flossUK

news@UK

The newsletter of FLOSS UK, the new name for the UK's oldest Open Systems User Group, UKUUG

Published electronically at http://www.flossuk.org/Newsletter

Volume 23, Number 3 **ISSN 0965-9412** **September 2014**

Contents

From the Secretariat

Jane Morrison

The AGM this year will be held on Thursday 25th September at the Ambassadors Hotel, 12 Upper Woburn Place, London WC1H 0HX, starting at 6:15 p.m. Full details including the Agenda etc. were sent to all paid members at the end of August.

We hope you will be able to attend. Full details can also be found on our web site.

The un-conference being planned for Saturday 25th October in London is still in the planning stages but we shall provide you with full information as soon as we have it.

We are running further Perl tutorials (Intermediate and Advanced) in London on 11th, 12th 13th & 14th November. These are always very popular and early booking is advised, see full details including booking information on the enclosed flyer.

Spring 2015 event 'FLOSS UK DEVOPS' - York, 24th, 25th & 26th March, please see the Call for Papers enclosed with this issue. Why not submit a proposal? Closing date for abstracts is 14th November.

The full list of speakers and on-line booking for this event should be available early in December.

The next Newsletter will be the December issue and the copy date is 14th November. As usual any articles, letters etc. can be sent for inclusion to: **`newsletter@flossuk.org`**

Chairman's Report

Kimball Johnson

Events

As always Summer is quiet, but work is progressing on the Spring Conference 2015 in York, more details will come out soon for that. We have also had the Barcamp Birmingham since the last newsletter, and while I was not able to attend myself, I have heard that it was a good day, with plenty of interesting talks, my thanks goes out to Quentin for running the event, and to everyone who came.

Available for booking now is the ever popular Perl Training by David Cross, in November. Places are limited so book soon to avoid disappointment. We are also finalising the plans for FLOSS UK's first ever Dynamic Languages conference, in association with Shadowcat Systems. This will be a one day event in Manchester, held on a Saturday in November. The call for papers and ticket sales will start shortly, please watch the website and mailing list. Please also contact me (**`kimball.johnson@flossuk.org`**) if you would like to sponsor the event.

I would also like to encourage anyone with ideas for courses they would like to be run to email the office with their ideas.

Resignation of Chairman

As I announced in the June Newsletter, I am stepping down as Chair from the AGM. I will remain on Council to provide some handover to the new Chair. I would like to take this

opportunity to publicly thank the rest of Council, Jane and the other volunteers that keep FLOSS UK running.

Support for local user groups

The budget is still available for FLOSS UK to assist local user groups by helping them to obtain speakers for their events and assist with travel expenses. In addition we have a small budget for assisting with projects that would benefit the Free Software or Free Hardware communities in some way.

If you have an idea and wish some support, please contact Jane on **office@flossuk.org** and it will be discussed by Council.

Get Involved

FLOSS UK exists to serve its members and we are always on the lookout for people who are keen to get involved in any capacity, whether that be through volunteering to help with organising events, writing newsletter articles, or entirely new activities which haven't been tried before. If you would like to help out in any capacity please do get in touch via **office@flossuk.org**.

From the Editor

Paul Waring

Eagle-eyed readers will have noticed a change in editorship as Roger Whittaker passed on the baton to me from the September issue. Many thanks are due to Roger who has edited the newsletter since before I became involved in FLOSS UK.

There will be no immediate changes to the newsletter style or content whilst I get to grips with the editorship, but I will be looking at the newsletter process in more detail towards the end of the year. Please feel free to email **newsletter@flossuk.org** with any ideas or suggestions.

We are always looking for newsletter content across the following categories:

- Articles: 500-1000 words on a technical subject. Longer articles can be split over multiple issues if required. Past issues have included topics as diverse as Arduinos, ZFS and software patents.
- Event reports: Both FLOSS UK events and those which are of interest to our membership in general, e.g. Oggcamp.
- Book reviews: 250-500 words on a technical book. We can now provide review copies of O'Reilly books in electronic (PDF, ePub, Mobi) and print formats. Subscribe to the mailing list at **http://lists.ukuug.org/mailman/listinfo/books** to find out which books are available for review.

