 last as long as GPL v2 has.

Georg started out by noting that GPL v2 has been a huge success. Written 15 years ago, it has survived a huge number of changes in the environment in which it operates, and a huge expansion in the software it covers. Many of the changes being made for v3 are small clarifications, found during the last 15 years. Those changes on their own would not justify a new license version and all the associated upheaval, but since that work is going on for other reasons, it makes sense to make minor internationalisation and clarifications changes to make everyone's life easier. There are also technical advances (such as that really innovation, dynamic linking) which are unclear since v2 predates their adoption.

There are, of course, a few very large changes, and these are what Georg focused on for the majority of his talk, and what the majority of the questions at the end covered. In short, the three changes are Digital Rights Management (DRM), Software Patents and License Compatibility. There are also questions, still under discussion, about Termination of the license, so this wasn't covered other than mentioning it was still under discussion. That said, the central principal of the update has been "Change as little as possible".

DRM, in the view of the FSF, has three central problems; the loss of control of your computer (ie Power), treating the user as the enemy (Security), and moving costs after purchase (Financial). However, despite the issues, v3 does not address DRM directly. Unlimited use of software for any purposes is a central FSF principal, and that use can include writing DRM software. You can build DRM enforcement software using GPL v3. However, it requires that there is the means for users to exercise their rights; and gives others permission to write interoperable software. This means, that while you may write DRM into GPL software, you can not prevent people from working around it (and distributing those work-arounds) should they have the skill to do so.

GPL v2 has a "Liberty or Death" clause (aka the "Truth in Labelling" clause, and the main change between GPL v1 and GPL v2), says 'if somebody uses a patent or something else to effectively make a program non-free then it cannot be distributed at all'.
`http://fsfeurope.org/projects/gplv3/fisl-rms-transcript.en.html#liberty-or-death`
– ie if a software patent can be used against GPL v2 software, the software can not be distributed at all. In GPL v3, there is an explicitly patent grant to users of software that they can use (the software and therefore) the patents royalty free, worldwide. There are also protections prevents collusion between 2 patent holders. Company A knowingly licenses patent to Company B, which then releases software under GPL v3; Company A can not then sue any users of the software. This explicitly protects downstream packagers and gives them equal rights.

When GPL version 2 was released, there were no other licenses similar to it. That has now changed with a huge number of GPL alike licenses. GPL v3 attempts to do compatibility by saying what is and is not permitted, rather than by saying which licenses it matches against. It also allows for additional requirements to be added to the base v3 license, but says what those requirements may or may not be. You can vary the license in the areas of attribution and copyright, liability, and publicity limitations. These address many of the requirements of other licenses, without affecting the core license itself. The biggest example of this is the Affero GPL which requires that if your software is run over the web, you need to make a copy of it over the web as well. There are also options to limit the use of software patents with licensed code, but it explicitly does not allow patent aggression.

In whole, the license is more modular, allowing different bits to be slotted in where necessary. Many of the changes have been to make companies happier using GPL , in areas which does not compromise the users' freedom. But the FSF is extremely keen in hearing from everyone, and accepting all comments to find out things that the user base thinks of that no one else has, looking at it from a new and different viewpoint.

Answering questions from the audience, there will be a LGPL draft based around the changes in GPL v3, but only once the main process is nearer completion. Big software companies and users are generally supportive. While there have been some high profile misunderstandings of the new license and its intent, Georg was keen to point out that clarity was increased, and simplicity enhanced, when those who thought they understood the new license were mistaken in the press. While there are many disagreements between the many licenses of the open source world, there is far more that unites them than divides them, and everyone is welcome to pick their own license.

GPL v2 and v3 compatibility is likely to be an issue. Some software is licensed under GPL v2 and can not therefore be moved forward to v3. The FSF approach is that there is nothing that can be done about that, and they advised developers against taking the words "or later" out of GPL v2 in the first place. There are also going to be issues around code which adds the optional clauses to the license adding more restrictions into the license.

In summary, Georg presented a very informative, extremely clear talk on GPL v3 and why they are making the changes that are being made. Without any legalese, the reasoning was approachable and clear (hopefully some of that will have transferred, through my notes, into my words above). While we would organise an event talking about the updates to the BSD license, it is unlikely that there will be any for a long time. Hopefully, the updates to GPL v3 will mean that there are no new versions of that needed for a similar period.

EU Commission proposes to criminalise European software
FFII Press Release

Brussels, 12 May 2005

The Commission's recently relaunched "Enforcement Directive" (IPRED2, 2005/0127 (COD)) proposal aims to criminalise all intentional and commercial IP infringements in order to "combat organised crime" and to "protect national economies and governments". This however results in the Commission exceeding its competence and is criminalising many EU businesses with unjustified and ill-conceived measures.

A company may infringe on a patent if it thinks the patent would not stand up in court. This is common business practice, in particular in the software industry where most patents are granted on insufficient legal grounds. And while Commission is seeking to criminalise this practice, the US is reconsidering its "treble damages" policy in such cases precisely because of widespread abuse.

Jonas Maebe, FFII board member, comments: "Does the Commission really intend to criminalise Europe's entire software industry? Can it name even one computer program which does not infringe on a single patent granted by the European Patent Office? It seems they want to replace the Lisbon goals with an Alcatraz program."

"The EU-Commission proposed means which divert law enforcement resources and which are not well suited to combat organised crime" adds André Rebentisch, FFII WIPO representative. "Appropriate definitions for counterfeiting and copyright piracy are already available in other EU regulations, but here the Commission prefers rather vague terminology which puts our knowledge economy at risk.”

Ante Wessels, FFII analyst, notes: “In only 10 of the EU’s 25 member states patent infringement is a crime today. Does this lead to distortion in trade, does it give the countries in which it is not a crime a competitive advantage? Nobody has ever claimed such a thing. Therefore there is no legal ground for including patent infringement in this directive. There are 10 more IP rights for which this question has to be answered.”

Pieter Hintjens, FFII President, concludes: “We’re very concerned when we see IP enforcement being idolized like this, regardless of the consequences. There is a huge and vital debate about whether we need patents at all in the software industry. This law ignores that debate and seeks to enforce those patents, labeling businessmen as common criminals, terrorists, or mafiosi.”

A full analysis is available at:
`http://wiki.ffii.org/Ipred2060510En`

Background information

Patent infringements currently constitute a criminal offence in 10 of the 25 member states. In the Netherlands the government previously already proposed to take patent infringements out of criminal law, exactly because in practice criminal provisions are generally unsuited and unused for handling such issues.

The proposal stresses that law enforcement bodies should start investigations at their own initiative, i.e. without a complaint from right holders. Law enforcement officials however are often unaware, and rightly so, about private or even public licensing agreements. See e.g. a UK Trading Standards official having a hard time believing that companies can legally resell the freely distributable Firefox web browser
`http://business.timesonline.co.uk/article/0,,9075-2051196,00.html`

Apart from patents, many other rights are subsumed under “IP” where the line between infringement and non-infringement is very blurry. See e.g. the Da Vinci Code case (copyright), or Microsoft vs MikeRoweSoft (trademarks). Criminal law however requires very clear boundaries. Not being able to know beforehand whether one commits a criminal offence or not is unacceptable both morally and in terms of justice and human rights.

In case of infringement, the right holder is usually interested in compensation (civil law), not punishment (criminal law). Criminal law must be reserved for criminals, otherwise it risks to lose all authority, effectiveness and respect.

Criminal law enforcement is paid for by the public. As Dutch Minister of Justice Donner said: “[Commissioner] Frattini mentioned counterfeiting a Ferrari, but isn’t that Ferrari’s business?”. The directive also received a lot of attention in the Netherlands because this is the first time Brussels interferes with criminal measures without member states having a veto. For more information, see:
`http://wiki.ffii.org/IpredDonner060428En`

Both the Dutch Minister of Justice, and Professor in Law Reto M. Hilty (Max Planck Institute for IP) have noted that the only ground for this directive proposal can be that it solves a distortion in trade between member states. There are no known indications that this indeed is the case. For more comments by Professor Hilty on this directive, see:
`http://www.ipred.org/Hilty`

Links

Full analysis of the text: “IPRED2: European Community goes criminal”
`http://wiki.ffii.org/Ipred2060510En`

EDRI/FIPR take on the new proposal:
`http://www.edri.org/edrigram/number4.9/ipcriminal`

Directive text:
http://register.consilium.europa.eu/pdf/en/06/st08/st08866.en06.pdf

The FFII is a not-for-profit association registered in twenty European countries, dedicated to the development of information goods for the public benefit, based on copyright, free competition, open standards. More than 850 members, 3,500 companies and 100,000 supporters have entrusted the FFII to act as their voice in public policy questions concerning exclusion rights (intellectual property) in data processing.

FSF Press Release: Protesters Provide a Nasty “Vista” for Gates

Free Software Foundation

As Microsoft developers gathered in Seattle to hear Bill Gates’s keynote speech on the future of Microsoft and the coming release of its updated operating system Vista, protesters wearing bright yellow Hazmat suits swarmed the entrance of the city’s convention centre, delivering an unsettling message to the corporation: your product is defective and hazardous to users.

The surprise protest marked the launch of DefectiveByDesign.org, a direct-action campaign that will target Big Media and corporations peddling Digital Restrictions Management (DRM). “Flash protests, direct actions, and practical ways that people can get involved and help stop the stupidity of DRM,” is how campaign manager Gregory Heller described the grassroots effort.

An initiative of the Free Software Foundation (FSF), Defective By Design is urging all technologists to get involved at the start of the campaign. “Technologists are very aware of the dangers of DRM,” said Peter Brown, Executive Director of the FSF. “We see this as the tip of the iceberg and it is our duty to do something about it.” The tech community is uniquely qualified to lead this effort, in Brown’s view. “We know about the collusion of Big Media, device manufacturers and proprietary software companies to lock us down,” he continued. “Their aim is to put Digital Restrictions Management (DRM) into all our computers and homes”.

Brown’s case is simple: the computers, high-definition screens, phones, music players and video players that are currently being sold are “defective by design”. These products don’t respect the user’s right to make private copies of their digital media. These devices make no provision that would allow art, literature, music or film to ever fall into the public domain. Effectively, the media purchased for these devices does not belong to the user – rather, the networking of these DRM’d devices means that as the user watches a film, reads an e-book or switches channels on their HDTV, their habits can be recorded and actions monitored. The result is that over time, DRM technology will negate, if not completely eliminate, the rights of the individual.

“In any other industry, such limitations or invasions would be considered major flaws. A media player that restricts what you can play is like a car that you won’t let you steer,” said Brown. “Products containing DRM are defective – only, unlike other products, these defects are deliberately created by an industry that has long stopped caring about us.”

With DRM in place, media conglomerates can change the rules whenever they want, leading to more restrictions on the individual.

“Media bosses scream ‘pirate’ equating sharing with murder and kidnap, then sue our college students. They then steal our rights and impose crippled products upon us,” said Henri Poole, Chairman of CivicActions and a coalition partner in the campaign. “Media bosses have long been the ‘gatekeepers to the market’ for artists. Now they are threatened by new distribution methods that give artists new freedoms and direct access to an audience. DRM is the media bosses attempt to re-impose their rule”.

Today's event is the first in a series planned by DefectiveByDesign.org that will mobilise individuals to make a stand against DRM.

About Defective By Design

DefectiveByDesign.org is a broad-based, anti-DRM campaign that is targeting Big Media, unhelpful manufacturers and DRM distributors. It aims to make all manufacturers wary about bringing their DRM-enabled products to market. The campaign aims to identify "defective" products for the consumer. Users are being asked to stand up in defence of their existing freedoms and to take action by joining at http://DefectiveByDesign.org

About the Free Software Foundation

The Free Software Foundation, founded in 1985, is dedicated to promoting computer users' right to use, study, copy, modify, and redistribute computer programs. The FSF promotes the development and use of free (as in freedom) software - particularly the GNU operating system and its GNU/Linux variants - and free documentation for free software. The FSF also helps to spread awareness of the ethical and political issues of freedom in the use of software.

