MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS

Build a Scalable Monitoring System with Grafana, InfluxDB and Riemann

Managing Multiple Perl 5 Installations with Perlbrew

Distributed Version Control with Raspberrypi3, Fossil and FreeBSD

What Blockchain Can Do Depends on Who You Are

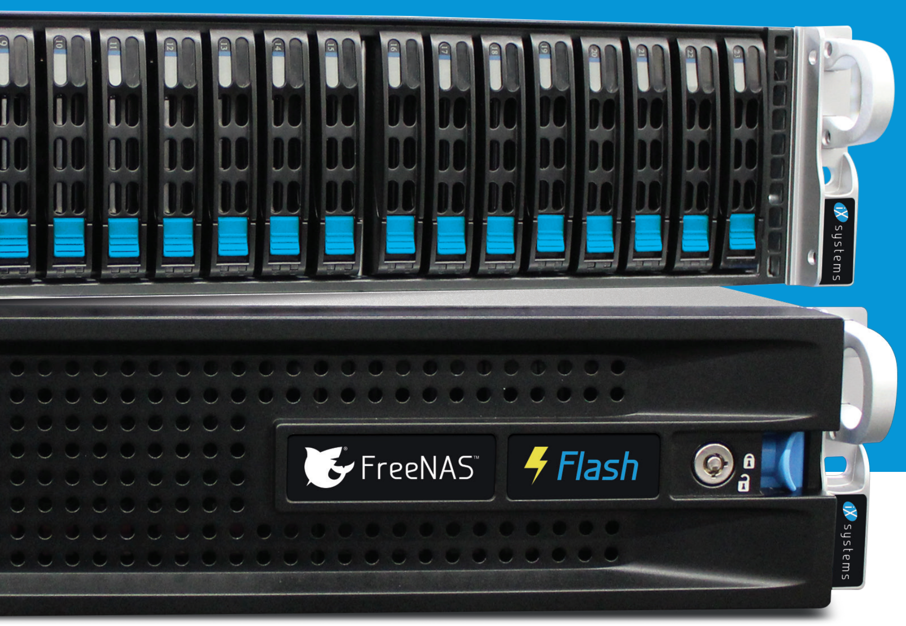
What's Next for Feature Development in FreeNAS/TrueNAS?

Interview with Rob Somerville

VOL 12 NO 01
ISSUE 01/2018 (101)
ISSN 1898-9144

IS AFFORDABLE FLASH STORAGE OUT OF REACH?

NOT ANYMORE!



IXSYSTEMS DELIVERS A FLASH ARRAY FOR UNDER $10,000.

Introducing FreeNAS® Certified Flash: A high performance all-flash array at the cost of spinning disk.

- Unifies NAS, SAN, and object storage to support multiple workloads
- Runs FreeNAS, the world's #1 software-defined storage solution
- Performance-oriented design provides maximum throughput/IOPs and lowest latency
- OpenZFS ensures data integrity
- Perfectly suited for Virtualization, Databases, Analytics, HPC, and M&E
- 10TB of all-flash storage for less than $10,000
- Maximizes ROI via high-density SSD technology and inline data reduction
- Scales to 100TB in a 2U form factor

The all-flash datacenter is now within reach. Deploy a FreeNAS Certified Flash array today from iXsystems and take advantage of all the benefits flash delivers.

Call or click today! **1-855-GREP-4-IX** (US) **|** **1-408-943-4100** (Non-US) **|** **www.iXsystems.com/FreeNAS-certified-servers**

Editor's Word

Dear Readers,

I hope that all of you are well and are excited for the new BSD issue. I am sorry for the delay hope after reading the articles, that they were worth the wait.

In this BSD issue, you will have the chance to learn how to build a scalable monitoring system with Grafana, InfluxDB and Riemann. This article is very a useful tutorial written by Sanel Zukan. In this article, he is going with a solution that will be very lightweight (you can run it on a single server), flexible (you can easily scale it to many servers and monitoring hosts) and fast (able to ingest thousands of events per second). I hope you are excited about this solution and are anxious to start reading it. Our expert, Abdorrahman Homaei will discuss Distributed Version Control. He claims that although Fossil's advanced features, installing Fossil and FreeBSD on RPI3 is minimalistic. Fossil lets you create your own professional Distributed Version Control server that can compete with something like Git. Also, please don't miss the article written by Luca Ferrari. This month, he will share his thoughts on Perlbrew. You'll learn how to install, initialize, and run Perlbrew, how to install different Perl 5 interpreters on the same machine and how to use a specific one depending on your needs. You'll also learn how to manage, rename, clone, and delete Perl 5 interpreters, and how to return back to system wide Perl 5.

Finally, you may find interest in this month's column provided by Rob Somerville. This includes the interview with Rob that we prepared for you. Rob has been in our team for many years now, and we hope you will enjoy this interview in which you can meet him more personally. I hope to present more members from our team in such interviews in the coming months. Starting this month, we are introducing a new column Expert Speak by E.G.Nadhan, Chief Technology Strategist for the Central Region from Red Hat. In this role, Nadhan works with executive leadership to innovatively drive Digital Transformation while providing thought leadership across emerging technology paradigms in the industry.

Please also do not miss our specials like the interview with Kris Moore, which was prepared by our reviewers. Please see full table of contents for more information.

I would like to express my gratitude to our sponsors and to our experts who contributed to this publication and invite others to collaborate with our magazine. The next issue of BSD Magazine will be published in about 3 weeks at the end of February. If you are interested in learning more about the future content, or if you would like to get in touch with our team, please feel free to send your messages to ewa@bsdmag.org, and I will be more than pleased to talk to you and answer all your questions.

Thank you and enjoy reading!
Ewa & The BSD Team

Table of Contents

In Brief

Specials

Perl

FreeBSD

Expert Speak by E.G.Nadhan

Interview

Column

MAGAZINE BSD

Editor in Chief:

Ewa Dudzic
ewa@bsdmag.org
www.bsdmag.org

Contributing:

Sanel Zukan, Luca Ferrari, José B. Alós, Carlos Klop, Eduardo Lavaque, Jean-Baptiste Boric, Rafael Santiago, Andrey Ferriyan, Natalia Portillo, E.G Nadhan, Daniel Cialdella Converti, Vitaly Repin, Henrik Nyh, Renan Dias, Rob Somerville, Hubert Feyrer, Kalin Staykov, Manuel Daza, Abdorrahman Homaei, Amit Chugh, Mohamed Farag, Bob Cromwell, David Rodriguez, Carlos Antonio Neira Bustos, Antonio Francesco Gentile, Randy Remirez, Vishal Lambe, Mikhail Zakharov, Pedro Giffuni, David Carlier, Albert Hui, Marcus Shmitt, Aryeh Friedman

Top Betatesters & Proofreaders:

Daniel Cialdella Converti, Eric De La Cruz Lugo, Daniel LaFlamme, Steven Wierckx, Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omokanwaye, Radjis Mahangoe, Katherine Dizon, Natalie Fahey, and Mark VonFange.