If you have any ideas for newsletter items, please do get in touch via email: **newsletter@flossuk.org**.

Birmingham Bar Camp

Quentin Wright

Report on the Birmingham Barcamp held on 7th June 2014 at the Studio Cannon Street

About 25 people turned out at the early hour of 9am for what is now becoming an annual event in Central Birmingham. Attendees were mostly a goodly mix of LUG members, variously from Wolverhampton, Birmingham, Coventry and Rugby. The day started with half hour of networking and consumption of Danish pastries and a fair amount of coffee.

The traditional barcamp approach was made, passing out post-it pads and pens and asking for topics on which attendees could give a short presentation or lead a discussion. Additionally topics could be proposed where the requester would like information. When all requests had been received the proposer was asked to briefly describe the topic and a show of hands was made to gauge interest. We then worked through as many of the items as possible, starting with the most popular.

Although the format of the day is flexible and in the hands of the attendees we ended up running with a single group and time slots of half an hour maximum for presentation and discussions.

Thanks must be given to Keith White and Alex Willmer for helping run the event as well as tidying up at the end!

What follows is a brief summary of the content of each topic covered.

NSA/Security Stuff - Tony Brookes

Tony started with a reference to the historical perspective, talking about Francis Walsingham who was the 'Spycatcher' of the Elizabethan era. Walsingham arranged to entrap Mary Queen of Scots by intercepting and deciphering secret letters. Modern day encryption is weak, so for example mobile phone encryption is possibly purposefully weaker than it might be. Of course there are other methods of using technology to gain information. Again using the example of mobile communication, data derived from cell logs can be used to confirm location, and this was material in the Soham enquiry of a few years ago. The State will always use the technology.

Note that the amount of information that can be retrieved from a firewall is scary! However you can reassure yourself that you're pretty safe because of the volumes involved.

Some of the areas covered in discussion:

- There's a difference between the law of man and the law of nature.
- Should we be encrypting more data? Of course as soon as you are seen to be encrypting material you are likely to be the subject of suspicion.
- Tor is a very important technology.
- Heartbleed demonstrates that all code is inherently dangerous even if it's open source.

Oggcamp - Mark Johnson

Mark described the evolution of Oggcamp as an Offshoot of LUGRadio and maintains much the same anarchic spirit. Technology is used to support the organisation and in particular extensive use is made of Campfire Manager. This year's event will be held in Oxford on the 4th/5th October 2014 at The Oxford Hotel and will include an OpenHardware Jam and a Picnic for Geeks. It's free and more information can be found at `oggcamp.org`.

Wifi Captive Portals - Jon Knight

Everything you need to create a captive portal can all be done with iptables and ebtables, however as an alternative there's a BSD-based application called Chillispot that will do the

job as well.

Open Source Geophysics and Mapping - Tony Brookes

Commercial products are 5 to 10k but there are open source packages that are of excellent quality. Tony gave us a whistle stop tour of Q-Gis showing the overlaying of 4 maps for architectural purposes. He then moved on to show us OpenStreetMap.

Old maps OS Maps for 23 Counties can be acquired (1901). Historically there was no convention that N was North which makes older maps interesting! The Unpaper application was mentioned. The discussion then Moved on to Geophysics. There's lots of information available about equipment and satellite systems although the jargon can be problematic. Tony is aiming for 1cm accuracy. At the moment working with a GPS chip, an Arduino and a Linux box he is achieving 1m accuracy. The RTKlib library has proven to be useful. There are four different constellations, that's to say GPS systems and developers are trying to use them all. Work is going on with small drones and high accuracy GPS, also kites. Recent innovations involve the use of 3D Modelling and the infrared iPhone unit.

Self hosted Cloud - Mark Johnson

In his own computing activity Mark is seeking to use FLOSS software and doesn't want to rely on other people and organisations. Typical areas where cloud hosting is useful or desirable is for file storage, sync and sharing, music storage, online/collaborative document editing, calendering and social networking.

For file storage it's possible to replace Dropbox or Google Drive with OwnCloud - this has desktop syncing but this fails with large number of files. It's possible to mount files using Webdav. There's also BTSync but this is still proprietary and the developers not responsive. Another open source alternative is Syncthing.