For further information, see:
`http://www.fsf.org`

System Administrator Appreciation Day

Ray Miller

Friday July 28th 2006 sees the 7th annual System Administrator Appreciation Day - the one day of the year when computer users across the globe stop to think of the people who keep their computers and networks running. This year, UKUUG would like to do something fun to add to the celebrations.

We're offering a copy of "Even Grues Get Full" (The fourth User Friendly collection) to the member who comes up with the best idea for our contribution. This could be a poster design, a slogan for a T-shirt or mug, or an idea for some sort of sysadmin get-together. Use your imagination! Please send us suggestions to reach us no later than 7th July 2006, and keep your eyes peeled for something from UKUUG when the day comes around. Send your suggestions to:
`ukuug@ukuug.org`

For more information about this and previous years' System Administrator Appreciation Days, see
`http://www.sysadminday.com/`

BSD in a Panic

Michael Lucas

My employer's main business is designing Web applications, but once those applications are built our clients turn around and ask "Where should we host this?".

That's where I come in, building and running a small but professional-grade data centre for custom applications. As with any new business, our hosting operation had to make the most of the resources we had. Our resources were strictly limited to cast-off hardware from the web developers and free software. The only major expense was a big-name commercial firewall, purchased for marketing reasons rather than technical ones. With FreeBSD and a whole mess of open-source software, we built a reliable network management system that provides the clients with a great deal of insight into their equipment.

The clients, of course, pay for their own hardware and so have fancy high-end rackmount servers with their chosen applications, platforms, and operating systems. We've since upgraded the hardware – warranties are nice, after all! – but have seen no need to change the software. One day, a customer that had expected to use very little bandwidth found that they had enough requests coming in to use close to twice the bandwidth we had for the entire data centre. This affected every customer, slowing the entire hosting environment to speeds comparable to a snail in molasses. If your $9.95/month web page is slow you have little to complain about, but if your $50,000/month Web application is slow you pick up the phone and scream until it stops. To make matters worse, my grandmother had died only a couple days before.

Visitation was on Tuesday, and the funeral was Wednesday morning. Monday morning I handed the problem to a minion and said "Here, do something about this." I knew bandwidth could be managed at many points: the Web servers themselves, the load balancer in front of them, the commercial firewall, or even the router. Tuesday after the visitation I found my cellphone full of messages. Internet Information Server can manage bandwidth – in eight megabyte increments and only if the content is static HTML and JPEG filles. With several Web servers behind the load balancer, that fell somewhere between useless and laughable. The load balancer did support traffic shaping, if we bought the new feature set. If we plopped down a credit card number, we could have it installed by next Sunday. Our big-name commercial firewall also had traffic shaping features available, if we upgraded our service level and paid an additional (and quite hefty) fee for the feature set. That left the router, which I had previously investigated and found would support traffic shaping with only a flash upgrade.

I was on the phone until midnight Tuesday night, making arrangements to do an emergency OS upgrade on the router on Wednesday night. I had planned to go to the funeral in the morning, give the eulogy, go home and take a nap, and arrive at work at midnight ready to rock. The funeral turned out to be more dramatic than I had expected and I showed up at work at midnight sleepless, bleary-eyed, and upright only courtesy of the twin blessings of caffeine and adrenaline. In my email, I found a note that several big clients had threatened to leave unless the problem were resolved Thursday morning. If I hadn't already been stressed out, the prospect of choosing a friend to lay off would have done the trick.

Still, only a simple router flash upgrade and some basic configuration stood between me and relief. What could possibly go wrong? The upgrade went smoothly, but the router behaved oddly when I enabled traffic shaping. Over the next few hours, I discovered that the router didn't have enough memory to simultaneously support all of our BGP feeds and the traffic shaping functionality. Worse, this router wouldn't accept more memory. At about six in the morning, I got an admission from the router vendor that they could not help me. I hung up the phone.

The first client who had threatened departure would be checking in at seven thirty AM. I had slept four hours of the last forty-eight, and had spent most of that time under a fiendish level of emotional stress. I had already emptied my stash of quarters for the soda machine, and had been forced to pillage a co-worker's desk for his. The caffeine and adrenaline that had gotten me to the office had long since worn off, and further doses of each merely slowed my collapse. We had support contracts on every piece of equipment and they were all useless. All the hours of work I had put in, and my team before me, left me with a sum total of absolutely nothing. I made myself sit still for two minutes simply focusing on breathing, making my head stop sliding around loose on my shoulders, and ignoring the loud ticking of the server room clock.

What could be done in ninety minutes – no, now only eighty-eight? I really had one only option. If it didn't work, I would be choosing someone to lay off or filing for unemployment myself. 6:05 AM. I slammed the floppy disk into the drive and started downloading the OpenBSD install floppy then grabbed a spare desktop machine, selecting it from amongst many similar machines by virtue of it being on top of the pile. The next few minutes I alternated between hitting the few required installation commands and dismantling every unused machine unlucky enough to be in reach to find two decent network cards. By 6:33 AM I had two Intel EtherExpress cards in my hands and a new OpenBSD 3.5-snapshot system. I logged in long enough to shut the system down so I could wrench the case off, slam the cards into place, and boot again. OpenBSD's builtin PF packet filter includes all sorts of nifty filtering abilities, all of which I ignored in favour of the traffic-shaping functions. By 6:37 AM I was wheeling a cart with a monitor, keyboard, and my new traffic shaper over to the rack.

Here, the killer problems manifested. I didn't have a spare switch that could handle our Internet bandwidth. The router rack was jammed full, leaving me no place to put the new shaper. I lost almost half an hour finding a crossover cable, and when I discovered one it was only two feet long. The router, of course, was at the top of the rack. Fortunately, if I put the desktop PC on end and left it sitting on the cart, the cable just reached the router. I discovered this about 7:10 AM. I stacked everything so it would reach and began re-wiring the network and reconfiguring subnets. I vaguely recall my manager coming in about 7:15 AM, asking with taut calmness if he could help. If I remember correctly, as I typed madly at the router console I said "Yes. Go away".

At 7:28 AM we had an OpenBSD traffic shaper between the hosting area and our router. All the client applications were reachable from the Internet. I collapsed in my chair and stared blankly at the wall. While everything seemed to work, the proof would be in what happened as our offending site started its daily business. I watched with growing tension as that client's network traffic climbed towards the red line that indicated trouble. The traffic grew to just short of the danger line – and flatlined. Other clients called, happy that their service was restored to its usual quality. (One complained that his site was still slow, but it turned out that bandwidth problems had masked a problem with his application.) The offending client complained that their web site was even slower than before, to which we offered to purchase more bandwidth if they'd agree to buy it.

Today, I have two new routers and new DS3s. The racks are clean again, without extra cables from thrown-together solutions. The desktop machine has been replaced by two OpenBSD boxes in a live-failover configuration, providing protection for our big-name commercial firewall as well as shaping traffic. My thrown-together OpenBSD desktop machine is sitting in the corner of the hardware room. The sign on it says "DO NOT TOUCH: EMERGENCY USE ONLY". Should the clock tick down on some other problem, well, at least I won't have to spend the thirty minutes it took to install.

Introducing the Template Toolkit Part 3

Dave Cross

Using Templates from Perl

Over the last couple of issues we have been looking at using the Template toolkit from the command line using the utilities "tpage" and "ttree". None of the examples that we have looked at have involved us writing a single line of of Perl code.

That's fine for simple projects, but as things get more complex it makes sense to do a lot of the heavy lifting in something a bit more powerful than the Template Toolkit's presentation language. The TT language was never designed to be a general purpose programming language. It's a specialised language for controlling the presentation of data.

As the Template Toolkit is written in Perl, it's easiest to use it from within Perl programs. So that's what we will use in this article.

Having decided that we are going to split the processing between a Perl program and a template the next thing we need to do is to decide exactly where to make this division. In my experience this is usually a pretty simple decision to make. Most programs fall quite neatly into one section that gathers all of the required data and another which presents the data to the user.

If you're having trouble deciding where to make this split then it's often useful to consider an alternative display medium for your data. For example, if you're building a plain text file consider what you would need to change if you were to build an HTML page containing the same data. The data is exactly the same, it's just the presentation that has changed. So the bits of processing that need to change are the bits that should be in the template.

Using the Template Toolkit from Perl

The main interface to the Template Toolkit from a Perl program is the Template module. Template is an object oriented module, but don't let that scare you. It's really very simple to use.

Like all Perl modules, you load the Template module into your program with the `use` statement like this.

```
use Template;
```

You then need to create a Template processor object using the "new" method. This can be as simple as this

```
my $tt = Template->new;
```

But there is also an optional parameter to `new`. We'll look at that a bit later on.

To use the Template processor object, we call the `process` method, passing it the name of a template to process.

```
$tt->process('template.tt') or die $tt->error;
```

The template processor looks for the template file in the current directory (we'll see how to change that later) and processes it in exactly the same way as `tpage` or `ttree` would. The results of processing the template are written to `STDOUT` (but we'll see how to change that very soon).

Notice that if there is any problem processing the template then `process` returns a false value. We can check for that and use the `error` method to produce a suitable error message as we terminate the program.

Passing Variables to the Template

Of course most templates need some kind of input in order to do anything useful. With `tpage` and `ttree` we used the `--define var=value` options to pass variables into the template. There must be a way to do something similar from Perl.

And, of course, there is.

The `process` method takes an optional second parameter which defines the variables that the template will use. This parameter is a reference to hash. The keys of the hash are the names of the variables and the values are the associated data.

You can therefore define variables like this:

```
my %vars = (name  => 'Dave Cross',
email => 'dave@example.com');

$tt->process('template.tt', \%vars) or die $tt->error;
```

This code defines two variables called `name` and `email` which can be referenced within the template. You don't have to stop at scalar values like the ones seen here. You can build any kind of complex data structure.

```
my %vars = (invoice => {
number => '101',
date   => '1st April 2004',
client => 'Example Inc.',
addr   => ’1000 Example Road, Exampleton’,
lines  => [
{
desc  => ’Reversing polarity’,
price => ’1000’
},
{
desc  => ’Regeneration care’,
price => ’2000’
}
]
});
```

This example shows a complex, multi-levelled data structure that models an invoice. The `invoice` variable is a hash and its parts can be accessed within a template as, for example, `invoice.number` and `invoice.date`. The value for `invoice.lines` is an array, so you can access the individual items as, for example, `invoice.lines.0.desc` or you could use it in a `FOREACH` loop. We’ll see more of this example later, but if you want more information about using complex data structures in Perl, see the `perlreftut` and `perldsc` manual pages.

Most Perl programs that use the Template Toolkit will spend a large part of their time building an appropriate data structure to pass to `process`.

Controlling Output

As I mentioned previously, `process` sends its results to `STDOUT` by default. You can change by using its optional third parameter. This can take a number of different types of value. The most common of them is a string which is assumed to be the name of a file. The output from the template is written to this file.

Another option is to pass a reference to a scalar variable. In this case. the output from the template is put into that variable. This is useful if you want to post-process the output in some way.

There are a few other more esoteric alternatives. For details of these see the documentation that comes with the Template Toolkit.