Special Thanks:

Denise Ebery
Katherine Dizon

Senior Consultant/Publisher:

Paweł Marciniak

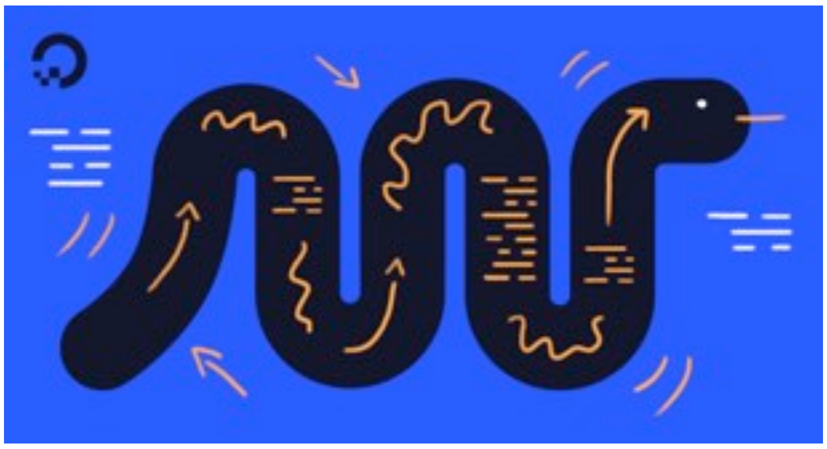
Publisher: Hakin9 Media SK,
02-676 Warsaw, Poland Postepu 17D, Poland
worldwide publishing
editors@bsdmag.org

Hakin9 Media SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail: editors@bsdmag.org

TrueNAS
X20
iX
systems

In Brief

How To Code in Python: A DigitalOcean eBook



The DigitalOcean team is happy to share the "How To Code in Python 3" tutorial series as an eBook that can serve as both a teaching tool for beginners and a point of reference for more seasoned developers. Their goal in making this tutorial series available in an eBook format is to facilitate access to this educational content. This is especially significant for people with limited internet access, long commutes without wifi, or who primarily access written material from mobile devices. You can now download the free eBook in one of the following formats:

How To Code in Python EPUB

How To Code in Python PDF

Source: https://www.digitalocean.com/

ZFS vs. OpenZFS

You've probably heard us say a mix of "ZFS" and "OpenZFS" and an explanation is long-overdue. Michael Dexter clears up what ZFS and OpenZFS refer to and how they differ.

From its inception, "ZFS" has referred to the "Zettabyte File System" developed at Sun Microsystems and published under the CDDL Open Source license in 2005 as part of the OpenSolaris operating system. ZFS was revolutionary for completely decoupling the file system from specialized storage hardware and even a specific computer platform. The portable nature and advanced features of ZFS led FreeBSD, Linux, and even Apple developers to start porting ZFS to their operating systems and by 2008, FreeBSD shipped with ZFS in the 7.0 release. For the first time, ZFS empowered users of any budget with enterprise-class scalability and data integrity and management features like checksumming, compression and snapshotting, and those features remain unrivaled at any price to this day. On any ZFS platform, administrators use the zpool and zfs utilities to configure and manage their storage devices and file systems respectively. Both commands employ a user-friendly syntax such as'zfs create mypool/mydataset' and Micheal welcomes you to watch the appropriately-titled webinar "Why we love ZFS & you should too" or try a completely-graphical ZFS experience with FreeNAS.

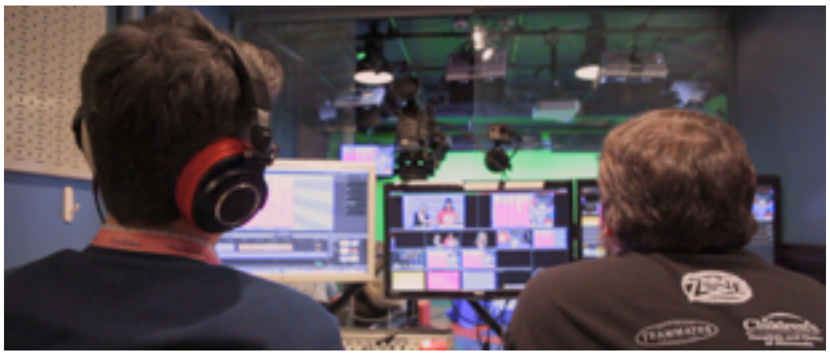
Yes, ZFS is really as good as people say it is. After enjoying nearly a decade of refinement by a growing group of developers around the world, ZFS became the property of database vendor Oracle, which ceased public development of both ZFS and OpenSolaris in 2010. Disappointed but undeterred, a group of OpenSolaris users and developers forked the last public release of OpenSolaris as the Illumos project. To this day, Illumos represents the official upstream home of the Open Source OpenSolaris technologies, including ZFS. The Illumos project enjoys healthy vendor and user participation but the portable nature and compelling features of ZFS soon produced far more ZFS users than Illumos users around the world. While most if not all users of Illumos and its derivatives are ZFS users, the majority of ZFS users are *not* Illumos users, thanks significantly in part to FreeNAS which uses the FreeBSD operating system. This imbalance plus several successful ZFS Day events led ZFS co-founder Matt Ahrens and a group of ZFS developers to announce the OpenZFS project, which would remain a part of the Illumos code base but would be free to coordinate development efforts and events around their favorite file system. ZFS Day has grown into the two-day OpenZFS Developer Summit and is stronger than ever, a testament to the passion and dedication of the OpenZFS community.

Oracle has steadily continued to develop its own proprietary branch of ZFS and Matt Ahrens points out that over 50% of the original OpenSolaris ZFS code has been replaced in OpenZFS with community contributions. This means that there are, sadly, two politically and technologically-incompatible branches of "ZFS" but fortunately, OpenZFS is orders of magnitude more popular thanks to its open nature. The two projects should be referred to as "Oracle ZFS" and "OpenZFS" to distinguish them as development efforts, but the user still types the 'zfs' command, which on FreeBSD relies on the 'zfs.ko' kernel module. My impression is that the terms of the CDDL license under which the OpenZFS branch of ZFS is published protects its users from any patent and trademark risks. Hopefully, this all helps you distinguish the OpenZFS *project* from the *ZFS technology*.

Source:
https://www.ixsystems.com/blog/zfs-vs-openzfs/

TrueNAS Makes the Cut for Avid Editing

TrueNAS is proven to speed up M&E editing. IndieStor's MIMIQ is an application that allows you to use Avid bin-locking with a TrueNAS unified storage appliance. The bin-locking feature prevents an active Avid media project from being overwritten, allowing multiple users on a network to make changes and work on the same project.



The MIMIQ application combined with TrueNAS is a cost-effective alternative to Avid® ISIS®/NEXIS® in your media production environment. With TrueNAS and MIMIQ, you won't be constrained to the limits of Avid proprietary hardware while working on Avid Media Composer projects. TrueNAS is capable of bandwidth speeds exceeding 25Gb/s. This makes it sufficient for multiple 4K 60FPS video streams resulting in a faster workflow in a shared production environment. Capacity for a single TrueNAS unit starts at a handful of gigabytes and grows to nearly five petabytes with expansion shelves.

TrueNAS uses OpenZFS and gives better latency, higher performance, and improved reliability compared to legacy storage vendors. OpenZFS is a self-healing, copy-on-write file system that allows editors to protect their creative products from corruption, bit-rot, data decay, and computer or human error.