For music streaming to replace Spotify or Google Music there's OwnCloud's app although this is rather flakey. There's Ampache which is a module for Apache which streams using HTTP.

For documents there still seems to be no full replacement for Google Drive although Etherpad Lite is written in node.js. This is a multiplayer notepad rather than a multiplayer Word and no-one has got the collaborative aspect working. There is an application called sheetnode which is good for spreadsheets.

Ownclode 6 shows promise and does calendaring and address book very well.

For social networking it's difficult to replacing Facebook and Twitter. Maybe Diaspora Status.net (RIP) and pump.io show the way.

As a couple of general points, for encryption one can get certificates from **`startssl.com`** and for DNS **`freedns.afraid.org`** + CNAME can be used.

Software Defined Radio - Ron Wellsted

Software Defined Radio (SDR) refers to the replacement of traditional radio hardware by equivalent modelling in software. Fast Fourier Transforms are used to represent signals, so for example a square wave can be produced by the sum of odd harmonics. To implement an SDR either direct conversion or direct sampling receivers can be used. Direct conversion uses sample rates of 48k, 96k and 192k samples per second (SPS). Typically when using direct sampling it is necessary to sample at about 8 times the frequency. A typical digital signal processor chip costs about £30.

Examples of typical receivers are the Softrock, Hunter SDR and RTL2832U/Elonics E4000. Commercial Transceivers include the Elekcraft and the Baofeng UV-5R. Software environments that are supported include Windows, Linux/OS X, iOS, and Android. Applications for Windows Software are HDSDR, SDR, PowerSDR and for Linux Quisk, Lysdr and GNURadio. For iOS, iSDR and for Android SDRTouch.

PostgreSQL is Cool - Chris Ellis

The philosophy of the PostgreSQL team is 'Do it once and get it right'. This has been particularly the case with the recent implementation of replication.

Many features of PostgreSQL were mentioned:

- PL/Proxy: A procedural language which allows scaling by partitioning a database across servers.
- Recursive queries: great for working with hierarchies.
- Range types: put people into timeslots, can select integer and time ranges.
- Array types: can be used to do away with link tables.

Samba 4 and Active Directory - Liban Hannan

Active Directory is a Microsoft product. The Samba project has been working to produce Samba 4 which provides Active Directory support. The schema can be extended and there is basic support for Exchange. It can also be used with Mac OpenDirectory.

Replication is difficult to use with Samba 4 - you have to write your own rsync scripts to cope with Windows configuration templates. Samba 4 does not support Forests or multiple domains.

Authentication works. Roaming profiles are available for Windows. There can be some issues with UIDs.

If you are intending to implement Samba 4 with Active Directory bear in mind that problems are resolved quickly by the team.

Podcasting - Pete Cannon

Pete is the co-host of The Dick Turpin Road Show, a humorous fortnightly podcast looking at Linux. The show is done very much for fun and is aimed at an audience of 50, but now up to a hard core of 300 listeners, plus another 200 more occasional listeners.

It's possible to record on virtually any device. Originally Skype recorder was used but it isn't possible to adjust the sound levels. Now to produce a podcast Audacity is used to record locally. There can be issues of bandwidth with the hosting services as volume increases. The voice element is mono and now the show has started to have 'bumper music' (music between sessions) this music has recently been made stereo. Don't do a live show as this will just produce foul ups.

They had thought about using video but there are issues of production and editing. No-one seems to watch videos. You will get an amazing number of complaints. 'It's too loud'. 'My player doesn't work'. Initially it took days to do editing as there's a desire to be perfect. Keeping to a time limit can be difficult. The best way is just to watch the clock. CherryTree is used for running notes. As a final tip: to put an item on YouTube - put a picture up and then

add the sound.

Bitcoin - Alex Willmer

Bitcoin - peer-to-peer, as few intermediaries as possible. You can run your own email server, and your own wallet. It's possible use an on-line wallet. A wallet has a collection of private keys. Each key has an associated public key and an address. The private key allows you to spend money. Software has evolved and it's now probably more convenient to have your Bitcoin wallet on your phone. Can request that someone makes a payment, the client software shows a QR code which can be scanned by the recipient who signs the transaction. The transactions are placed on the network. Miners verify the transaction and record it in a block. Can wait for 5 more blocks to be added after which the transaction can be considered as confirmed.