More Options

Last month we saw some other options that `ttree` uses to control exactly how the template is processed. We can do the same thing with the Template module. In fact this method gives us even more options. The processing options are set up when you create a template processor object with `new`. The `new` method take an optional argument which is a reference to a hash of options.

```
my %options = (INCLUDE_PATH => ’./lib’,
OUTPUT_PATH  => ’./out’,
PRE_PROCESS  => ’header.tt,
POST_PROCESS => ’footer.tt’);

my $tt = Template->new(\%options);
```

In this code we set four options. `INCLUDE_PATH` defines a directory where the template processor will look for any templates. If you want more than one directory then set this option to a reference to an array of directories.

```
INCLUDE_PATH => [ ’.’, ’./lib’, ’/opt/templates’ ]
```

`OUTPUT_PATH` defines the directory where any output files will be written.

`PRE_PROCESS` and `POST_PROCESS` define templates that will always be processed before and after and templates passed to `process`. This can be useful if, as in this example, you want to add a header and footer to every template.

I often use `PRE_PROCESS` to process library templates that contain configuration data. Both of these values can also be set to an array reference if you want to pre- or post-process multiple templates.

```
PRE_PROCESS => [ ’config.tt’, ’header.tt’ ]
```

Creating Invoices

For this month’s example we’ll look at creating invoices. Assume that we have details of invoices in a database and that we have a Perl module called `Invoice` that gives us access to the invoice data. See the sections below: “The Invoice database” and “Invoice.pm” for more details on how these are set up.

Here’s a simple text template for an invoice

```
INVOICE [% invoice.id | format(’%05d’) %]

Date: [% invoice.invdate %]

To: [% invoice.client.name %]

[% FOREACH addr_line = invoice.client.address.split(’\n’) -%]
[% addr_line %]
[% END -%]

[% FOREACH line = invoice.lines.sort(’line_no’) -%]
[% line.description | format(’%-40s’) %] £[% line.price | format(’%.2f’) %]
[% END %]
[% ’Total:’ | format(’%40s’) %] £[% total | format(’%.2f’) %]
```

This template expects an Invoice object to be passed to it in the variable `invoice` and also the total value of the invoice in the variable `total`. Most of the data that it needs is in the invoice object and it can access this by calling the object’s various methods using the dot notation. Notice that these dots can be strung together to create expressions like `invoice.client.name` to get the name of the client associated with the invoice.

We’ve also made good use of the `format` plugin. We use it to ensure that the invoice number always has five digits and to ensure that the prices all have two decimal points, but we also use it to ensure that the descriptions are all padded to forty characters and therefore the price column lines up correctly. All of these considerations are about the presentation of the data, so they quite rightly belong in the template.

The Invoice Program

Here’s the program that calls the invoice template.

```
#!/usr/bin/perl

use strict;
use warnings;

use Template;
use Invoice;

my $id = shift || die "No invoice number given\n";
my $fmt = shift || ’txt’;

my $inv = Invoice->retrieve($id) || die "Invoice $id not found\n";

my $total;
$total += $_->price foreach $inv->lines;

my $tt = Template->new;

my %vars = ( invoice => $inv, total => $total );
```

```
$tt->process("$fmt.tt", \%vars, "inv$id.$fmt") or die $tt->error;
```

The program starts by loading the `strict` and `warnings` modules - which no Perl program should be without. It then loads the `Template` and `Invoice` modules which we will specifically need for this application.

The program expects two arguments. The first is the number of the invoice to process. This is mandatory and the program dies if it isn't given. The second argument is optional and defines an output format. The default format is `txt`.

The program then retrieves the invoice from the database using a method that `Class::DBI` has created for us in our `Invoice` class. Having successfully retrieved the invoice we calculate the total by adding together the price fields from all of the lines in the invoice. Again, the `lines` method was automatically created by `Class::DBI`.

Then all that's left to do is to create our template processor object and process the template. We use three arguments to the `process` method. The first argument is a string containing the name of the template to use. For the text format we'll use `txt.tt`. The second argument is a hash containing the variables. The third is the name of the output file. For text invoices, the name will be something like `inv1.txt`.

Running this program with some simple test data gives the following output.

```
INVOICE 00001

Date: 2004-05-01

To: The Doctor
The TARDIS
Somewhere in space and time

Reversing polarity                      £1000.00
Regeneration care                       £2000.00

Total: £3000.00
```

But what happens when we decide that we also want to put out invoices on our web site? Or that we want to create RTF versions of invoices that can be edited in OpenOffice.org?

This is where the work of separating the data collection from the presentation really pays off. By just creating a new template, we can create a new view of our data very easily. Here is an example HTML template.

```
<html>
  <head>
    <title>Invoice [% invoice.id | format('%05d') %]</title>
  </head>
  <body>
    <h1>INVOICE [% invoice.id | format('%05d') %]</h1>

    <table>
      <tr>
        <td>Date:</td><td>[% invoice.invdate %]</td>
      </tr>
      <tr><td colspan="2">&nbsp;</td></tr>
      <tr>
        <td valign="top">To:</td>
        <td>[% invoice.client.name;
invoice.client.address.split('\n').join('<br>') -%]
        </td>
      </tr>
    </table>
    <table>

[% FOREACH line = invoice.lines.sort('line_no') -%]
      <tr>
        <td>[% line.description %]</td>
        <td>£[% line.price | format('%.2f') %]</td>
      </tr>
[% END %]
      <tr>
        <td align="right">Total:</td>
        <td>£[% total | format('%.2f') %]</td>
      </tr>
    </table>
  </body>

</html>
```

Basically this does the same things as the text template, but it produces HTML instead of plain text. Instead of worrying about lining up the text columns, we have used HTML tables. Instead of displaying a pound sign, we use the `&pound;` HTML entity. Simply put this template into `html.tt` and our existing script can be used unchanged to create HTML pages. The fact that we find it so easy to create another view of the data means that we must have got the separation of processing and presentation just about right.

The Invoice database

We'll assume that we have a very simple database that stores details of our invoices. The database has three tables which contain data about clients, invoices and invoice lines. Let's look at the tables one at a time.

The Client table

Each of our clients has one row of data in the client table. Each row contains the client's name and address together with a unique identifier for the client. In MySQL, the definition of the table looks like this

```
CREATE TABLE client (
id int(11) NOT NULL,
name varchar(50) default NULL,
address varchar(250) default NULL,
PRIMARY KEY  (id)
);
```

The invoice table

Each invoice that we send will create one new row in the invoice table. It contains the invoice number and date along with the id of the client that the invoice was sent to. The table definition looks like this

```
CREATE TABLE invoice (
id int(11) NOT NULL default '0',
invdate date default NULL,
client int(11) default NULL,
PRIMARY KEY  (id),
INDEX fk_cli (client),
FOREIGN KEY (client) REFERENCES client(id)
);
```

Notice that we have also given the table a foreign key which declares that the `client` column in this table is a reference to the `id` column in the client table.

The line table

Within an invoice we have a number of invoice lines. Each of these represents one of the items that the invoices charges for. An invoice line has a number, a description of the goods or services sold and a price. It also contains the number of invoice that it belongs to.

Here’s the table definition:

```
CREATE TABLE line (
invoice int(11) NOT NULL default ’0’,
line_no int(11) NOT NULL default ’0’,
description varchar(250) default NULL,
price float(10,2) default NULL,
PRIMARY KEY  (invoice,line_no),
INDEX fk_inv (invoice),
FOREIGN KEY (invoice) REFERENCES invoice(id)
);
```

Notice that once more we’ve defined a foreign key linking the line table back to the invoice table.

Invoice.pm

When you are creating a Perl application that talks to a database it’s generally a good idea to isolate all of the database interaction in one place. And usually it makes sense to write a module to handle it all.

Writing Perl modules really isn’t as hard as you might think and there are plenty of tools out there to make it as easy as possible. I have recently started using the module `Class::DBI` for all of my database work. It’s a wrapper around Perl’s standard database interface module (called `DBI`) and it’s a very easy way to create very useful classes built around database tables. Here are the contents of my file `Invoice.pm` which contains all the database code for this application.

```
package Invoice;

use Class::DBI::Loader;
use Class::DBI::Loader::Relationship;

my $loader = Class::DBI::Loader->new(dsn => ’dbi:mysql:invoice:tma2’,
user => ’invoice’,
password => ’inv01ce’);

my @rels = (
’a client has invoices’,
’an invoice has lines’);

$loader->relationship($_) for @rels;

1;
```

That’s about a dozen lines of code and when we load it into our program (with `use Invoice`) we’ll get three new classes `Client`, `Invoice` and `Line` which are object-oriented interfaces to our three tables.

The cleverest thing about this tool is that it also understands the relationships between our tables. This means that if we’ve got an `Invoice` object, it is very easy to get its associated `Client` and `Line` objects.

For more information about how this all works, search for `Class::DBI`, `Class::DBI::Loader` and `Class::DBI::Loader::Relationship` on **`http://search.cpan.org/`**

UKUUG/Apple Technology briefing - OS X for Intel

Roger Whittaker

This event was jointly organised by UKUUG and Apple and took place on the afternoon of 20th April 2006 in central London.

There were two speakers: Graham Lee of Oxford University Physics Department, whose title was “Integrating Intel into a PowerPC Network”, and Eric Albert of Apple who gave a longer talk covering many aspects of Apple’s move to Intel processors, with particular reference to scientific computing.

Graham Lee manages a lab consisting of Apple machines which are used for practical programming work for students, particularly using the Xcode development tools. Recently this lab has begun to use new Intel machines alongside the existing PPC machines. From the outside, as he demonstrated with a photograph, the Intel and PPC machines are virtually identical. His aim was to ensure that users could log in and use any machine in the lab without any difference in their experience and without even needing to know which architecture they were on.

In particular Graham discussed the universal binary format and the lipo tool for creating these from their architecture-specific counterparts. The Xcode tool produces host-architecture-only binaries by default, and he described the steps he has taken to make it produce universal binaries (essential if students are to be able to log in and continue their work at any machine in the lab).

As yet, installation and netboot images are still architecture dependent: he uses tools from `bombich.com` to solve this problem.

For the open source tools from fink and darwinports, universal binaries are still not an option. This means that the lab has been forced to maintain dual directory trees for the two architectures.

Graham’s slides are available at:
`http://www.ukuug.org/events/apple06`

Eric Albert’s talk took up the remainder of the afternoon. He began by talking about the reasons for the switch to Intel, though it should be said that he discussed this topic purely in technical terms, with very little said about any possible commercial or political reasons for the change. At various points throughout the talk he mentioned various necessary technical preparations which were being made for the change well before any announcement was made. Indeed, Mac OS X has always been compiled internally on both architectures, so the development and testing process has been a long one.

He went into a fair amount of detail about the technical details of the transition, particularly “endian” problems. The layout of universal binaries was explained, and Eric was at pains to explain that there is no performance hit as a result of the use of universal binaries, and that the additional disk space used is not a major problem, the overall size of the OS having only increased by about 30% as a result.

Eric also described Rosetta, Apple’s on-the-fly translator which can execute PPC binaries on the x86 architecture. Although there there is inevitably a performance hit, Rosetta works well with certain limitations: it is not able to deal with “classic” (Mac OS 9) applications. It also cannot of course deal with G5-specific (64-bit) applications (as 64-bit Intel Apples do not yet exist), and can’t handle browser extensions or kernel extensions.

Other issues that caused problems for the switch of architecture were Apple’s legacy “resource fork” and “resource file”, the handling of aliases, byte order in UTF-16 unicode files as well as problems compiling code with all but the most recent versions of `gcc`.

Eric also touched on virtualisation, the EFI boot architecture and the advantages that OS X has in not having any legacy baggage on the Intel platform to support, which has meant that it has been easier to support the advanced features of the modern processors to the full.

The talk ended with some discussion of scientific software on OS X for Intel, including OsiriX, Wolfram Mathematica and GeneSpring, all of which are now available in universal binary format.

UKUUG wishes to thank Massimo Marino, Alan Bennett and Eric Albert from Apple, and Josette Garcia from O’Reilly for all their work associated with this event.