Source:
https://www.ixsystems.com/blog/truenas-makes-cut-avid-editing/

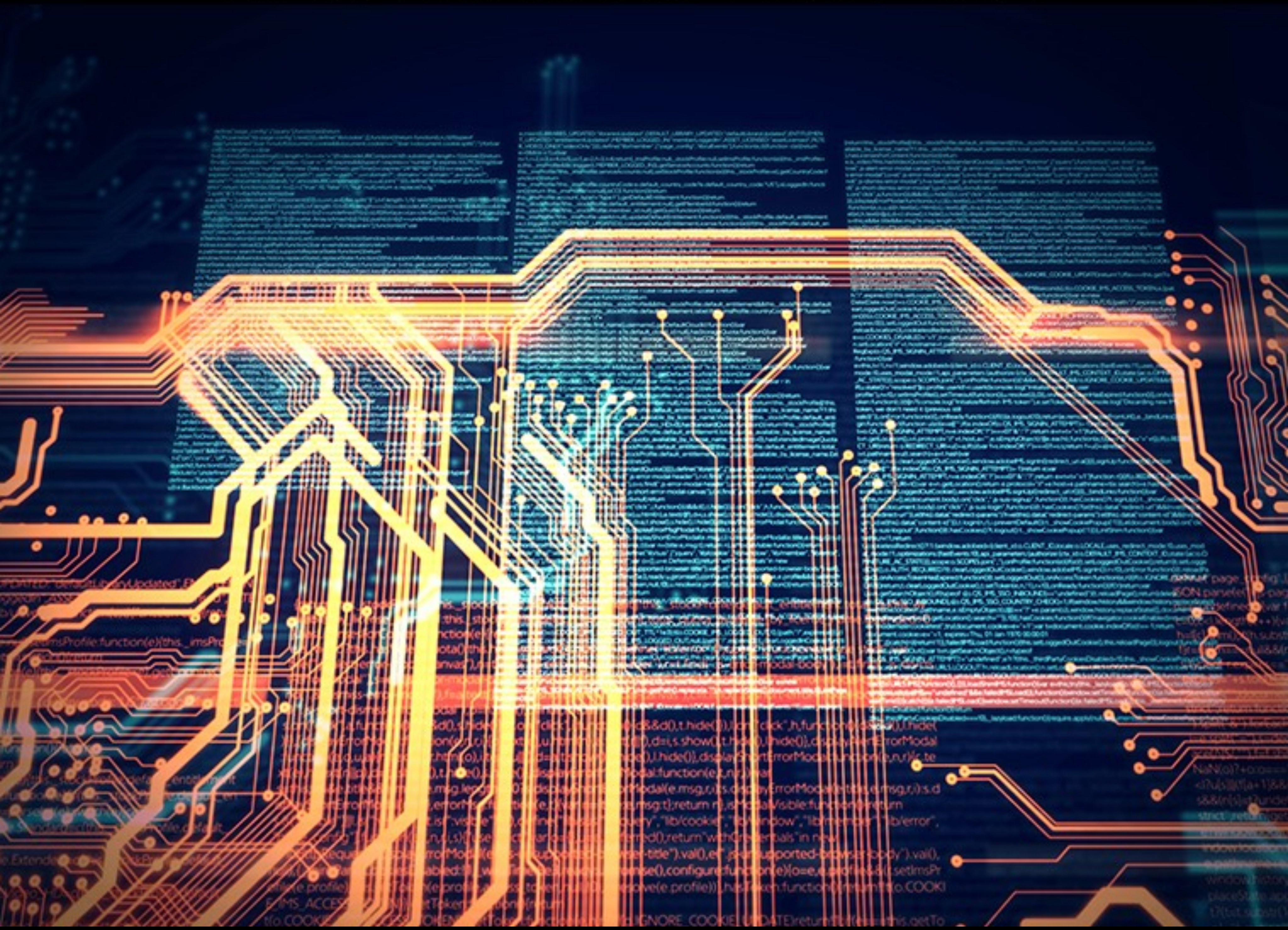
OPNsense® 18.1 Released

For more than 3 years now, OPNsense is driving innovation through modularising and hardening the open source firewall, with simple and reliable firmware upgrades, multi-language support, HardenedBSD security, fast adoption of upstream software updates as well as clear and stable 2-Clause BSD licensing. The team presents to you the sum of another major iteration of the OPNsense firewall. Over the second half of 2017 well over 500 changes have made it into this release, nicknamed "Groovy Gecko". Most notably, the firewall NAT rules have been reworked to be more flexible and usable via plugins, which is going to pave the way for subsequent API works on the core firewall functionality.

These are the most prominent changes since version 17.7:

- FreeBSD 11.1, PHP 7.1 and jQuery 3 migration
- Realtek vendor NIC driver version 1.94
- Portable NAT before IPsec support
- Local group restriction feature in OpenVPN and IPsec
- OpenVPN multi-remote support for clients
- Strict interface binding for SSH and web GUI
- Improved MVC tabs and general page layout
- Shared forwarding now works on IPv6, in conjunction with "try-forwarding" and improved reply-to multi-WAN behaviour
- Easy-to-use update cache support for Linux and Windows in web proxy
- Intrusion detection alert improvements and plugin support for new rulesets (ET Pro, Snort VRT)
- Revamped HAProxy plugin with introduction pages
- Moved interface selection to menu and quick search for firewall rules, DHCP and wireless status
- Alias backend rewrite for future extensibility
- Plugin-capable firewall NAT rules
- Migration of system routes UI and backend to MVC (also available via API)
- Reverse DNS support for insight reporting (also available via API)
- Fully rewritten firewall live log in MVC (also available via API)
- New plugins: zerotier, mdns-repeater, collectd, telegraf, clamav, c-icap, tor, siproxd, web-proxy-sso, web-proxy-useracl, postfix, rspamd, redis, iperf, arp-scan, zabbix-proxy, frr, node_exporter

Source:
https://opnsense.org/opnsense-18-1-released/



What's Next for Feature Development in FreeNAS/TrueNAS?

Interview with Kris Moore

By Eric Geissinger, Luca Ferrari and Imad Soltani

Enterprise Ready
Economical
Fast
Protected
HA
Highly Available
Safeguard Your Data
ZFS
Open Source Development Model
systems

Can you please introduce yourself, your background, and your current involvement with the projects?

My name is Kris Moore, and I am the Vice President of Engineering at iXsystems. This position includes oversight of the FreeNAS and TrueNAS projects, as well as TrueOS and other related endeavors here at iXsystems. For longer than a decade, I've been heavily involved with the BSD community, founded the PC-BSD project (Now TrueOS) and co-hosted the video podcast, BSDNow, for several years alongside Allan Jude.

How difficult is it to contribute to the projects as a developer?

iXsystems has a strong background in open-source, and as such, we make an effort to help get folks involved at every level. All of our projects are currently hosted on GitHub which makes getting access to the source and contributing very easy. We actively participate in pull-requests, perform reviews and will help in any way we can to ensure a developer can get their changes merged into the project. For less technical folks, we even keep our documentation and handbooks on GitHub and run them as an open-source project, which allows for greater contribution possibilities.

How difficult is it, for a developer, to provide a plugin and deploy it to several installations?

Under the old Warden-based plugin system for FreeNAS, it was rather challenging to develop and deploy plugins to FreeNAS. In 2017, we undertook a massive effort to re-write our entire plugin framework and based it upon **iocage**. This new framework makes plugin creation much easier than before, by simply creating a small JSON manifest of the FreeBSD version to be installed, list of packages to install and other configuration options.

What are the main differences between FreeNAS and TrueNAS? What's new in FreeNAS/TrueNAS 11.1?

FreeNAS is the world's most popular open-source software-defined storage. Users can freely download it from our website, install it on their preferred system,and have enterprise-quality NAS functionality at their fingertips. TrueNAS is our commercial version of FreeNAS, specifically tied to work on an appliance sold by iXsystems. Extra care goes into the TrueNAS product to ensure things like hardware compatibility and regression testing. Then we add extra features such as HA, Fiber Channel, vCenter support, proactive-support, and more.