Miners compete to verify and complete blocks by solving 'proof of works'. The miner who completes a block first wins. They receive a block reward and transaction fees. The prize incentivizess miners to keep the Bitcoin network running. Cooperating is more profitable. The blockchain is a public record of all transactions. If a branch occurs then the longest chain wins. On average a block is added every 10 minutes. By 2011 6.5 million BTC have been generated, now 12.8 million have been generated. 21 million is all the BTC that will ever exist. At this rate the limit will be reached by 2033!

Scratch a Hardware Itch

Les Pounder

Hardware hacking has seen a surge of interest in recent years thanks in part to development platforms such as the Arduino and Raspberry Pi, and to a lesser degree many other prototyping platforms. The ever popular Raspberry Pi has found popularity with schools via the Raspberry Pi Foundation's Education team, and with hackers thanks to its ease of use. The Raspberry Pi is enabling a new generation to experience the delights of finding out how things work. But for this issue let us change our focus from the world of Raspberry Pi and micro computers and look to the other popular choice, the Arduino micro controller.

The Arduino was created in 2005 by a team of artists, developers and researchers in Interaction Design Institute Ivrea, Italy, as a replacement for an earlier prototyping tool called BASIC Stamp. The goal of the Arduino project was to enable artists to easily use machinery or electronics in their installations via a board that was significantly cheaper than previous platforms, in fact they likened the price to a good meal and glass of wine.

The most popular Arduino board is the UNO which retails for around £30 and is the most versatile board at its price point. Powered by an ATMEL ATmega328p processor, the UNO runs at 16mhz and only has 32kb of flash memory. Now this might not seem like a lot, but it more than enough power for many different projects.

The Arduino project is an open source project, and their code and hardware is available for inspection. This openness has enabled different groups and businesses to create their own Arduino compatible boards, under the restriction that the board must not carry the Arduino branding. These clone boards are generally of good quality, but I would always recommend purchasing an original Arduino board if you are new to the community.

The Arduino boards are programmed using the Arduino development environment and the software is available for Windows, Mac and Linux. The language to program the Arduino

is a high level language based upon Processing, a language used by artists to sketch ideas for electronic based projects, hence this is why projects in Arduino are commonly called ‘sketches’.

Educational uses of Arduino have yet to find the same success as the Raspberry Pi, but there is one software project based on the Arduino that is looking to introduce Arduino using the popular visual language Scratch. Scratch for Arduino often abbreviated to S4A[1] is a software package that comes in two parts. The first is a version of the Scratch application modified to work with the Arduino that is installed on a PC running Windows, OSX or Linux. The second part is a custom Arduino sketch that should be uploaded to your Arduino using the Arduino IDE. Ensuring that your Arduino is connected to the same computer that is running the modified version of Scratch.

S4A uses a simple interface, almost identical to the typical Scratch interface, that is split in to three main columnar sections. A palettes section where you can find many different types of blocks grouped into palettes, with palettes for control, movement, sound and many more. In the centre of the screen is where the blocks are dragged into and this programs the Arduino. Lastly there is a stage and sprite area, that shows any outputs such as sensor data.

Using S4A you can simply drag blocks of code from the palettes into a sequence in much the same way as building using Lego. The blocks will link together and form a sequence in much the same was as procedural code does. Inside the motion palette there are special blocks for using the analogue / digital pins of an Arduino UNO. Your Arduino UNO will need to plugged into your computer and an LED inserted with its long leg into pin 13 and the short leg into GND.

A simple test is to blink an LED attached to pin 13, use the block ‘When green flag is clicked’ from the Control palette, next use a forever loop, also from the Control palette, to repeat the sequence that we will now create. Our first block of code inside the loop is ‘wait 1 second’ found in the Control palette, with that done change to the Motion palette and select ‘Digital 13 On’, if you can’t see the number 13, use the drop down menu to find the correct pin. Move back to the Control palette and insert another ‘Wait 1 second’ to allow the LED to receive power for 1 second, then move back to the Motion palette and select ‘Digital 13 Off’. That’s it! Click on the green flag and your LED should start flashing.