C in a Nutshell: A Desktop Quick Reference

Peter Prinz and Tony Crawford

O'Reilly Media
ISBN 0-596-00697-7
608pp.
£ 28.50
Published: January 6, 2006

reviewed by Graham Lee

"C in a Nutshell" caught my interest in a way that no other C reference I have read ever managed to achieve. A large part of its refreshing approach comes simply from the fact that this is a new book, not a revised edition of an existing work. This means that features of the language new with C99 are treated alongside the longer-standing features, rather than being relegated to sidebars or footnotes. For instance, the chapter on functions includes a section on the new inline keyword, and the new boolean and complex floating-point types are discussed in with the familiar `int`, `char` and friends. Such newer features are still marked out as such in the text, and this approach is useful as it reminds those familiar with older variants of C of the 1999-specific revisions, without giving newer readers the notion that such portions are somehow novel or side issues.

The heavy use of example code in "C in a Nutshell" complements the full treatment of the topics in the text. Where a complete discussion of pointer operations takes up three pages, a half-page example function demonstrates the commonly-used aspects more succinctly. The chapter on memory management is almost exclusively devoted to an implementation of a binary search tree, again to demonstrate 'real-world' use of the matter under study. About a third of the book is given over to the chapter on standard library functions, and with most if not all modern Unix platforms distributing a comprehensive online manual, again the examples of using standard library functions are what make this section. It isn't going to break me out of my ingrained "man foo" habit, but for those times when the man page is just a little too terse this will be a great fallback. A quick word of warning though: in discussing `math.h` functions, the examples relate only to situations where the functions set `errno` on error. Consideration of floating-point exceptions is covered elsewhere in the book. This had me going for a good few hours as I tried to work out why the example didn't behave the same way on my system as theirs.

The final section (under 100pp) of the book covers the GNU tools `gcc`, `make` and `gdb`. I can't work out why this section should exist at all; not only are there better (and certainly more complete) discussions of the GNU toolchain available, but the style shifts from being a reference to an overview. A reader with a serious interest in using these tools for their C development would be better off with more specific references for them, such as the texinfo documentation or a different book.

The content is mercifully low on errors; it's all too common for books with large quantities of code fragments to contain gremlins, but in this case both the code and text are of high quality. The few mistakes I noticed didn't seriously affect the book's utility as a reference; for instance in the previously-mentioned binary search tree example, the text and the code disagree on whether equality is treated with the greater-than or less-than case in a condition. In practice it wouldn't matter which were used. As with much of the Nutshell series, this book is aimed at the competent programmer who needs a quick reference, not at the beginner. As a teacher of C programming, I had been looking for a reference work which covered the C99 version of the language standard, and did so in a readable format free of omissions and errors. "C in a Nutshell" did not disappoint, and the utility of the standard library reference was a welcome surprise.

Linux Server Hacks: Volume 2

William Von Hagen and Brian J Jones

O'Reilly Media
ISBN 0-596-10082-5
478pp.
£ 20.99
Published: January 6, 2006

reviewed by Mike Smith

I don't believe it: I knew I'd reviewed the original Linux Server Hacks book for the newsletter, but didn't realise it was nearly three years ago! How time flies.

As the name suggests, this second volume is not a 2nd edition, but a completely new book. To set the scene, for both the book and this write-up, the authors explain in the preface that they both owned the original work so my expectation is for more interesting and more advanced hacks in this volume. This volume also has nearly twice the number of pages of the volume one.

The book is split, in the usual way, with chapters on various different topics including (but not limited to) authentication, remote access, services, storage, security, troubleshooting, monitoring and recovery.

I've read a few chapters now and, lets get to the point: I don't like this book.

I have a feeling I've said this before in a review – a hack is something clever; a combination of techniques and tricks to come up with something new. So far I have found this hacks book to be just a set of simple HOWTOs.

Even worse, there are a couple of "hacks" that tell you how you should 1) go about getting Linux introduced into your (corporate) environment, and 2) prioritise your work (by writing lists, and stating when you need to get something done, no less). What have these activities got to do with Linux server hacking?

There are even "hacks" that talk about setting up remote printers in Windows and OS/X – in a Linux Server hacks book for heavens sake. I think they've really lost the plot. (Yeah, okay its related to CUPS, but come on.)

Some of the HOWTO-like activities are: installing DHCP (using up2date, apt-get etc); using PAM; configure Kerberos (not possible in the couple of pages given to the task); lock an account by using an asterisk in /etc/shadow (!); stop all logins with: touch /etc/nologin (!!).

Some of the recommendations are: use VNC; setup ntp; use macros in VIM; remove files to free up space (wtf?).

There may be something useful, but if there is I'm not going to find it as I've given up – it's a disappointing book.

Skype Hacks

Andrew Sheppard

O'Reilly Media

ISBN 0-596-10189-9

342pp.

£ 17.50

Published: December 16, 2005

reviewed by Mike Smith

See the combined review below.

VoIP Hacks

Theodore Wallingford

O'Reilly Media

ISBN 0-596-10133-3

326pp.

£ 20.99

Published: January 6, 2006

reviewed by Mike Smith

I thought it a good idea to review this pair of books to compare their content, as the subject matter of the two is related. Skype is obviously one form of VoIP, and we should expect the VoIP book to look at SIP phones, H.323 and hopefully a range of wider issues. This review also continues a theme from previous newsletters, as I have looked at a number of the members of the hacks series now.

Both books are a little thicker than the previous hacks books I have studied – they still have 100 hacks, and this is an indication of richer content, and bodes well.

Skype Hacks

This book is split into 12 chapters, ranging from the basics of setting up and configuring Skype through to advanced uses for business, mobile devices, chat, voicemail and tools. The last chapter on automation looks interesting, with email, calendar and SMS integration.

I've had Skype for a long time, but not a regular user, and at one stage did buy some SkypeOut credit just to try it out. I should have heeded hack #21, which warns you not to forget that you need to actively use SkypeOut at least once every 180 days, or you lose your outstanding credit! With renewed interest because of this review, I tried to use some of my apparent 9.77 Euro credit reported by the Skype client – only to find out when I attempted a connection that credit is no longer there. Rats.

Most people look at Skype, and VoIP in general, because they want to reduce their telephony costs. Of course this is true with Skype when calls are placed only over the Internet. However you have to think twice if you're considering SkypeIn or SkypeOut. One factor in the cost of calls is the unit of time that call durations are rounded up to for billing – whether per second, per 6 seconds or per minute, say. SkypeOut uses per minute billing (rounding up to the next whole minute) so if you have something less it may not be a straight-forward reduction in costs for you (though in general I'm sure it probably is). There's a whole chapter dedicated to the financial side in this book.

[I hadn't heard the term Dry DSL previously – meaning a DSL service without an associated telephone number and voice service. We're getting to the stage where this would be a real benefit in the UK as we could run all of our home and office voice over it and do away with the analogue services. Let's hope.]

There's a nice Skype toolbar, outlined in Hack #43 that will let you call telephone numbers using Skype from a browser, and you can create shortcuts to common numbers/contacts by using command-line parameters, which is great (Hack #49).

Hack #54 outlines how Skype can be used to provide call centre facilities. I find this amazing, but not sure if anyone is using it in this manner.

There are all sorts of other tips, such as setting up freecall numbers, and alternative international numbers for your mobile. Jyve needs to be checked out, and there are add-ons like Gizmoz which looks fun (animated faces that presumably move in unison with your voice).

Overall I like the variety of hacks in this book and recommend it if you're set on using Skype. It contains lots of food for thought in possible configurations that will reduce your dependency on, and perhaps even eliminate, your fixed line voice services.

VoIP Hacks

This book has its 100 hacks split into just 7 chapters, one of which covers Skype. The 13 hacks in that chapter can be considered a distillation and the most important messages from the Skype Hacks book – though they are not the same. Examples are integration with address books, Jyve and a few other topics.

The Skype protocol is proprietary, and the dominant VoIP protocol is now SIP (Session Initiation Protocol). A new contender is IAX (pronounced eeks) has the advantage that it works in NAT environments, so we'll have to watch how its usage grows. The rest of the book covers infrastructure and clients using these protocols.

I found much of this text, and especially the first chapter on broadband VoIP providers to be US centric. So the advice is sound, but the examples not always appropriate. You can get a list of UK providers here:
`http://www.voip-info.org/wiki/view/VOIP+Service+Providers+Residential`

There's a chapter on Asterisk, which I am sure you'll know is now an important way to run a telephony server on Linux (and Mac and BSD) systems. I've had a look at Asterisk previously without much success due to lack of time (and a hardware SIP phone), but again this review gave me the impetus to have another go (there are a bunch of softphones listed at
`http://www.iptel.org/info/products/sipphones.php`

SJPhone and Firefly look reasonably good. Some of the hacks cover the basics of configuring Asterisk and a little on reporting and some tricks (like obtaining automated weather reports).

There's actually a pre-built VMWare VM with Asterisk (using astLinux) that might save you some time if you want to experiment – have a look on the VMWare website.

There's a chapter with a dozen hardware hacks. Many of these were specific to one particular model of IP Phone or other hardware. I think its unlikely typical readers will use the array of hardware addressed in each of the hacks, so I think the content has somewhat limited use.

The penultimate chapter looks at monitoring and some internals of the VoIP protocols, looking at latency and jitter, graphing the stats and traffic shaping. This has a lot of good (but brief) information on diagnostics.

The last chapter, "Hard-Core Voice", covers some advanced topics, such as interfacing Skype with Asterisk, and indeed the whole lot with your home (IP) phones so that incoming Skype calls with ring everything. Good stuff.

The book provides hacks for Windows, Linux and Mac based applications, and there are number of Mac-only hacks, so the relevance of the whole to the UKUUG community may be slightly reduced, but I still think this is a worthwhile wide-ranging set of hacks for general knowledge.

Wireless Hacks

Rob Flickenger and Roger Weeks

O'Reilly Media

ISBN 0-596-10144-9

463pp.

£ 17.50

Published: December 30, 2005

reviewed by Mike Smith

Yet another book with Rob Flickenger as (co-)author. This hacks book is now in its second edition, though unfortunately I can't say where the enhancements are, not having read the first. At over 400 pages though, it's a reasonable size, with the usual 100 hacks split into chapters on different topics.

From the title I was expecting a book on WiFi networking (everything seems to be 802.11[something] these days), and although this is covered in large measure, the hacks aren't restricted to this topic.

In fact we start off with a chapter on Bluetooth, mobile phones and GPS. This covers things like remote control using mobile phones (over Bluetooth) of applications on OS/X, Linux and Windows. Hack number 16 is about controlling a Home Theatre using a Palm, which gives me the opportunity to tell you about an article I wrote on linkstationwiki.org – having replaced the firmware on my Buffalo LinkStation, I've been extending the PCast media server menus accessed by a Buffalo LinkTheatre I treated myself to recently. Not really relevant to this review but may be of interest if any of you have a similar setup.

So then we do move on to the WiFi world - a bit of NetStumbler, WiFiFoFum, Kismet and the like, Ethereal, whatever that is, and ngrep. One tool I hadn't come across though (for Windows unfortunately, so apologies for mentioning it) is Qcheck - apparently a free tool from NetIQ that lets you measure the actual bandwidth of your wireless network (not the inaccurate bandwidth reported by the stock applications). Interesting.

Then we have a section on security – nothing really exciting here, then a chapter with various hardware hacks: antennas, adding Bluetooth to a mouse and making a WiFi remote controlled car (using a WRT54G).

One area I've missed out on myself is the Linksys WRT54G router. Before I knew about the hackability of this I had already settled on a Netgear DG834G. The latter does have some hacks worth knowing and the firmware source code is available, but the community effort doesn't appear to be there. There's a bit about the WRT54G within the chapter on software hacks.