For version 11.1, we've added some important new functionality. On the ZFS side, we've added support for sequential scrub / resilver, which can greatly impact the time required to scrub a pool and rebuild a replaced disk. This helps reduce the mathematical chance that another drive may fail while you are already waiting for a rebuild to complete. We've also made some enhancements to our backup functionality. Currently, it supports a variety of cloud providers such as Amazon S3, BackBlaze B2, Google Cloud and Azure blob storage. This provides users with many more methods to move data to and from the public cloud effortlessly. Additionally, other enhancements such as Netdata, VM improvements, and Time Machine improvements all landed in 11.1, making it an exciting new entry in the FreeNAS line.

What's next for feature development in FreeNAS/TrueNAS?

We're constantly looking to make improvements to our feature set. The biggest change on the horizon is the new UI which will be Angular based. It will provide a much nicer looking interface for setup and management of your FreeNAS/TrueNAS system, while also having modern touches like mobile support. Additionally, we are looking at further improvements to the ZFS filesystem and additional features to land in our VM / Plugins support.

What are the advantages of using OpenZFS as a storage layer within the projects?

The OpenZFS file system sets the gold standard for enterprise features, reliability and support. Paired with the FreeBSD operating system, it allows commodity hardware to be used with the same type of uptime and resilience normally only seen on very expensive enterprise platforms. Features such as datasets, snapshots & clones, volume management and more make ZFS the best all-around choice for a NAS product.

Is it possible to run FreeNAS/TrueNAS without using OpenZFS, in order to assimilate pre-existing file systems or reduce resource needs?

No, FreeNAS and TrueNAS both are tuned to work specifically with ZFS, all the way from services (such as Samba/NFS) to the middleware and UI.

What feature(s) in FreeNAS/TrueNAS do you feel sets it apart from other open-source storage solutions? (OpenMediaVault, Openfiler etc.)

Many users come to appreciate the flexibility and power that FreeNAS/TrueNAS offer. The UI helps make many complex tasks easy to perform, while still offering a bevy of knobs to allow users and administrators to properly “fine-tune” for their specific environment. This is what allows the same code base to be used for everything from a small SoHO NAS (Perhaps serving multimedia with Plex), all the way up to a multi-petabyte rig serving as a high-availability storage backend for VMware.

Do the above NAS solutions integrate (or come already deployed) with backup tools for devices and operating systems?

FreeNAS is very comfortable in the backup target role. Users can easily use OSX TimeMachine to backup to FreeNAS, while Windows users can run a variety of clients to backup either over SMB/NFS or now even S3 locally.

With Spectre and Meltdown dominating the news, when do you expect a patch, (if there isn't one already), and are you expecting that patch to have a large performance degradation for storage and virtual workloads?

We, like the rest of the industry, are very concerned about the recent Meltdown / Spectre disclosures. We are in the process of vetting patches which will probably be released by the time of this publishing. Part of that is our QA team will be helping to benchmark the performance cost of these mitigations, and working with engineers to help reduce the footprint as much as possible. I expect we will see refinements to this mitigation work throughout 2018.

How is it difficult to integrate such a NAS solution in an existing enterprise network? What are the options and protocols to share and cooperate with foreign systems and devices?

Not very difficult at all. FreeNAS and TrueNAS at its core are designed to work in an enterprise environment, and already support most major protocols that you’d find in such an environment. Things such as LAGG/LACP, SMB, SNMP, NFS, iSCSI, Fiber Channel, Active Directory, LDAP and many more make up this long list of enterprise interoperability support.

What is the learning curve for administrators that do not use Unix, FreeBSD, OpenZFS and related technologies?

Part of what has made FreeNAS and TrueNAS so successful is their ability to help guide rather complex setups via the UI. While we have a full-featured API, we also try to expose any and all functionalities as a UI option somewhere. At first glance, this can be a bit overwhelming when presented with the sheer number of options available, but one very important source of help is the fantastic documentation. FreeNAS and TrueNAS both ship every version with a set of comprehensive user-guides, documenting the ins and outs of every option in an easy-to-read manner. When talking to users, it’s not uncommon for us to hear the statement that our documentation is the best in the industry. That plus the wealth of knowledge on our forums, google links, and youtube walkthroughs make FreeNAS and TrueNAS one of the easiest things to troubleshoot around.

What are the smallest and biggest systems deployed so far with a FreeNAS/TrueNAS installation?

We've seen rather humble systems running FreeNAS even using USB drives for storage all the way up to TrueNAS systems with over a PB stored in multiple shelves of drives.

Do the projects work fine for today's home needs, including sharing of photos and video streaming across wireless networking?

Yes, FreeNAS integrates well into this type of home-network. I'd venture to say that's how the vast majority of home-users' FreeNAS boxes are setup today. To do this, you'd just plug your FreeNAS system into your existing network setup and then both wired and wireless clients should be able to browse for and share data across any protocols you have enabled.

For the home user that is still running FreeNAS 9.10, what would the upgrade path be to get to 11.1?

The upgrade path is pretty straightforward. Via the UI, you'll just need to go to the updates page and be sure to change to the FreeNAS-11-STABLE train (on the drop-down menu). With that selected, click update and the system will automatically download and reboot to apply the updates. Once finished, you can launch the same UI as before and start exploring the latest features 11.1 has to offer.

Is it possible to run the projects in a headless setup?

Indeed! Many users setup FreeNAS in a headless environment, especially once the initial setup is complete. One testament to its reliability is how often we hear, "Oh yea, I've been running FreeNAS for years. I set it up years ago and haven't had to touch it since".

Thank you

You're welcome! It's been my pleasure, and I'm looking forward to seeing 2018 be a year that FreeNAS and TrueNAS make new leaps forward in functionality, ease of use and popularity. If you have other specific questions on either product, don't hesitate to contact us over on the Forums or directly at iXsystems.

Managing Multiple Perl 5 Installations with Perlbrew

Perl 5 is a very stable, feature-rich, programming language used in a multitude of environments and applications, including operating system management tasks. Perl 5 is available in several versions, and usually, all versions are backwards compatible. However, there are edge cases and new features that could not be available on all versions. When dealing with Perl 5 application, it is a relief to know that the exact version required can be installed, depending on the specific feature set or compatibility, without having to ask for administrator privileges or the need to touch the system's wide Perl 5 installation. Perlbrew allows exactly that.

What you need to know

- **Basic Perl 5 knowledge and terminology**
- **Basic FreeBSD shell knowledge**

What you will learn

- **How to install perlbrew, initialize, and run it**
- **How to install different Perl 5 interpreters on the same machine, and how to use a specific one depending on your needs.**
- **How to manage, rename, clone, and delete Perl 5 interpreters**
- **How to return back to system wide Perl 5**

When developing Perl 5 application, it is quite common to come across a particular feature of the language that is not available on the system you are going to deploy your application or script.

This usually happens when you deal with the old operating system, or on the other hand, when you deal with out-dated programs that could not run fine on modern Perl 5 instances.

The problem arises from the wide availability of Perl 5: being a feature-rich, easy, quick, stable language, it has been used worldwide to perform several operating system tasks. Thanks to its rich ecosystem, the Comprehensive Perl Archive Network (CPAN), a lot of applications have been developed using Perl 5.

This makes it pretty much impossible to find out a Unix operating system not running a version of Perl 5. However, while keeping interpreters and libraries up-to-date is a good habit, for several well-known reasons, it is not always possible.

Why should a Perl 5 version not be upgraded to the latest version? Well, it could be because some application stack has not been verified and tested on such version, or some features strictly require a specific version that happens to be in the past.