Programming real world projects is a much more entertaining and educational experience with children and adults alike building really great projects together.

[1] **https://github.com/lesp/FLOSSUK_Scratch**

Flask Web Development

Miguel Grinberg
O'Reilly Media
ISBN: 978-1-4493-7262-0
258pp.
$39.99
Published: May 2014

reviewed by Andrew Richards

This book aims to explain the use of Flask with Python by gradually building up a sample web application using Flask and various extensions. The prior experience of the reader will have a significant bearing on how well it succeeds in this aim. In my case, as someone without experience of similar products for Python or other languages, I ended up struggling. For those who have prior experience building dynamic websites with other tools and languages, this book should be fine.

Each chapter in the book sees enhancements to 2 sample applications, with these enhancements being discussed as they are deployed. These 2 sample applications are available as a repo from Github. Git is used to checkout the applications at various stages of their development, which proves very helpful since it is possible to skip backwards and forwards between versions (chapters); existing git knowledge will maximise your benefit here.

The text proceeds at a brisk pace - too brisk for me as it turned out - but at least initially this works well enough, and the reader is quickly brought to the point of producing 'Hello world' and then 'Hello your_name_here' Flask-generated web pages which are then steadily enhanced to add various functionality and styling. The general approach is to add appropriate Flask extensions to tackle each new issue, often with surprisingly little additional programming in between. Whilst the author is careful to include topics such as password security in an application and sanitising user input, the rapid pace of the book leaves some individual topics insufficiently explained, including to my mind the explanation of the 'core' of Flask: Whilst for example Flask decorators or helper functions are covered throughout the book, there is no chapter or appendix summarising these and other core aspects of Flask.

For me, the rapid pace often meant that I was just starting to grasp a given topic before the book moved onto the next topic. This had the effect of leaving me feeling that I had only a superficial understanding of significant parts of the content. The book wasn't quite enough for me, but developers who already have experience building dynamic websites with other tools and languages may well find the book a much better fit. Note also that it is well worth visiting the author's blog where a couple of tutorial videos are included which go some way to addressing my concerns around the book's pace and coverage.

The book uses a specific set of tools and extensions, and I think there is insufficient exploration of alternatives. There is a half-page near the end listing a few more Flask extensions, each with a scant explanatory sentence that is helpful, but I feel this could be significantly expanded; it would also have been interesting to have pointers throughout the text mentioning alternatives to the specific tools and extensions chosen by the author as well as some discussion of his choices.

That said, the breadth of the subjects covered is impressive, aiming to address many of the issues in building real-world web applications - including back-end databases, email, application structure (directory layout), authentication, RESTful APIs, [unit] testing and deployment. I hope the author will produce a second edition that's got maybe 350 pages compared to the

current 250 to expand on the items where he's been overly brief, summarise some of the key features of Flask itself and improve the index (too brief).

Overall then, your mileage will vary. If you lack modern web programming experience then you may struggle as I have, but with persistence you should still get what you need from this book; if you already have a good sense of the general structure and approach to web programming as well as being comfortable with another web programming environment I expect you'll find this book explains the Flask way of doing things well.

OpenStack Operations Guide

Tom Fifield, Diane Fleming, Anne Gentle, Lorin Hochstein, Jonathan Proulx, Everett Toews, Joe Topjian
O'Reilly Media
ISBN: 978-1-4919-4695-4
330pp.
$39.99
Published: May 2014

reviewed by Paul Thompson

Given the subtitle 'Set Up and Manage Your OpenStack Cloud', I expected a little more on the actual installation of OpenStack, but then I suppose the pain of a cloud install is usually over in a few weeks; it's the following months and years that is the real challenge. This book mainly focuses on the operations of an already-running cloud.

Being based on the almost-current release of OpenStack, Havana, and assuming that the underlying hosts' OS are either Ubuntu or a RHEL derivative seems to have been an attempt to cover what most readers are using in their proof-of-concept implementations, but for me it was disappointing that SUSE Linux Enterprise wasn't suggested.