I think some of the hacks are now showing their age – the first edition was published in 2003. So although still relevant, I suppose, explaining software like DriftNet and making antennas out of tin cans does take me back a few years.

There's a final section on wireless standards. Alas the standards cover a, b and g but not n and s. Pre-n equipment is already available, but I'm waiting for s now (I'm waiting for snow too, but lets not get confused). I'm hoping to persuade my neighbours to implement s equipment too, in the hope that we can aggregate our bandwidth (assuming this will be possible – I've not investigated). I know we should be able to do this already with AODV and the like.

So in summary, a reasonable book; parts are a bit dated; covers Windows, various PDAs (Palm and Windows based) and OS/X, and some Linux – and therefore partly of interest to the UKUUG membership.

I'm not raving about it but there are some interesting snippets of useful information. I now realise I have been quite harsh in one of my other reviews this month – perhaps too harsh, as any published book has an audience (just not always suitable for the particular individual performing the review).

Essential PHP Security

Chris Shiflett
O'Reilly Media
ISBN 0-596-00656-X
124pp.
£ 20.95
Published: October 28, 2005

reviewed by Paul Waring

As a PHP developer of many years, I've probably made my fair share of mistakes when it comes to security, so finding a book that concentrates purely on PHP security sounded like just the thing I've been looking for.

My first impression was that the book is rather slim, weighing in at just over 100 pages. Whilst PHP security is perhaps something of a niche topic, I got the feeling that perhaps this text would be a little short on detail. On the other hand, the book lacks the flowery prose and history sometimes used in computer books, so it was a refreshing change to get straight into the nuts and bolts of the subject matter from page one.

The first few chapters cover the most obvious security holes that often plague PHP scripts, including processing user input (primarily from forms), databases and cookie/session handling. At first this seemed like a good coverage of the common security problems, but I felt that the databases section was particularly short, especially given the number of websites that are relying on some form of database backend to provide various levels of functionality (e.g. forums). The SQL injection section was only a few pages, despite the fact that this one security hole alone has caused problems for many of the popular PHP scripts available for download.

The author also sometimes veers away from PHP problems and ends up discussing more general issues, such as the handling of cookies in previous versions of Internet Explorer. Whilst this is something that developers should be aware of, it isn't really a PHP security issue and relies on users being savvy enough to keep their software up to date.

There are sections where the author ends up explaining what is basically the same vulnerability several times. For example, when using some form of user input (such as `$_GET['filename']`) as a way of dynamically including files, he shows the same problem with both local and remote files, and so the reader is given a repeated explanation of why you should never use tainted data to refer to a filename.

Certain sections are also somewhat hand-wavey in that the author will mention topics such as SSL, and then suddenly move onto something else without giving any details as to how the topic in question can be used to enhance the security of your PHP scripts.

Having negatively criticised the book somewhat, I must admit that I did learn one or two useful bits of information. For example, I always set the error reporting value for my PHP scripts to be `E_ALL` to display all errors, but apparently including `E_STRICT` will warn about depreciated functions as well. The section covering problems with shared hosting was also fairly in-depth and is a topic which probably affects the majority of PHP developers, yet it is not usually considered when trying to write secure code.

Overall though, I think the book is a little short on detail. There are a lot of places where it highlights important security issues – thus bringing them to your attention – but doesn't really provide many good examples as to how to close the holes.

If you're a seasoned PHP developer it's unlikely that you'll learn a great deal from this book, although it may serve as a useful reminder of some of the security issues that you are likely to come across in your scripts. The majority of the book's content can be found on the many PHP websites, and given the lack of detail in the book I suspect most developers would get a more comprehensive answer by simply searching the web.

Personally, I don't believe that the book is worth the cover price, though if a second edition with more detail and better examples was released then I'd be inclined to pick up a copy. As it stands, the book has potential but some sections could do with expanding, so I would award it 6/10 overall.

The Book of Postfix

Ralf Hildebrandt and Patrick Koetter

No Starch Press
ISBN 1-593-27001-1
496pp.
£ 30.99
Published: March 25, 2005

reviewed by Robert Easthope

"The Book of Postfix" is a fantastic book explaining all aspects of the Postfix mail server system. Its all in there how to configure a basic setup where you just want to relay mail from one host to another, to very advanced content filtering and integration with database servers such as MySQL.

The first five chapters introduce mail servers in general and explain how to configure the DNS. There is a basic explanation on the structure of the system and what commands are available for administrative and other tasks. Personally I found chapter four very useful for explaining configuring mail relays and SMTP authentication.

The second part of the book concentrates on content control. Chapter six has some good examples of how to filter on header information, standard mechanisms available in all mail servers such as checking if the from address maps to a valid IP address in the DNS etc. Building restrictions based on keyword triggers is covered and also how to test these filters in a test environment and later on testing on real messages.

Part three of the book is on advanced configurations, building a server to answer to multiple domains, building in support for MySQL integration, configuring mail relays and SMTP authentication.

Part four is a small one chapter section on fine tuning POSTFIX. This appears to be for those who handle huge amounts of mail traffic and need to put limits on what the server can handle at any one time. It also explains in great detail how you can get statistical data on what the server is doing something that can be very useful in debugging problems.

The Appendix section is very useful as a reference for debugging problems. It explains clearly what to check on the system when postfix is not responding and details common errors in configuration files for postfix.

To sum this book up I would say it's a good all round guide to POSTFIX, it pitched at the right level that if you are new to mail servers it has enough introduction, but also a very good guide for those like myself how have previously used other mail servers such as EXIM and Sendmail.

Internet Forensics

Robert Jones
O'Reilly Media
ISBN 0-596-10006-X
238pp.
£ 28.50
Published: November 22, 2005

reviewed by Gavin Inglis

The computing world has needed a book like this for some time. Using crime scene forensics as a model, Robert Jones demystifies and explains the hidden information available online that anybody can use to protect themselves against – and perhaps track down – scammers, spammers, and other forms of cyber-scumbag.

No prior knowledge is assumed. Jones begins by illustrating how IP numbers work, then domain names, then ways to divert DNS requests. The more experienced reader will be tempted to skip forward, but this would be a mistake; at each point there are little nuggets of information which the average web professional may simply not be aware of. The text strikes exactly the right balance, with sufficient information to explain each concept, but very little padding.

The topics one might expect are all included: email headers and attachments, URL manipulation, vulnerabilities in web sites, HTTP headers and information leaked by web browsers. Diagnostic tools such as `dig`, `whois` and `traceroute` are demonstrated. Perhaps more importantly, substantial space is given over to interpreting their output.

Cleverly, the example scams, viruses and spams are all real, drawn from the author's own experience. He explains his thinking and reasoning in a straightforward and involving way that Gil Grissom could learn from.

This of course makes for some fascinating detail along the way. There's the jaw-dropping hole in eBay security that offered phishers a custom-made redirect to send to their victims. There's the classified PDF document from the US military about insurgent attacks in Iraq that was declassified by a bored Italian using cut and paste. And there's the wonderful idea that spam could be used to send coded messages between spies or criminals in much the same way that they used to use classified advertisements in the newspaper.

While the technology described is platform-independent, it becomes quickly clear that a UNIX-based machine is the best choice for applying the techniques explained by the book. Examining virus payloads on Windows is rather perilous. The new tools included with the book are written in Perl.

Jones's motivation is interesting. There is an early reference to open source developers as the "stewards and guardians" of the Internet versus those who have turned to more opaque platforms. During the closing chapter on what is being done to fight cyber crime, he expresses the hope that he might help to create a kind of "network Neighbourhood Watch" to make life difficult for the casual cybercrime. Sensibly, he advocates leaving the investigation of child pornography and extremist sites to the professionals.

"Internet Forensics" compiles a lot of information that has typically been available only by word of mouth or bitter experience. The average user will find themselves able to apply its techniques, and even if they choose not to, they will gain a deep understanding of how the Internet works. The book's clear style and firm grounding in reality make it an excellent read.

Mind Performance Hacks

Ron Hale-Evans

O'Reilly Media
ISBN 0-596-10153-8
304pp.
£ 17.50
Published: February 28, 2006

reviewed by Gavin Inglis

The predecessor of this book, "Mind Hacks", was subtitled "Tips and Tools for Using Your Brain". "Mind Performance Hacks" goes a step further, with "Tips and Tools for Overclocking Your Brain". This sums up the difference quite neatly; whereas the previous book was focused on understanding how the brain is put together, this title concentrates on specific applications to improve its performance.

Each of the eight chapters concentrates on one area of cognition. Ron Hale-Evans is clearly a fan of Frank Herbert's Dune books, and inspired by the Mentats – human computers – of that universe. This reader fully expected to find Hack #76: The Litany Against Fear in the chapter on Clarity.

Chapter one deals with memory. It introduces various systems which exploit visualisation, association and even rhyme to improve your recall. Some, such as the mnemonic system designed by World Memory Champion Dominic O'Brien, require a little preparation to use. Others are more intuitive, such as surreal images constructed around a familiar location. The Hotel Dominic system, designed by the author, can supposedly be used to memorise 10000 pieces of information, or more.

Information processing is covered next, from capturing and storing ideas you have during the day, to programming yourself with a deck of cards. Learning styles are explored, such as visual versus aural or kinaesthetic learning. This might help you understand your friends and colleagues better. Valuable for clutterbugs is the eerie Hack #18: pre-destroying rubbish through coded marks. The text reflects on the nature of the past and the mental traps it lays for us in the present.

The Creativity chapter is full of fun stuff to do, from using a completely irrelevant item as a trigger for brainstorming, to channelling heroes of stage and screen: "What would Mary Poppins do in this situation?" We're introduced to creativity decks such as Brian Eno's famous Oblique Strategies, and the odd constrained writing of the Oulipans. We voyage even into the unconscious, with simple techniques like keeping a dream journal and cunning tricks to access the hypnogogic state, halfway between wakefulness and sleep. Interestingly, sleep, nutrition and exercise are key components of the later chapter on maintaining mental fitness.

The sections on Maths and Decision Making are shorter and include the kind of 'rules of thumb' that used to be taught in schools but have been exterminated by the calculator. Whilst estimating square roots and applying game theory to decision making are certainly valuable, one wonders if the seven pages devoted to Hack #40: Count to a Million on Your Fingers really justify their inclusion.

It's surprising to find an examination of James Joyce's Finnegan's Wake in the chapter on Communication, and readers could be forgiven for not learning an artificial language such as Klingon. However there are some very valuable ideas here for turning a fear of public speaking into a positive state of excitement. This theme continues in the brave subsequent chapter, with rational thinking about emotional states and irrational behaviour. Here there is controlled breathing, meditation, and even interviewing oneself to get to the root of what we *really* think.

After all this mental graft it's a relief to encounter Hack #67, which advocates playing board games to improve strategic thinking, negotiation and deduction. Even easier is Hack #68, which suggests improving visual processing through first person shooter video games. At last, a way for teenagers to answer back parents who complain they spend too long on the Playstation. “Aw mum, but I’m reducing my attentional blink!”

“Mind Performance Hacks” is an entertaining and rewarding read for anyone interested in improving the way they use their brain. Many of the hacks are really a framework upon which to build your own systems. Relatively little of the material is brand new, but collecting it in one book makes it very accessible. The techniques are inspiring and practical.

Essential SNMP (2nd Ed)

Douglas R Mauro amd Kevin J Schmidt

O’Reilly Media
ISBN 0-596-00840-6
460pp.
£ 35.50
Published: September 23, 2005

reviewed by Greg Matthews

Network administration is a very important job, quite distinct from system administration. It is the network administrator (NA) who has to install, maintain and run the network while keeping a keen eye out for security issues. Any network larger than a modest home network is too big to keep tabs on manually. Every NA worth her salt will have some sort of network monitoring software running 24 hours a day. To this end the Simple Network Monitoring Protocol (SNMP) was designed to be a low-overhead protocol for managing IP devices. SNMP can be used to gather information from devices such as switches and routers and also to change information on these devices. In fact these days SNMP can be used to manage just about any host on the network from Unix machines to modem racks. Together with Remote Network Monitoring (RMON), it provides a full suite of tools for examining the health of a network and all its associated devices.