It is possible to write any philosophical essay on that, but the truth is that the Perl 5 version needed to run or deploy the application(s) could be in the past or the future.

Having to upgrade or downgrade the operating system Perl 5 installation is not a feasible solution in many environments.

First of all, it could cause endless discussions about privileges required to change the Perl 5 package itself, and second, it could just break things.

There is more than having to *simply* deploy an application on a particular Perl 5 version: what if the user wants to patch a particular Perl 5 version to test some experimental work (or contribution) without having to nuke all the other users' capabilities to run their Perl programs?

A good solution would be to install a Perl 5 version in a *private space*, particularly in the user's space (e.g., the home directory), so that no privileges are required and no system wide Perl 5 is changed. However, this approach requires some deep knowledge into the Perl 5 executables: knowing variables, modules to load local libs, and so on.

Therefore, while this approach is the preferred one for developers, it is not feasible for non-developer users.

Perlbrew fills the gap making it easy to manage several Perl 5 installations in a user's private space therefore removing the need for privileges without changing the system wide Perl 5 installation. Also, it hides all the machinery to make a specific Perl 5 version to work.

This paper introduces Perlbrew, providing simple instructions how to install and run.

Installing perlbrew

There are three main ways to install perlbrew:

- Using the *ports* collection (the port is `devel/p5-App-perlbrew`);
- By manually downloading and installing the latest version from the project web site;
- By installing it from the CPAN (the name is `App::Perlbrew`).

The above are the ways to obtain a *stable* version of perlbrew, while of course, it is possible to check out the source tree from the Git repository and go with that. But as usual in this approach, it would lead to a not-fully-working installation.

Installing from the Project Web Site

While the installation from the ports collection is, probably, the most FreeBSD-style one, getting the program from the project web site provides a better control on its installation, as well as an always up-to-date version.

Of course, being perlbrew a Perl 5 application by itself, it is required that the system has a Perl 5 interpreter installed system wide to bootstrap perlbrew. It is possible to check the installed Perl 5 version with the `-v` flag:

```
% perl -v

This is perl 5, version 24, subversion 3
(v5.24.3) built for amd64-freebsd-thread-multi

Copyright 1987-2017, Larry Wall
```

In order to install perlbrew directly from the project's website, a command line utility is required to download data via HTTP. For instance `wget(1)`, `fetch(1)` (always available on a FreeBSD plain install) or `curl(1)`. Assuming `wget(1)` is only available, installing perlbrew is as easy as providing a URL to the latter command, and piping the result through a shell interpreter:

```
% wget -O - https://install.perlbrew.pl | sh
...
## Installing perlbrew
Using Perl </usr/local/bin/perl>
perlbrew is installed:
~/perl5/perlbrew/bin/perlbrew
...
```

If everything works fine, perlbrew creates a directory tree in the user's home directory under the *perl5/perlbrew* directory. That's the main directory where perlbrew will install Perl 5 versions, and `perl5` is the local directory to enable per-user modules.

At the end of the installation, the program prints a message about sourcing a file into `.profile`. The idea is that the command perlbrew aliases within the user's shell to support all internal commands. At the moment, under `perl5/perlbrew/etc`, there are configuration files to support bash, fish, and csh.

The author works with `zsh(1)` that, having a variable and aliasing system compatible with the one of `bash(1)`, works well. Depending on the user's shell, the place and the syntax to *load* the functions could be slightly different. But in many cases, it does suffice to do something like:

```
$ echo "source ~/perl5/perlbrew/etc/bashrc" >
~/.profile
```

or, in the case of `zsh(1)`:

```
% echo "source ~/perl5/perlbrew/etc/bashrc" >
~/.zprofile
```

If the sourcing is placed on the *profile* file there is the need to start a new shell session, usually logging out and logging in again.

More complex solutions could simply source the perlbrew functions into *rc* files, so that it simply does suffice to *restart a new shell*. Adding the following piece of code to an *rc* shell file (e.g., .zshrc, .bashrc and so on) would do the trick:

```
if [ -z "$PERLBREW_SOURCED" ]
then

PERLBREW_SOURCED=~/perl5/perlbrew/etc/bashrc
    export PERLBREW_SOURCED
    source $PERLBREW_SOURCED

fi
```

After that, it does suffice to start a new shell by simply issuing the shell executable (e.g., zsh, bash, etc.). . This ensures perlbrew is fully loaded without the need to log-out and log-in again.

Checking the Installation

Once perlbrew has been installed and all the shell files have been loaded, it is possible to check the installation by issuing the perlbrew command, or with the version command to confirm the version installed:

```
% perlbrew version
/home/luca/perl5/perlbrew/bin/perlbrew  -
App::perlbrew/0.82
```

Getting to Know perlbrew

The perlbrew command works through *commands*: each action to be performed has to be specified via one *command* and, or, options. Running perlbrew without any command (or with a wrong command) provides a help screen with all the available commands:

```
% perlbrew

Usage:
    perlbrew command syntax:

        perlbrew <command> [options]
[arguments]

    Commands:

        init           Initialize perlbrew
environment.
        info           Show useful information
about the perlbrew installation

        install        Install perl
        uninstall      Uninstall the given
installation
        available      List perls available to
install
        lib            Manage local::lib
directories.
        alias          Give perl installations
a new name
        upgrade-perl   Upgrade the current
perl
...
```

The help command accepts the name of another command as optional argument, providing more info about the usage of the latter command.

For instance, perlbrew help init prints detailed instructions about the init command.

It is worth noting that after having installed perlbrew, it will not automatically activate itself.

That is **the user will continue to use the system-wide Perl 5**. This can be easily checked either asking the version of the perl interpreter or asking perlbrew for some info about the current installation:

```
% perl -v

This is perl 5, version 24, subversion 3
(v5.24.3) built for amd64-freebsd-thread-multi

Copyright 1987-2017, Larry Wall
...

% perlbrew info
Current perl:
Using system perl.
Shebang: #!/usr/local/bin/perl
...
```

As you can see, the perl interpreter has not changed, and in fact, even perlbrew states that the user is still using the *system perl*.

Perlbrew identifies Perl 5 versions via the release numbers. For instance, 5.27.7 identifies that specific version of Perl 5. If only a version number is provided, perlbrew automatically consider it as a Perl 5 version.

Otherwise, the prefix perl- can be used, and often, it is displayed through the commands output (e.g., perl-5.7.7). The usage of a prefix helps to avoid clashes with cperl instances.

Installing Perl 5 via perlbrew

Initializing Perlbrew

In order to start using perlbrew, the system has to be *initialized*. Using the init command is as simple as simple as shown below:

```
% perlbrew init

perlbrew root (~/perl5/perlbrew) is
initialized.
```

The initialization step must only be performed one time per installation, and does not enable any specific Perl 5 interpreter. To use another Perl 5 interpreter, it is necessary to download and install it.