The reference environment used for all examples throughout the book is described in some detail in the early chapters. It is a simple, yet fully-functional architecture capable of supporting a small production deployment, and care is taken to explain the rationale for the choices of each of the major components and the features selected – such as RabbitMQ over Qpid for the AMQP broker and MySQL over SQLite – especially useful where they deviate from the OpenStack defaults. The basic architecture essentially consists of multiple Compute nodes, a cloud controller, an external NFS storage server for instance storage and an OpenStack Block Storage server for volume storage. NTP synchronises time on all nodes and FlatDHCPManager in multi-host mode is used for the networking.

Those areas of OpenStack that are perhaps the most difficult to change post-installation are considered in greatest detail: disk partitioning and networking configuration. Description of fundamentals such as RAID should have been omitted entirely, and the assumption made that the target reader is already familiar.

Despite OpenStack being designed to be massively scalable, and hence have the ability to be widely distributed, a more centralised model is assumed, with the concept of a 'cloud controller' – but this conceptual simplification serves as a useful tool to help explain the OpenStack operations throughout the book. Due consideration is made however to the generally accepted best practice of separating services onto dispersed physical hosts; a list of common deployment scenarios is presented together with their justifications.

Attention is drawn to some important resilience and scalability aspects. Clustering the SQL database (used by Compute for stateful information) is recommended for failure tolerance. The message queue must be made fault-tolerant too, but it is rightly noted that clustered message queues are a pain point for many OpenStack deployments: RabbitMQ's native clustering support has scaling issues and Qpid (used by Red Hat derivatives) does not have native clustering so a supplemental service such as Pacemaker or Corosync is required. Nova-conductor (new to OpenStack) – which eliminates the need for all nova-compute services to have direct access to the database – is horizontally scalable so making it fault tolerant and highly available simply requires more instances to be launched across multiple servers.

The perennial question of whether to virtualise some services is addressed: the advice given here being to avoid virtualising nova-compute, swift-proxy and swift-object, however control servers can be – though more instances of the service may be required to offset the inevitable performance penalties.

A short chapter deals with the choices to be considered when building compute nodes: mainly about resources required to run the workloads, but also covered are the benefits and drawbacks of shared- versus non-shared file systems, particularly with respect to performance, scaling and live migrations. Scaling and storage are covered in more detail in later chapters where a table clearly explains the different OpenStack storage concepts: ephemeral, block and object – and which back-end commodity storage technology to choose for particular cases.

Networking usage in a cloud differs significantly from traditional network deployments and has the potential to be disruptive to both connectivity and policies. Chapter 7 offers suggestions on segregation such as creating a separate 'management network' and creating private networks for communication between internal components of OpenStack, such as the message queue and Compute. I found particularly useful the discussion on the strengths and weaknesses of OpenStack Compute's nova-network predefined deployment models (Flat, FlatDHCP, VlanManager, etc).

The remainder of the book assumes an already-implemented cloud and focuses on the day-to-day operations. While I found it an interesting read, and was able to pick up useful suggestions, this section is unlikely to be useful for an administrator to read in advance of implementing their cloud. It would be more appropriate to refer to its topics for hints on performing specific tasks as the need arises or for troubleshooting when an unexpected event occurs.

Instructions are given for such necessary tasks as installing the API-based CLI tools and obtaining the credentials for use with command-line clients. Suggested troubleshooting techniques include inspecting API calls and examining JSON responses using cURL and jq.

Discovering an OpenStack cloud's architecture can be daunting, but is important for anyone trying to troubleshoot on behalf of a customer or for an administrator taking ownership of an already implemented cloud. This book presents several techniques for getting an overview, and breaks down each area into easily-digestible parts such as exploring the servers / services and inspecting the network configuration.

Two chapters are dedicated to the fairly mundane tasks of managing tenants and user-facing operations – such as quotas; security groups; live snapshots, etc. They aren't quite step-by-step instructions, but are enough to guide a novice through the processes.

Chapters 12 and 13 are probably the most useful for anyone new to troubleshooting a cloud. Included is a fairly thorough description of techniques for capturing packets on internal interfaces and touches on the difficulties of working with internal VLANs. Debugging individual components leads on to the topic of logging and monitoring – which is not particularly mature

in OpenStack in that many services still have their own distinct approaches and formats.

The last instructional section is around backup policies, with the subsequent few chapters being discussion about topics such as the upstream OpenStack community and upgrades.