This all sounds great, and indeed SNMP is a powerful tool in the NA’s toolkit but remember, this is a protocol, not a shiny piece of software. I remember two things about SNMP from my days as a NA. First, the security of the protocol was practically non-existant. Second, every vendor published their own intractable Management Information Base (MIB) containing the schema for managing their particular hardware and many of them don’t publish this data in an accessible form.

The concerns about security have largely been addressed with the most recent version of the protocol (SNMPv3) although many vendors continue to ship hardware that only conforms to SNMPv2 or even v1. The Second problem can be solved by spending an enormous amount of money on Network Management Software (NMS) such as the ubiquitous, but pricey, HP OpenView, although the last few years have seen an increase in low cost and no cost software of increasing quality such as OpenNMS.

So it would seem things have improved since I last looked in any depth at SNMP. This book would have been incredibly useful when I was first struggling to construct “mrtg” configuration files to monitor the entire departmental network. The first few chapters of the book are a quick but effective introduction to SNMP and what has changed in version 3. They cut through most of the confusion that has resulted from the change in terminology, discuss the new secure authentication and privacy, and introduce the concepts of managers and agents, MIBs and OIDs (Object Identifiers). This is followed by a quick introduction to NMS before a burst of chapters on retrieving and setting MIB values, polling and traps. Examples are illustrated with “ethereal” traces of captured packets, which aims this book squarely at the professional administrator. Code examples are given in Perl using the `Net::SNMP` module, a good choice as Perl is extremely widely used in network administration tools, and scripts can be easily extended to suit the individual NA. Along with Perl scripts there are sections on the famous Multi-Router Traffic Grapher (MRTG), RRDtool and Cricket as well as a chapter on the use of Java with SNMP. If this wasn't enough, the appendices go into even more detail on HP OpenView, Perl and the command line tools available on Linux and Unix, as well as listing all the RFCs that are relevant to SNMP (there are a lot!)

It is difficult to find fault with the content of the book. Although HP OpenView does not get its own chapter, its status as the Grandaddy of NMS is reflected in the way it is compared to other software throughout the examples. This is a solid textbook in the O'Reilly mold. The only criticism I have is an unfair one, I had difficulty staying awake whilst reviewing it, I find SNMP just about the driest subject in IT but this book will enable you to get through all the tedious configuration as fast as possible allowing you to move onto more interesting things that much sooner.

Google Advertising Tools

Harold Travis

O'Reilly Media

ISBN 0-596-10108-2

366pp.

£ 20.99

Published: February 3, 2006

reviewed by Greg Matthews

I run an vacation apartment rental website `http://www.tripseurope.com/` and it is an extremely competitive market. You depend heavily on customers coming through your website, and consequently depend a lot on Google and their advertising tools. You have to get two things right; get your site listed in the first couple of search pages, and have excellent adverts placed around the web, neither of which are easy.

I read 'Google Advertising Tools' in the hope that it would provide some insights into improving the traffic to my site, but I was quite disappointed. The book covers three main areas:

- How to create a popular website
- How to make money by placing Google adverts on the site (Google Adsense)
- How to advertise your site with Google Adwords.

Of those areas, the Adsense section wasn't relevant to my website, but I'd still like to know more. Much of the work in improving your website search engine rankings (called SEO, or Search Engine Optimisation), involves regularly reading Internet forums to pick up tips or be aware of the latest Google updates, and frequently you come across people making lots of money through Google Adsense. I would love to try it, and have wondered how difficult it might be.

However, all three areas were ultimately disappointing. The format of the book was to repeat much of Google's help sections, with the occasional embedded text box giving tidbits of advice. These tidbits were rarely useful, being either obvious or even non-committal. For instance, in the section about the format of your Adsense adverts, the book goes to great lengths on what form fields to fill in on the Google site (all of which is explained by Google as you do this), with a final couple of lines suggesting what colours to choose. The author says "Use the color palette to match your site" (if you do this, it is speculated, some users may click on the ads because they think they are part of your site, not ads) and "Make a color choice that starkly contrasts with your site" (by doing this, the ads are made more noticeable, and thus it is more likely that they may be clicked). No sitting on the fence there then!

If we were to ignore the vast sections with the 'point and click' mechanics of creating and placing Google adverts, then there is very little left to this book. It could be that those remnants are pure gold, but unfortunately this isn't the case. The advice covers the basics of SEO, which could be useful to the absolute beginner, but anyone who has worked on their site for a while will have already picked these up.

So, in conclusion, if you have just created your first ever website then this book could be a good introduction to quickly give you the basics. Be aware that SEO is difficult and slow, with no overnight successes, so don't let some of the examples of earnings given by the book carry you away. Personally I'd recommend you save your money and read an SEO forum every day such as the Web Workshop SEO forum
`http://www.webworkshop.net/seoforum/index.php`

You'll learn a lot more. If you already know what SEO stands for, then this book isn't for you.

Greasemonkey Hacks

Mark Pilgrim
O'Reilly Media
ISBN 0-596-10165-1
495pp.
£ 17.50
Published: November 25, 2005

reviewed by Greg Matthews

This is another publication in the informal O'Reilly "hacks" series, one that in my opinion has never lived up to the early promise of "Linux Server hacks" by Rob Flickenger. Mark Pilgrim is a good choice for the editor of this book, he is the author of "Dive into Python" from APress and `diveintogreasemonkey.org`, a comprehensive online introduction to this Firefox extension.

Trying to describe exactly what Greasemonkey is to someone unfamiliar with the way that web pages are rendered is difficult. However, this tool is written by and designed for the sort of people who have already got their hands dirty messing around with web servers and browsers. Greasemonkey installs as a Firefox extension but offers no new Firefox features or chrome by itself. Instead, it provides a means to write or install scripts that can alter the web pages you visit. This might include increasing the usefulness of a site, adding alternative search engines to the Google site for instance; or fixing annoying bugs, such as that website that insists you use Internet Explorer for no good reason. The scripts themselves are JavaScript and the first 12 "hacks" in the book are in fact a very brief introduction on how to write Greasemonkey scripts. The next 88 hacks are roughly divided into chapters by topic.

Initially, Greasemonkey appears to be a tool looking for a problem to solve. Sure you can do some clever tricks with it and if you need to use a particularly annoying website a lot you might be frustrated enough to hack something together to fix it. Generally though, if a particular site doesn't really fit the surfers requirements, she surfs on somewhere else. In other words, Greasemonkey scripts seem to be too much work for little return.

So as you listlessly leaf through this not insubstantial book (over 450 pages), idly wondering if anyone has copied code off the printed page since the demise of the ZX Spectrum magazines, you might just happen across the answer to the thing that has bugged you most about a particular website. For me, it was the first "proper" hack of the book, "#13 Turn Naked URLs into Hyperlinks". This incredibly simple (˜20 lines) script simply makes non-marked up URIs clickable. Great. This encouraged me to read further and find one or two more that might be useful to me. Each hack is broken up into a brief section on the problem followed by the code (save lots of typing by finding the appropriate section of `http://hacks.oreilly.com/`), and then some notes on how to use it.

All in all, this book is an interesting demonstration of the potential for Greasemonkey. Most of the aptly named hacks simply scratch a particular hackers itch. This of course, is the classic Free Software development model. There will probably be something here to catch the eye of most readers but not enough to warrant the purchase. The problem is that Greasemonkey (and by extension, this book) is aimed at existing hackers who want to tweak their browsing experience. These people are either already using Greasemonkey or will be up and running and searching the web for scripts before this book arrives from Amazon or whichever bookseller offered the cheapest price when they used hack #94 "Compare Book Prices" (seven pages of code including base64 encoded PNGs). The contents of the book are really much better suited to a dynamic media like a wiki. It might be worth buying this for an aspiring young hacker or even your LUG library, but consider that the content will quickly become dated.

Using Moodle

Jason Cole
O'Reilly Media
ISBN 0-59600-863-5
240pp.
£ 28.50
Published: July 29, 2005

reviewed by Lindsay Marshall

With the recent merger of Blackboard and WebCT the choices for commercial course management systems (or whatever you want to call them) just got reduced to, well, one really, so the existence of Moodle is a good thing. The system is well supported and easy to install, and indeed this book tells you nothing at all about how to install Moodle, it assumes that you have a running system already. And of course it assumes that Moodle is the system for you, which actually may not be the case: the Moodle course model (like that of Blackboard) may not necessarily be a good fit for your course structure, particularly if you have a lot of interlinked modules.

The good thing about Moodle is the plethora of different features that it supports: wikis, weblogs, journals, quizzes etc., and this book focuses on these, devoting a chapter to each area. If you are using Moodle and you are the sort of person who likes to have everything in hard copy, nicely bound, then you might like it: clear instructions for using the features, good illustrations, and some tips on getting the best out of them. The problem is that once you've set up a couple of journals or a workshop or two, you'll know exactly what to do and you'll never look at the book again. In fact, if you are the slightest bit web savvy you probably wont need the book because, first, the interface really is easy to use, and second all the information is online anyway. Given its first steps nature, its not cheap either. I must stress that this is not a bad book, it is well written and presented, but for the experienced computer user it might be a bit simple.

If you need a CMS, Moodle is a good place to start. Whether you need this book is another question.

Security and Usability

Lorrie Faith Cranor and Simson Garfinkel (Eds)

O'Reilly Media
ISBN 0-596-00827-9
738pp.
£ 31.95
Published: September 2, 2005

reviewed by Lindsay Marshall

This one is a little different from your usual O'Reilly book in that it is a collection of papers devoted to five aspects of security and usability. There are thirty four papers in all and the five areas are "Realigning Usability and Security", "Authentication Mechanisms", "Secure Systems", "Privacy and Anonymity Systems" and "Commercializing Usability", plus a section devoted to three classic papers in the field.

There are some well known and respected names amongst the authors as well as newcomers with the papers being a slightly odd mix of theory and practice. The editors (rightly) justify this because of the youth of the field, and they also emphasise the security focused nature of much of the material. Their feeling is that the main push in improving the usability of secure systems must come from the security people, and I would have to agree: systems must be secure first of all, but that is no excuse for neglecting (or at least finding out about) users' experiences with systems. And so often, of course, a poor usability leads to security horrors like people leaving passwords written on whiteboards and such like.

This is a book to dip into rather than read at one sitting – I certainly haven't got to everything in it yet. As you would expect the quality of the papers is variable, though all are at least good. Your interest in the relevant area will also influence your perception of some of the material quite dramatically (I can't get excited about security on Lotus Notes for example.) The book is also pretty expensive: something to have in the project library rather to have as a personal copy. My biggest complaint is the lack of material on Trust and/or Reputation, this is an area with a lot of relevance to security and usability and where there is much interesting work. It could easily have filled another section, though by then the book would have been reaching Harry Potter dimensions.
It's good. Buy it for your team library.

Web Site Measurement Hacks

Eric T Peterson

O'Reilly Media
ISBN 0-596-00988-7
430pp.
£ 17.50
Published: September 2, 2005

reviewed by Lindsay Marshall

I hope the person who suggested turquoise as a good colour for emphasis in text isn't involved in designing a web site for a business. Ughh!!! The whole book looks washed out and pale because of it, particularly as the body typeface is very thin. With my ageing eyes I found this really hard to read. I suppose old people aren't supposed to be involved in the bright new world of the Web so that's probably OK then. Oh, and while I'm complaining, I think website is one word not two now. (But to balance that out I think that e-mail should have a hyphen in it.)