Installing another Perl 5 Interpreter

To install a new Perl 5 interpreter, follow the following steps:

- choose the right version from the available ones;
- ask perlbrew to install;
- Optionally, switch to such version of Perl 5.
- Choose the version among those available

The command, available, shows all known Perl 5 versions that can be downloaded and installed. If a version has been installed on the current system, an i is placed at the beginning of the line.

```
% perlbrew available

   perl-5.27.7  available from
<http://www.cpan.org/src/5.0/perl-5.27.7.tar.g
z>
   perl-5.26.1  available from
<http://www.cpan.org/src/5.0/perl-5.26.1.tar.g
z>
   perl-5.24.3  available from
<http://www.cpan.org/src/5.0/perl-5.24.3.tar.g
z>
   perl-5.22.4  available from
<http://www.cpan.org/src/5.0/perl-5.22.4.tar.g
z>
   perl-5.20.3  available from
<http://www.cpan.org/src/5.0/perl-5.20.3.tar.g
z>
   perl-5.18.4  available from
<http://www.cpan.org/src/5.0/perl-5.18.4.tar.g
z>
   perl-5.16.3  available from
<http://www.cpan.org/src/5.0/perl-5.16.3.tar.g
z>
   perl-5.14.4  available from
<http://www.cpan.org/src/5.0/perl-5.14.4.tar.g
z>
   perl-5.12.5  available from
<http://www.cpan.org/src/5.0/perl-5.12.5.tar.g
z>
   perl-5.10.1  available from
<http://www.cpan.org/src/5.0/perl-5.10.1.tar.g
z>
    perl-5.8.9  available from
<http://www.cpan.org/src/5.0/perl-5.8.9.tar.gz
>
    perl-5.6.2  available from
<http://www.cpan.org/src/5.0/perl-5.6.2.tar.gz
>
...
```

Assume the latest version, Perl 5.27.7, is chosen for the installation. The name, perlbrew , recognizes a Perl 5 interpreter via the first column of the available output, so in this case perl-5.27.7. As already stated, it is possible to omit the initial perl-, hence using the name 5.27.7, unless it clashes with cperl versions.

Install the chosen versions

The install command performs a download of the specified version, compiles it, and installs into the perlbrew directory tree.

```
% perlbrew install perl-5.27.7
Fetching perl 5.27.7 as
/home/luca/perl5/perlbrew/dists/perl-5.27.7.ta
r.gz
Download
http://www.cpan.org/src/5.0/perl-5.27.7.tar.gz
to
/home/luca/perl5/perlbrew/dists/perl-5.27.7.ta
r.gz
Installing
/home/luca/perl5/perlbrew/build/perl-5.27.7/pe
rl-5.27.7 into
```

```
~/perl5/perlbrew/perls/perl-5.27.7

This could take a while. You can run the
following command on another shell to track
the status:

  tail -f
~/perl5/perlbrew/build.perl-5.27.7.log

perl-5.27.7 is successfully installed.
```

The installation could take some time, depending on the resources available on the machine.

It is possible to see all installed versions using the list command:

```
% perlbrew list
   perl-5.27.7        (installed on Fri Jan  5
12:05:42 2018)
```

Switch to the new installed Perl 5

Installing a Perl 5 interpreter does not mean it will be used immediately after. To start using a specific interpreter, the switch command must be run, specifying of course the version the user wants to run:

```
% perlbrew switch perl-5.27.7
% perl -v

This is perl 5, version 27, subversion 7
(v5.27.7) built for amd64-freebsd
(with 1 registered patch, see perl -V for more
detail)

Copyright 1987-2017, Larry Wall
...
```

As you can see, immediately after switching to the specified Perl 5 interpreter, the perl command returns the version that is different from the system-wide installed Perl 5 (which in this article was 5.24.3).

Returning to the System Perl 5

If, for any reason, you need to run some commands using the system-wide available Perl 5 interpreter, you can turn perlbrew *off* with the off command. As a result, all perlbrew facilities (i.e., locally installed Perl 5) will be disabled:

```
% perl -v

This is perl 5, version 27, subversion 7
(v5.27.7)
,,,

% perlbrew off
perlbrew is turned off.
% perl -v

This is perl 5, version 24, subversion 3
(v5.24.3)
...
```

Also, it's worth noting that, turning the perlbrew facilities off *downgrades* the Perl 5 interpreter, from 5.27.7 to 5.24.3, with the latter now being the system-wide interpreter.

Also, it is important to note that the off command works only in the current shell, while the switch-off command turns off perlbrew on all subsequent shell instances.

To activate again the Perl 5 installed via perlbrew, it's suffice to re-run the switch command:

```
% perlbrew switch 5.27.7
```

Managing Perl 5 installations

There are several commands are essential when managing the Perl 5 installation. One useful command is alias: it allows for a renaming of a Perl 5 installation to a more human-readable name.

The alias command requires a fully qualified name, not a simple version number. For instance, to name the just installed Perl 5.27.7 to *bsdmagv1* , it is possible to create the new alias:

```
% perlbrew alias create perl-5.27.7 bsdmagv1
% perlbrew list
   bsdmagv1              (5.27.7)
(installed on Fri Jan  5 12:05:42 2018)
```

```
 * perl-5.27.7
(installed on Fri Jan  5 12:05:42 2018)
% perlbrew switch bsdmagv1
```

As shown above, bsdmagv1 is now a usable name that refers to version 5.27.7.

The alias command is useful for post-install management. However, the install command accepts an extra option --as that allows for specifying a name that will be used as the main name for the installed Perl 5:

```
% perlbrew install perl-5.8.9 --as
my-first-perl
...
my-first-perl is successfully installed.
```

Therefore, the Perl 5 will be available to perlbrew via the special name, my-first-perl:

```
% perlbrew list
 * perl-5.27.7               (installed on
Fri Jan  5 12:05:42 2018)
   my-first-perl   (5.8.9)    (installed on
Fri Jan  5 12:49:37 2018)
```

In case a Perl 5 distribution is no longer necessary, the uninstall command can be used to delete it.

The clean command removes the compilation objects and downloaded archives, and that is useful to get back disk space after installations:

```
% du -hs ~/perl5
480M    /home/luca/perl5

% perlbrew clean
Removing
/home/luca/perl5/perlbrew/build/perl-5.27.7
Removing
/home/luca/perl5/perlbrew/build/perl-5.8.9
Removing
/home/luca/perl5/perlbrew/dists/perl-5.27.7.ta
r.gz
Removing
/home/luca/perl5/perlbrew/dists/perl-5.8.9.tar
.bz2
Done

% du -hs ~/perl5
133M    /home/luca/perl5
```

Installing Modules

Perl 5 is famous for its rich module ecosystem, kept in the CPAN. There are several tools to manage CPAN modules from the command line, and perlbrew uses `cpanm`, a zero-configuration client.

To use `cpanm`, you have to install it so that perlbrew itself can use it: the install-cpanm command does what it says:

```
% perlbrew install-cpanm

cpanm is installed to

    /home/luca/perl5/perlbrew/bin/cpanm
```

`cpanm` is installed widely through perlbrew, so that it can be used for all the Perl 5 instances with a single installation.

Once cpanm is working, it is quite easy to install a new module, for instance `Archive::Zip`:

```
% cpanm Archive::Zip
...
Successfully installed Archive-Zip-1.60
4 distributions installed
```

It is also easy to *clone* all installed modules from an instance to another via the command clone-modules as you can see here:

```
% perlbrew clone-modules 5.27.7 5.26.1
...
```

The above command will clone all modules installed on instance 5.27.7 to instance 5.26.1. This command is useful when you have a quite customized Perl 5 installation, and want to back up all the modules you have installed overtime.

Conclusions

perlbrew is a powerful tool to manage several Perl 5 installation under a *private* user space. It allows for running either modern or ancient versions of Perl 4 without having the user to be granted administrative privileges, or changing the system wide Perl 5 installation.