I managed to glean some useful morsels from this book, but felt it was too verbose. It seems to have been aiming for too broad a target audience: newcomers through to the experienced. I will, however, keep it in mind as a reference for those not-so-routine tasks that are bound to occur in an OpenStack cloud.

IPv6 Essentials, 3rd Edition

Silvia Hagen
O'Reilly Media
ISBN: 978-1-4493-1921-2
412pp.
$39.99
Published: June 2014

reviewed by Paul Waring

Despite IPv6 being defined as far back as 1996, the slow adoption of the 'next generation' protocol has resulted in a third edition of this book which aims to cover all aspects of IPv6.

The first chapter discusses why IPv6 came about and why you should deploy it, focussing on extended address space (perhaps the most pressing reason in the short term), autoconfiguration and support for extensions. The author also deals with some of the common excuses for not deploying IPv6 - these pages alone are worth printing out and pinning up on the office walls of your network manager.

Chapter two is devoted entirely to addressing, with clear descriptions of the three address categories: unicast, multicast and anycast. All the special addresses are covered, along with IPv4-mapped addresses. This is followed up with a chapter which goes into substantial detail about the protocol's structure, including all the header fields, although much of this should be transparent and only of interest when debugging or developing applications.

Chapter four launches into ICMPv6, covering Router and Neighbour Solicitation and Advertisement. This is one of the longest chapters in the book and I felt completely out of my depth here, having not really needed to do much with ICMP on IPv4 networks. The next chapter is similarly detailed, covering Layer 2 support for IPv6 (i.e. the various 'on the wire(less)' standards). One thing is definitely clear at this point: no one can accuse the author of failing to go into sufficient depth on any topic!

Security - one of my favourite topics and something which is often overlooked in technical books - gets full coverage in chapter six. Much of this advice applies to networks in general, but there is also some coverage of IPv6-specific security issues and advice on how to monitor this traffic even if you think you are running an IPv4-only network.

Chapter seven is possibly of the most immediate interest, as it covers the various transition mechanisms which are available - essential reading given that IPv6 and IPv4 networks will have to co-exist for the foreseeable future. Dual-stack, tunneling and translation are discussed, with pros and cons and some helpful diagrams.

Chapter eight is likely to only be of interest to those working in telecommunications or running cross-site networks, as it covers the implications of mobile IPv6. The author then wraps up by summarising all of the previous chapters and outlining ways to plan for IPv6 adoption. Finally, there are two useful appendices, listing all the relevant RFCs and some further recommended reading.

Overall, this is probably the most comprehensive IPv6 resource I have come across, with substantial amounts of detail on the core topics and references to all the relevant RFCs. You do need a deep understanding of networking to get the most out of the book though, and only the first few chapters will be relevant if you just want to play around with IPv6 on your home network.

As an aside, this was also the first time I have tried an O'Reilly book in electronic format. In the past I have preferred a paper copy, but the DRM-free PDF (ePub and Mobi formats are also available) displayed almost perfectly on my iPad (a few diagrams had odd colours) and saves precious space on my bookshelf.

Getting Started with OpenShift

Steve Pousty, Katie Miller
O'Reilly Media
ISBN: 978-1-4919-0047-5
104pp.
$19.99
Published: May 2014

reviewed by Alain Williams

Summary: I wish that there were more books like this. It did not tell me everything about OpenShift, but hand held me through initial set up and creation of a simple app. This getting started is often the most difficult part of finding out about something new, especially at a time when you are not sure if you really want to use it, so is it worth the bother?

This book coming up was nice timing since RedHat had talked a lot about OpenShift at a road show that I had been to a month before. I was also exploring the newly available CentOS 7 (in a virtual machine), which I used to play with OpenShift. If you do not have a RedHat/CentOS environment available you might find things a little harder.

What is OpenShift?

It is RedHat's PaaS (Platform as a Service). This is a mechanism for easily creating and maintaining the operating system environment (Platform) to support an application in the cloud, typically a web application. You do not need to worry about how the operating system works or is installed and updated, where and what the hardware is. There are mechanisms for automatically scaling the platform to cope with workload fluctuations.