Anyway, so it looks ugly, what about the content? Well, since I'm being grumpy, let me say that the "Hacks" series of books really annoys me. There are always some pieces of computing ambergris brought up from the depths: rare, perfumed and desirable, but there is also too much of the stuff that comes out the other end as well. But to be fair this book does seem to have more than its fair share of useful stuff, particularly if you are new to messing around with logfiles and stuff. The biggest danger I feel is that if you were to actually try and do everything mentioned in the book (and I suspect that some people might) you could spend all your time measuring and analysing and never do anything about making sure that your site was worth visiting. Not too expensive and I will certainly add this to my reading list of books for when I teach about website management.

The headings really get to me though: "Leverage Referring Domains and URLs", "Measure Organic Search". Grrrr.

SQL Cookbook

Anthony Molinaro

O'Reilly Media
ISBN 0-596-00976-3
504pp.
£ 28.50
Published: January 6, 2006

reviewed by Lindsay Marshall

See the combined review below.

Web Site Cookbook

Doug Addison

O'Reilly Media
ISBN 0-596-10109-0
280pp.
£ 28.50
Published: February 24, 2006

reviewed by Lindsay Marshall

Why cookbook and not website? Who can tell. I like the cookbook series. Which is slightly odd, because it is quite like the Hacks series which I pretty much loathe. For a start, the presentation isn't an annoying numbered list of short snippets, instead, its a set of chapters (made up of numbered sections, some of which are really long). There is also a feeling of solidity about the ideas presented, not like some ephemeral Amazon API thing that might stop working tomorrow. There's also proper PHP code, not that Perl nonsense.

There are some cracking ideas in the web book: adding watermarking to images on the fly is a neat trick. There are some rubbish ones: disabling right click to prevent image downloading is a) stupid, b) irritating, and c) just asking for people to work round it. There are also rather too many obvious ones: adding a doctype to your document and validating it, using basic CSS etc. This one is a difficult call. For experienced and knowledgeable users there is probably too much "how to boil an egg" material, but for people needing a leg up it's a good collection of ideas and pointers. (If you value your eyesight avoid some of the illustrations which feature amazingly tiny text)

The SQL book is twice the thickness of the web one, though I suspect that is because there are lots of examples of query results which take up a lot of space, rather than there being more information. (Also, the vagaries of all the popular database systems (you know who I mean) also contribute to the bulk). Regular readers will know of my woeful ignorance of databases and how to use them to best effect, so I had high hopes for this one. Especially the mystery of joins. As yet however I am still disappointed. The layout is not fantastic and great swathes of SQL are not attractive. And like all SQL books there seems to me to be some essential explanatory thread missing: I understand the examples, I see what's going on, but it doesn't connect. The terminology bogs me down too. What the hell is a correlated subquery? In amongst all the stuff I don't understand, there's also a lot of "boil an egg" stuff here too: use an ORDER BY to sort your results, use INSERT to insert a new record! Much the same as the web book really.

So, both books, probably not for the 42 degree thetans, but useful for the rest of us.

Time Management for System Administrators

Thomas A Limoncelli

O'Reilly Media
ISBN 0-596-00783-3
226pp.
£ 17.50
Published: December 9, 2005

reviewed by Lindsay Marshall and Alain Williams

Lindsay Marshall's review

Save time, don't bother reading this book. Or the review. But since you are still here, let me expand on my reasons for saying this. The book is well written, the advice in it is sound and well known to work. There are even some ideas specific to system administrators that are pretty good (though there are also some that will put you to sleep: using aliases in the shell for instance) The big problem is that there are hundreds of books on time management that cover exactly the same ground and recommend the same solutions, most of them cheaper and prettier than this one. Just not as geek oriented.

At the very end of the book the author even suggest that you read some "traditional" time management, and, inevitably, recommends David Allen's Getting Things Done – the Internet's favourite time management guru and his book. You really would be better of reading GTD first, then have a quick trawl round some of the GTD websites, pick up some tips and apply them to your own situation. System Administration is just not that different from any other job. (Avoid subscribing to the GTD cult though – some people are scarcely obsessed with the whole thing)

And if you can't be bothered, go on with leaving things to the last minute, and try my hot tip: if it's really important, someone will get back to you about the things you haven't got round to doing yet.

Alain Williams' review

A system administrator is typically someone who attempts to do project work while being interrupted by users or customers. If you are not careful you end up spending all your time dealing with the interrupts and never getting anything done; I find this an easy state to get into.

The chapters contain nice summaries that I have been using to try to put into practise before reading the next one, some are much easier than others, some I was already doing. The sort of advice is: prioritise, what must be done now, what can wait; let users know what you are doing and when you will fulfil their request; get users agreement with your timescales; help users to help themselves; automate common tasks.

He will also get you to think about what you do best and when you work best (I am not a morning person). Schedule your work to take advantage of that. How you can organise the windows in your virtual window manager to best effect. How you can organise your team so that you all get some work done. How you can record the tasks that you have to do - if you record it you use less brain power trying to remember the jobs that you have to do today.

Email: how many hours a day do you waste reading it and how many are just sitting in your in-box waiting to be processed? I thought so. This is one chapter that I am going to have to reread, and then again to try to get it right. Yes: I am an ineffective email junkie.

Stress hinders productive work. How to cope with stress, how to avoid it, how to get the best from holidays (leave the laptop at home). If you are stressed what can you do (e.g. try yoga). My complaint about this chapter is that it didn't advise on how to cope with an ex-wife.

A part of organising your work time is also organising your leisure time, your life goals. Yes: many people just life from day to day, year to year and never get where they want because they don't steer themselves.

I liked this book, easy to read and contains good advice. I have fallen asleep reading other time management material, this one kept me awake.

I am reading a chapter a week and have not reached the end yet; if I had I might have sent this review in by the newsletter copy date.

PHP Hacks

Jack Herrington

O'Reilly Media

ISBN 0-596-10139-2

464pp.

£ 20.99

Published: January 3, 2006

reviewed by Alain Williams

My feelings on this book are mixed.

There is a lot that is not so much about 'PHP Hacks' but tips in an environment where PHP may be used. For instance: Hack 60 is all about Apache's `mod_rewrite`, useful and a good summary, but not much to do with PHP. Hack 5 is about using CSS to create HTML boxes, wrapped up in some PHP code. Much of the code has little or no comments, the adjacent text is only sometimes really enlightening.

The author uses some PHP-specific concepts such as output buffering (`ob_start()`): it would be nice if he had taken the opportunity of making this a real PHP Hack and explained why he used them since many people will have overlooked this PHP functionality.

Hack 87 'Build GUI Interfaces with GTK' I thought was really interesting – let's use PHP with a GUI but not in a web environment, this will break new ground for me. Unfortunately the description of what it doesn't really tell me how it works and points to a URL where I can get more information. A whole chapter of Hacks on this would have been great. The previous Hack on Custom Maps with MapServer is much better, well written background, concepts and code with comments.

There is some use of PEAR modules, this is welcome since PEAR is notoriously badly documented. I similarly liked Hack 68 'Create Objects with Abstract Factories' and the ones that followed, Hack 36 shows PHP5's `__call`, `__set` and `__get` with some nice explanations and diagrams. I can write and use simple objects – these will help me get my head round more OO ideas.

The chapter on graphics is worth reading if you want to see how to manipulate images (e.g. scale them) or drawing graphs. Also useful are some scripts that implement a login system, many people seem to find this a difficult thing to get off the ground. This is database independent (using PEAR DB), he includes MySQL table definitions and sample user entries so it is quite a complete example.

Similarly creation of a simple shopping card (Hack 66) is covered but there are no comments and not a lot of description of the PHP or JavaScript. It would have been nice if this had been integrated with Hack 62 which is a, well written, description of a PayPal 'Buy Now' button.

Most of the book would be better titled a PHP Cookbook since I expect "Hacks" to tell me things about the language that I may find hard to understand or demonstrate neat tricks that are peculiar to the PHP language; there is some of that but a lot which is neat snippets that happen to be written in PHP; and some that belongs elsewhere.

Understanding the Linux Kernel

Daniel P Bovet and Marco Cesati

O'Reilly Media
ISBN 0-596-00565-2
920pp.
£ 35.50
Published: November 25, 2005

reviewed by Aaron Wilson

I was looking forward to reading this book, to improve my knowledge of the Linux Kernel and help to understand better (or at least begin to know where to look) to solve some of the issues that have occasionally cropped up when some of the hardware I have used doesn't behave. Having little knowledge of the Kernel internals it is often hard to understand what is going on.

I was pleasantly surprised that this book describes a relatively recent kernel (2.6.11), having expected kernel development to have leap massively forwards in the time taken to prepare and publish this book. I was also re-assured by the claim in about the book that all I would need to get stared is 'some skill in C programming language and perhaps some knowledge of an assembly language' – knowing that in the dim and distant past as part of a course I studied some assembly language. As this book is not an easy read, I would say that the pre-requisites are understated, and at times my lack of 80x86 assembly language at times made understanding – in spite of the good explanations in the text – hard work.

This brings me onto my next comment; this is not a book for the faint-hearted. I found it easy to read but hard to understand. Particularly as my knowledge of the architecture of the internals of a modern 80x86 PC is, to say the least sketchy. Also, I feel having studied a course in operating system design principles would be an advantage (although not a pre-requisite).

The authors encourage the reader to look at the source code while reading the text, and as I read this I thought it a good sentiment. However, while reading the book I felt no great need to rush and look at the code, the relevant extracts being presented. I also feel that at times the decision to work from the ground up – and hence at times need to use forward references at times makes understanding harder than it needs to be.

This is not to say that this is not a book worth reading; from the start of the book I found this an excellent and informative read. It is not a book that I feel can be taken in in one reading, and I want to return to it and read again so that I can glean even more understanding of the Linux

Kernel. I also think that this book does the job it sets out to do – that is to acquaint the reader with the inner workings of the Linux kernel. Recently, having attended a one-day tutorial on the Linux Kernel – which gave a great overview of this book but none of the detail – I think this is a hard task to do, as there are so many things that need to be covered to give some understanding. However, this book does it well and I would recommend reading it.

This book covers only the basics of the kernel – but as the authors themselves acknowledge, it would need more a library than a book to cover the whole topic. It does however cover the everything necessary to enable the reader to go and find their way around the kernel, and help them to understand what they find there.

Open Sources 2.0

Chris DiBona, Mark Stone and Danese Cooper (Eds)

O'Reilly Media
ISBN 0-596-00802-3
488pp.
£ 20.95
Published: October 14, 2005

reviewed by Roger Whittaker

This book is a follow-up to the original "Open Sources: Voices from the Open Source Revolution" which was published in January 1999, both in book form and on the web under licences allowing redistribution of the text.

The web version of that book is available at:
`http://www.oreilly.com/catalog/opensources/book/toc.html`

The original "Open Sources" contained chapters by Eric Raymond (two chapters), Richard Stallman, Linus Torvalds, Michael Tiemann, Bob Young, Larry Wall and Bruce Perens among others and appeared at a time of considerable general excitement about the ideas contained in it. The book was at least in part a collection of manifestoes and evangelism for those ideas, and most of the content was re-printed from other sources. Looking at it again now, there's a definite air of *bliss was it in that dawn to be alive* about it.

More than six years later, the new book is necessarily different in tone and content. Subtitled "The Continuing Evolution", it is a longer book, with 24 chapters in all, laid out in two sections: "Open Source: Competition and Evolution" and "Beyond Open Source: Collaboration and Community".

The chapters in the first section are broadly about software, while the second section is mostly concerned with Open Source software as merely one example of a wider change in human affairs, largely facilitated by the Internet, which has led to a realisation that collaborative working by the many can have powerful consequences.