References

Perlbrew official web site https://perlbrew.pl/

Perlbrew Git repository https://github.com/gugod/App-perlbrew

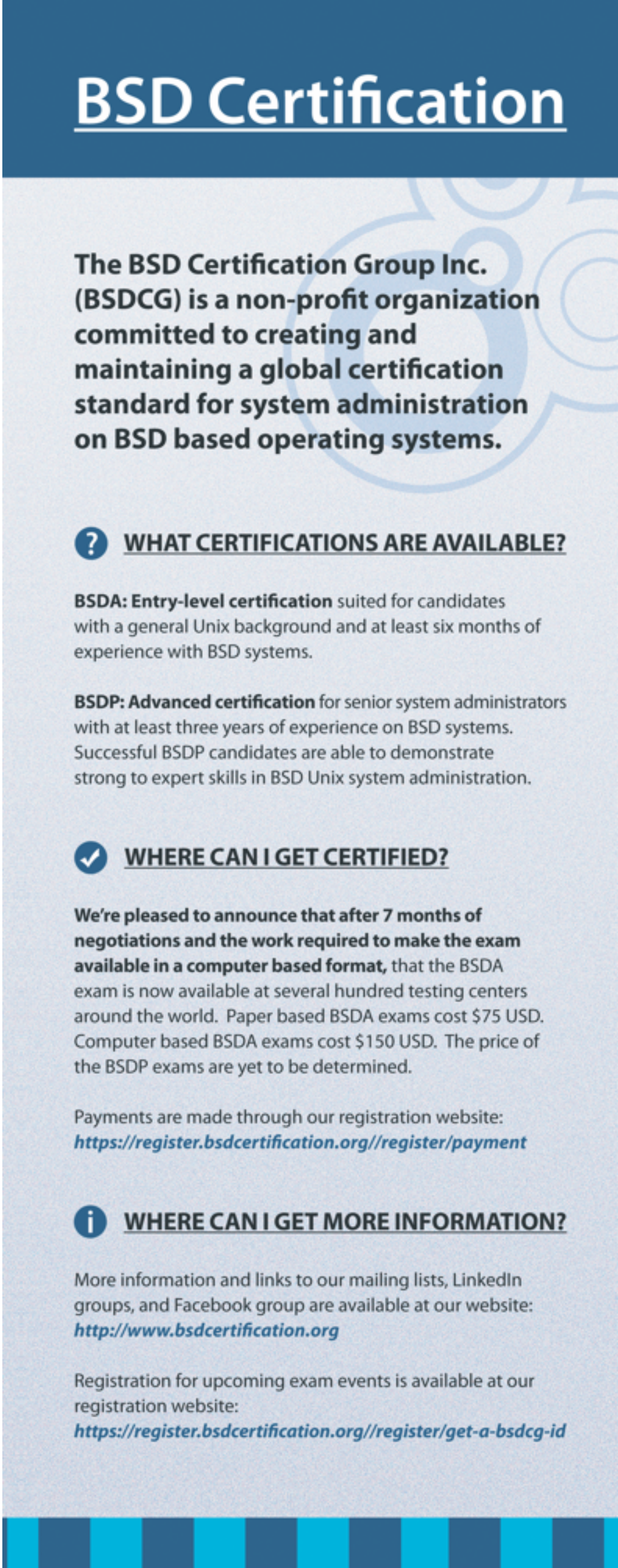
Perl 5 official website http://www.perl.org/

CPAN https://www.cpan.org/

Meet the Author

Luca Ferrari lives in Italy with his beautiful wife, his great son and two female cats. Computer science passionate since the Commodore 64 age, he holds a Master Degree and a PhD in Computer Science. He is a PostgreSQL enthusiast, a Perl lover, an Operating System passionate, a Unix fan and performs as much tasks as possible within Emacs. He considers the Open-Source as the only sane way of doing software and services.

His website is available at http://fluca1978.github.io



Build a Scalable Monitoring System with Grafana, InfluxDB and Riemann

If you run your own web server, VPS instance or cluster, a monitoring system is the essential tooling you simply can't live without. Especially in recent years when microservices became a popular solution to decouple complex architectures (splitting application or system into microservices makes them very easy to scale, deploy and with proper tooling, easy to monitor).

Monitoring systems can not only show us server resource usage metrics such as processor utilization, memory, and disk usage space but also alert us when something unexpected happens like a server reboot or when a service is subject to a DDoS attack.

Modern monitoring systems are also utilized from business and security perspectives: they can show us if our architecture needs expansion, like buying more hardware before our customers see any service degradation, or detecting unusual access to the server or data transfer from the server.

The monitoring solution is a vital part of every company and organization with a non-trivial network or application infrastructure, so we must choose it wisely. Things are even more complicated knowing there are tons of monitoring solutions out there in the form of free, proprietary and cloud offerings.

Necessary software and workflow

In this article, I'm going with a solution that will be very lightweight (you can run it on a single server), flexible (you can easily scale it to many servers and monitoring hosts) and fast (able to ingest thousands of events per second).

Ingredients are:

- FreeBSD of course (11.1 is the latest stable version).
- Telegraf (https://www.influxdata.com/time-series-platform/telegraf/) agent for collecting events.
- Riemann (http://riemann.io/) for events ingestion and manipulation.
- InfluxDB (https://www.influxdata.com/time-series-platform/influxdb/) for storing metrics.
- Grafana (https://grafana.com/) for charting graphs.

Here is a flow diagram to visualize how our monitoring solution will work (see Figure 1):

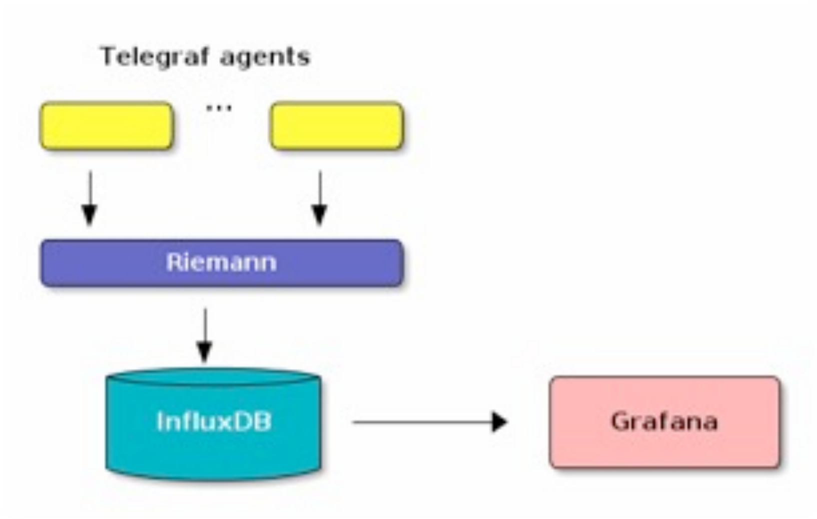


Figure 1. A flow diagram

Telegraf agents should be installed on servers you are planning to monitor, including monitoring server itself. With one agent per server, Telegraf is able to monitor many system resources from a single binary.

Preparing the system

Note: I'm using root account for executing all commands here.

Besides a bare bones FreeBSD install with network connection, you will also need **Java** for **Riemann** and some proxy server, like nginx, to route web traffic to **Grafana**.

Riemann, InfluxDB, Telegraf and Grafana are already packaged for FreeBSD 11.1 so we can install them all with:

```
$ pkg install riemann influxdb telegraf
grafana4
```

FreeBSD comes with four Grafana package variants: *grafana*, *grafana2*, *grafana3* and *grafana4*. We want the latest version, which is *grafana4*.

The pkg install command above will also pull Java 8 as dependency, so you don't have to install it manually. Everything will take around 320MB of space, so make sure you have enough disk space as well.