A few OpenShift buzz words that might help:

- An application (app) is typically a web accessible service. The book's example app gave you a random insult each time you visited its web page.
- A gear is the container in which the app runs. The word 'container' should be understood as the Linux kernel lightweight virtualisation mechanism for separating several running environments on one (probably virtual) machine.

- A cartridge is a component that you will use when building an application, eg: Python, PHP, MySQL. Your gear is a Linux system complete with Apache, so you do not need to install these basics.

The book starts by guiding you though signing up for an account that allows you to create free apps and also installing some necessary command line tools.

The book shows the command lines that I needed to type to achieve the next step. There is a web interface that I eschewed. Most of the work was done on my machine, making local changes and then pushing them with git. I could ssh to the gear, but that was mainly to poke around and see what was running.

Next is writing the first version of the app (the code I stole from the O'Reilly git repository) and testing it. A later chapter shows how to use a database that contains Shakespearean insults. The example application was written in Python, which is not a language that I know a lot about, that was not a problem.

Other chapters show how to: connect via ssh; change the git configuration; look at log files; monitor the gear; back it up. There is a chapter on managing access control for the various members of a team, each of who might have different needs.

All the code is available at `github.com` - great so that I did not waste time typing. Also good as some bugs in the book were fixed.

The GitHub web site suggested that different versions of the code, chapter by chapter were available through git. This did not appear to always work.

One of the examples (creating jenkins) failed in spite of several attempts. I suspect that this was problem with the cartridge than anything else. I did not lose a lot.

This is one problem with a book such as this: it will date quickly and some of the examples not work as what is available through OpenShift changes or is upgraded to newer versions. The cartridge metrics (talked about in the book) no longer exists.

One annoyance, not of the book, was that I needed to sign up to `openshift.redhat.com` - somewhere else that expects you to monitor the web site for changes to T&C rather than email you.

Contributors

Kimball Johnson is a Developer at Shadowcat Systems in Lancaster. He has been programming since a very early age, starting with BBC Micros, then MS DOS and Windows Systems, however was enlightened with a copy of Debian GNU/Linux Woody at university. From this developed an affinity to Systems Administration, but like all good admins he is lazy and so tries to automate as much of his job as possible, and now follows the way of the Infrastructure as Code.

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

Les Pounder works closely with North Western Linux and Free Software groups to promote the use of Open Source software as opposed to proprietary software. He is also the organiser

of UCubed, a free Linux and open source event in Manchester, and an organiser of Barcamp Blackpool and Blackpool Geekup, and has been head of crew at Oggcamp. He writes for Linux Format magazine and contributes to Linux podcasts including Fullcircle, Ubuntu UK Podcast and Linux Outlaws.

Andrew Richards is an IT consultant specialising in Linux and Unix based email servers, general sysadmin and networking. He dabbles in C and Python and has written a number of open source add-ons and patches to qmail. In his free time he enjoys arty foreign films, badminton and cycling. Find him at `www.acrconsulting.co.uk`

Paul Waring currently works in teaching and software support at a large UK university. Outside of work he can usually be found filing documentation bugs against various open source and free software projects. He also edits the FLOSS UK newsletter.

Alain Williams is a folk dancer who particularly likes contras, squares and zesty Playford. In his spare time he runs a Linux consultancy doing freelance system administration, web applications and Open Source technology teaching.

Quentin Wright is a Director of Sunfield Technology working with Linux and Ruby and system integration projects. In his spare time in between playing with reluctant motor vehicles he is pre-occupied with Web and Javascript programming.

Contacts

Kimball Johnson
FLOSS UK Chairman
Preston

Gavin Atkinson
Council member
York

Tim Fletcher
Council member
Manchester

Holger Kraus
Council member
Leicester

Ian Norton
Council member
Morecambe

Stephen Quinney
Council member
Edinburgh

Quentin Wright
FLOSS UK Treasurer
Warwick

Paul Waring
FLOSS UK Newsletter Editor
Manchester

Alain Williams
FLOSS UK System Administrator
Watford

Sam Smith
Events and Website
Cambridge

Jane Morrison
FLOSS UK Secretariat
The Manor House
Buntingford
Herts
SG9 9AB
Tel: 01763 273475
Fax: 01763 273255
office@flossuk.org