Chris DiBona's chapter "Open Source and Proprietary Software Development" mostly covers familiar ground, but in a fresh and interesting way. He concludes with the story of a conversation he overheard in the mid-90s when a lawyer said "You know, if TCP/IP had been properly protected and patented, we could have rigged it so that every packet cost money; they really missed the boat on that one." DiBona's comment: "Where would the Internet be if this was true? I don't know, but I do know one thing: the Internet would not be running TCP/IP".

Jeremy Allison's chapter "A Tale of Two Standards" is perhaps the only 'technical' chapter in the whole book, comparing the POSIX standard with Win32. He looks particularly at how Microsoft's access control in the Win32 standard theoretically permits a high level of security, but in practice has failed because its complexity has meant that application writers have almost entirely ignored the possibilities that it affords. He also writes interestingly about how the release by Microsoft of Services for Unix provided a fully POSIX environment on Windows, and speculates about the motives behind this release.

There are three separate chapters looking at the progress of Free and Open Source software in different parts of the world, namely Europe, India and China.

A highlight of the second part of the book is Pamela Jones' chapter "Extending Open Source Principles beyond Software". This is a description of the history of Groklaw. She describes how Groklaw became a collaborative community with a common goal, and how the spirit of open collaboration helped it to succeed in its aims (of documenting and researching the SCO legal cases and related matters) far beyond her expectations. But she also describes in some detail and with not a little bitterness the fly in the ointment: the problems that she has had with trolls, astroturfers and undercover enemies trying to use Groklaw to undermine its aims.

Perhaps predictably there are also chapters about Wikipedia (by Larry Sanger) and about Slashdot (by Jeff Bates and Mark Stone). Sanger's Wikipedia article also goes into detail about the problems of governance in an open collaborative project: problems which led in part to his resignation from the project.

The Slashdot chapter goes into considerable detail about the governance and community of Slashdot, and ultimately comes to a conclusion that a community of this kind needs a "benevolent dictatorship" combined with systems which allow the best to come to the top: a design which has been inherent in Slashdot from the start, though to what extent it succeeds is possibly debatable. There's also (slightly surprisingly to me) a long discussion of the "Hellmouth" the discussions that went on on Slashdot immediately after the Columbine school massacre. Jon Katz's article "Why Kids Kill" from that time is also included as an appendix to the book.

Slightly further from our world, but on a topic of huge importance to humanity is Andrew Hessel's chapter "Open Source Biology". The parallels with the world of Open Source Software, both in terms of open versus closed "intellectual property" and methods of working are clear, and Hessel explains the history and the threats and challenges.

Doc Searles also has a chapter in the second section entitled "Making a New World". As usual for him, there are some interesting insights: in particular he examines the way in which through what he calls "collapsed distinctions" the arguments about free and open source software are over-simplified, obscuring important issues and making the discussion one-dimensional (as distinct from the literally two-dimensional diagrams which he uses to illustrate his argument).

This book does much more than simply restate familiar arguments. Parts of it are thought provoking and will repay reading and re-reading.

Linux Multimedia Hacks

Kyle Rankin

O'Reilly Media
ISBN 0-596-10076-0
330pp.
£ 20.95
Published: November 29, 2005

reviewed by Roger Whittaker

The "Hacks" format is particularly well-suited to this title which includes a good deal of practical information about disparate subjects, linked only by the work "multimedia".

There are five sections: Images (12 hacks), Audio (34), Video (26), Broadcast Media (16) and

Web (12), making up a round 100. Apart from the last section (Web), which is a curious mixture of oddments, the sections proceed in what for me at least was order of familiarity, with the Images and Audio sections covering mostly things that I have discovered for myself over the years, but much of the material in the Video and Broadcast Media sections was new to me and will make a useful reference in future.

In the Images section there is a useful discussion of the use of the `convert` command line utility from ImageMagick, with example scripts. There is discussion of the various tools available for use with digital cameras, and a fairly lengthy look at f-spot. The last hack in the section is about creating screen capture movies as animated GIFs, by using a set of scripts which the author has created for the purpose.

The Audio section includes coverage of the various music players which are available for Linux, and how to add mp3 functionality to your Fedora setup (but equivalent information is not given for other distributions, unfortunately). CD ripping and encoding tools are discussed, including something that I only discovered purely by accident: the fact the KDE's Konqueror browser will rip audio CDs to mp3, ogg or FLAC formats. Using an iPod with Linux is discussed, but at a low level, and oddly Banshee is not mentioned, which currently seems to be the best way to do this. Various utilities for tagging mp3s are discussed, including picard. One of the oddest hacks in the book concerns playing music files backwards (to hear whether or not the famous words "Paul is dead" were ever spoken).

The Video section discusses mplayer in detail, as might be expected, and how to get DVDs to play in Linux. Oddly there is very little discussion, either here or in the Audio section of the legal issues which make all this so fraught with difficulty, but there is informative technical detail about how to get things working. Interesting hacks (which I have not tested) include converting dual layer DVDs to single layer, and converting DVD movies to self-booting movies on CD. The oddity in this section is "watching videos in ASCII art": I doubt if there will be many takers for that one.

The Broadcast Media section discusses watching TV on your computer, and MythTV as a digital video recorder. Of more interest to me at least were the hacks about ripping audio streams to files and browsing streaming radio stations.

The final section "Web" includes information about getting various browser plugins to work properly, but also a section on Asterisk (probably too short and slightly out of place here) and one on w3m (not really anything to do with multimedia).

Overall this is an interesting book that brings together useful information about the topics covered, and thus makes a useful reference.

Information Dashboard Design

Stephen Few

O'Reilly Media

ISBN 0-596-10016-7

223pp.

£ 24.99

Published: February 10, 2006

reviewed by Sam Smith

"Information Dashboard Design – The Effective Visual Communication of Data" seems like an odd book for us to review; it isn't.

Much of what we spend our time doing is waiting for things that are about to go wrong and fixing them; hopefully before anyone notices. An Information Dashboard is an at-a-glance thing which tells you which part of the building is on fire (58), and whether or not it should be.

The most common example is the Nagios "network outages" and "network status" displays. A quick, easy, obvious way to see that everything is working just fine. While Nagios is a good example of dashboard design, there are many, many, bad examples, and there are areas where lives could be made simpler with a few perl scripts and a simple web page.

The book talks about what is useful, what works, and what doesn't. While much of it may be common sense when you can sit and tweak it until it's just right; when you are building these things for others with different expertise to see, the outcome can be very different, and can increase or decrease questions.

While the comments and examples within the book are mainly aimed at the quick presentation of status information; they are equally applicable to many varied forms of presentation. whatever you're drawing graphs or charts of, there are common mistake highlighted as things which should be avoided, along with why. Much as there is with the use and, more often, abuse of design in many documents.

One of the most valuable parts of the book is that there are many many illustrations of the concepts being described. Well thought out and detailed, points are well made, with strong advice on how to avoid making the same mistakes, and tips on doing good things better.

While not a book that everyone will have any interest in, if you spend time dealing with the meaning of graphic information, status displays or reports, you might find it useful to have a brief skim before handing it to the person who created what you're looking at.

Credits

We are grateful to Dru Lavigne for allowing us to reprint her report on the Durham Conference which was originally published on her blog at:
`http://blogs.ittoolbox.com/unix/bsd`

The article by Dave Cross was first published in Linux Format and is reproduced here by kind permission of the author.

The article by Michael Lucas appeared first in the compilation "BSD Success Stories", a pamphlet put together by Dru Lavigne for O'Reilly:
`http://linux.oreilly.com/news/bsd_ss.pdf`

Contributors

Dave Cross runs Magnum Solutions Ltd, an Open Source consultancy based in London. He is a well-known member of the Perl community and a frequent speaker at Perl and Open Source conferences. Since 2002, Dave has been the Perl Mongers Group Co-ordinator for the Perl Foundation. He is the author of "Data Munging with Perl" (Manning, 2001) and a co-author of "Perl Template Toolkit" (O'Reilly, 2003). His company and personal web pages are at
`http://mag-sol.com/`
`http://dave.org.uk/`

Robert Easthope is a Systems Adminstrator at UKERNA (the United Kingdom Education and Research Network Association).

Gavin Inglis works in Technical Infrastructure for the EDINA National Data Centre. His interests include punk music, Egyptian mythology and complicated diagrams.

Dru Lavigne is a network and systems administrator, IT instructor and author. She has over a decade of experience administering and teaching Netware, Microsoft, Cisco, Checkpoint, SCO, Solaris, Linux and BSD systems. A prolific author, she pens the popular FreeBSD Basics column for O'Reilly and is author of BSD Hacks. She is currently the acting chair of BSD Certification Group Inc., a non-profit organization with a mission to create a standard for certifying BSD system administrators.

Graham Lee is a UNIX Systems Manager in the University of Oxford Physics department. However, as the only local sysadmin to have needed use of CP/M this millennium, he feels "Information Archaeologist" would be more appropriate.

Michael Lucas lives in a haunted house in Detroit, Michigan with his wife Liz, assorted rodents, and a multitude of fish. He has been a pet wrangler, a librarian, a security consultant, and now works as a network engineer and systems administrator with the Great Lakes Technologies Group. He's the author of Absolute BSD and Absolute OpenBSD, and is currently preparing a book about NetBSD.

Lindsay Marshall developed the Newcastle Connection distributed UNIX software and created the first Internet cemetery. He is a Senior Lecturer in the School of Computing Science at the University of Newcastle upon Tyne. He also runs the RISKS digest website and the Bifurcated Rivets weblog.

Greg Matthews is a Senior Unix and Linux administrator for the Natural Environment Research Council.

Ray Miller is a director of UKUUG and Chairman of UKUUG Council. He works as a Unix Systems Programmer at the University of Oxford, where he manages the Systems Development and Support team in the University's Computing Services.

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 (Fibre-optic Industry Association) that she also runs from the Manor House office.

Mike Smith works in the Chief Technology Office of a major European listed outsourcing company, setting technical strategy and working with hardware and software vendors to bring innovative solutions to its clients. He has over 15 years in the industry, including mid-range technical support roles and has experience with AIX, Dynix/ptx, HP-UX, Irix, Reliant UNIX, Solaris and of course Linux.

Sam Smith has been on UKUUG Council for 3 years and is currently the treasurer, with many random interests in addition to OpenBSD and Mac OS X. He's also active in the UK Online Democracy group mySociety and the Manchester Universities' Gilbert and Sullivan Society.

Paul Waring is a student at the University of Manchester and is currently in the final year of a BSc in Computer Science. When not studying or at the gym, he works as a freelance web and software developer to help pay the beer bills.

Roger Whittaker is a UKUUG Council member and works for Novell Technical Services as a Linux Support Engineer.

Alain Williams is a Council member of UKUUG and works as an independent Unix and Linux consultant, running Parliament Hill Computers Ltd.
`http://www.phcomp.co.uk/`

Aaron Wilson is a Unix Systems Programmer in the Systems Support and Development group at Oxford University Computing Services. Outside of work he enjoys watching a rubgy (suppoting London Irish), reading and going to the theatre amongst other interests.

Contacts

Ray Miller
Council Chairman; Events; Newsletter
Oxford
Tel: 01865 283284
`ray.miller@ukuug.org`

Mike Banahan
Ely
`mike.banahan@ukuug.org`

Sam Smith
UKUUG Treasurer; Website
Manchester
`sam.smith@ukuug.org`

Alain Williams
Watford
`alain.williams@ukuug.org`

Roger Whittaker
Newsletter
London
`roger.whittaker@ukuug.org`

John M Collins
Welwyn Garden City
`john.collins@ukuug.org`

Dean Wilson
London
`dean.wilson@ukuug.org`

Newsletter
`newsletter@ukuug.org`

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