If you are running older FreeBSD version (or any BSD variant) where these packages are not present, make sure to install the latest available Java version for your system. Riemann, InfluxDB, Telegraf and Grafana all provide ready-made binaries on their respective project sites, so you don't have to compile them manually.

Services setup

Let's configure each of installed programs and then run them.

Configuring and starting Grafana

The default Grafana configuration example is located in /usr/local/etc/grafana.conf, so let's adjust it (Here, I'm showing only snippets that I've changed):

```
...
; setup Grafana in production mode
app_mode = production
...
; setup domain if necessary
domain = localhost
```

Also, make sure to add grafana_enable in /etc/rc.conf:

```
# enables Grafana
grafana_enable="YES"
...
```

And start it with:

```
$ service grafana start
```

Configuring and starting InfluxDB

InfluxDB configuration is in /usr/local/etc/influxd.conf; defaults are reasonably good and I'm going to leave it as is. Let's enable it and start.

Again, add to /etc/rc.conf:

```
...
# enables InfluxDB
influxd_enable="YES"
```

and start it with:

```
$ service influxd start
```

Now we need to create a database, where Riemann will write the data. This will be done with influx shell:

```
$ influx

> CREATE DATABASE riemann
```

To be assured you have it created correctly, list all available databases inside InfluxDB shell:

```
> SHOW DATABASES

name: databases

---------------

name

_internal

riemann    <--- riemann database
```

Optionally, you can protect access to this database with username and password, but before that, make sure to consult InfluxDB *Authentication and Authorization* [1] document.

Configuring and starting Riemann

Riemann configuration is written in Clojure (http://clojure.org), a LISP-like language which takes some time to get used to, assuming you haven't used any LISP-like language before.

Luckily, Riemann comes with sane defaults, so I'm going to add only InfluxDB endpoint for receiving events.

Configuration is in */usr/local/etc/riemann/riemann.config*, but we are going to replace it with the following content:

```
;; InfluxDB database details where Riemann will store the data.
;; Setup :username and :password if you added security step
;; behind riemann database access.
(def influxdb-creds {
  :host "localhost"
  :port 8086
  :db "riemann"
  ; :username "riemann"
  ; :password "riemann_password"
})


(def influxBatchSender
  (batch 100 1/10
      (async-queue! :agg {:queue-size 1000

:core-pool-size 1

:max-pool-size 4

:keep-alive-time 60000}
                              (influxdb
influxdb-creds))))


;; Riemann log file location
(logging/init {:file "/var/log/riemann/riemann.log"})

;; listen on the local interface over TCP (5555), UDP (5555)
;; and websockets (5556)
(let [host "0.0.0.0"]
  (tcp-server {:host host})
  (udp-server {:host host})
  (ws-server  {:host host}))


;; Expire states from its core's index every 60 seconds. Default is 10.
(periodically-expire 60)


(let [index (index)]
  (streams
   (default :ttl 60
       index
       influxBatchSender)))
```

In this configuration, we are stating that all Riemann logs go to the /var/log/riemann/riemann.log file, and that Riemann will listen on TCP (port 5555), UDP (port 5555) and Websockets (port 5556). This gives us ability to receive metrics via TCP or UDP, but also to pool Riemann live metrics via riemann-dash[2] (if you decide to use it).

Now, like with Grafana and InfluxDB, let's setup rc.conf and start Riemann.

In /etc/rc.conf add:

```
riemann_enable="YES"
```

and start it with:

```
$ service start riemann
```

If you got this error on startup:

```
/usr/local/etc/rc.d/riemann: WARNING: /usr/local/etc/riemann/riemann/riemann.config

is not readable.

/usr/local/etc/rc.d/riemann: WARNING: failed precmd routine for riemann
```

it means you are hit with Riemann's startup script bug. Fixing it is easy: just edit /usr/local/etc/rc.d/riemann and modify riemann_config line with:

```
: ${riemann_config:="/usr/local/etc/riemann/riemann.config"}
```

Configuring and starting Telegraf agent

By now, we have the system ready to ingest events. The last part is to setup Telegraf agent, and run it on the local server (or on a different location you'd like to monitor).

Telegraf configuration is stored in /usr/local/etc/telegraf.conf, and the most important part is to enable the Riemann protocol and set Riemann as a destination. Here is configuration snippet that does this:

```
# # Configuration for the Riemann server to send metrics to

[[outputs.riemann]]

  ## The full TCP or UDP URL of the Riemann server

  url = "tcp://localhost:5555"

  ## Riemann event TTL, floating-point time in seconds.

  ## Defines how long that an event is considered valid for in Riemann

  # ttl = 30.0



  ## Separator to use between measurement and field name in Riemann service name

  ## This does not have any effect if 'measurement_as_attribute' is set to 'true'

  separator = "/"



  ## Set measurement name as Riemann attribute 'measurement', instead of prepending it to the Riemann service name

  # measurement_as_attribute = false



  ## Send string metrics as Riemann event states.

  ## Unless enabled all string metrics will be ignored

  # string_as_state = false



  ## A list of tag keys whose values get sent as Riemann tags.

  ## If empty, all Telegraf tag values will be sent as tags

  # tag_keys = ["telegraf","custom_tag"]



  ## Additional Riemann tags to send.

  tags = ["telegraf-output"]



  ## Description for Riemann event

  description_text = "metrics collected from telegraf"



  ## Riemann client write timeout, defaults to "5s" if not set.

  # timeout = "5s"
```

With this, we are telling Telegraf to send events via TCP to localhost on port 5555 (where our Riemann instance is listening) and to add tag telegraf-output with some description.

Tags are very useful for Riemann - you can easily apply transformations on metrics by looking for a specific tag or list of tags. For instance, you can discard all events not tagged with telegraf-output.

First, let's enable it. Add to /etc/rc.conf:

```
telegraf_enable="YES"
```

and start it with:

```
$ service telegraf start
```

Alerting

We have the following options for receiving alerts and notifications, with their pros and cons.

Grafana only alerting

Grafana offers relatively simple (comparing to Riemann) alerting capability. You can visually edit it directly on chart by moving indicator up or down to declare alert thresholds.

For advanced options, Grafana offers conditional blocks with limited number of aggregating functions.

The limitation of Grafana alerts is in its simplicity - it only supports email notifications, and alert rules can be applied just on a graph panel for now.

Riemann only alerting

Riemann offers various alerting capabilities: you can get notifications via email, receive SMS via PagerDuty [3] or get messages via Slack [4]. You can even forward messages to other services like Logstash [5].

However, the most powerful features of Riemann alerting are event analyzing and aggregating functions. For example, Riemann can send one email per hour for the same or similar notifications, keeping your inbox safe from flooding.

Or it can send a notification when there is a certain percentage of service failures across your cluster.

Bad thing about Riemann alerting is complexity - you have to know how Riemann works and have to understand a little bit of Clojure code.

Combining Riemann and Grafana alerting

This is the most powerful option, where you can get the best from both worlds. For example, you can use Grafana alerts to get notified when there is high CPU or memory usage, and use Riemann for more sophisticated notifications, like getting notified when network traffic is unusually high on more than 2 servers at the same time, outside working hours.

Cons of this approach is necessary expertise of both applications and maintenance - alerting conditions are spread between Grafana and Riemann and you can easily end up getting similar notifications from both services.

Wrapping up

I'll leave setting up nginx proxy server for Grafana to you, because that should be fairly an easy task (or you can take a look at this [6] configuration).

To complete Grafana setup, go to running instance with your browser (default url is http://localhost:3000) and from the main menu, choose *Data Source*, like in Figure 2.

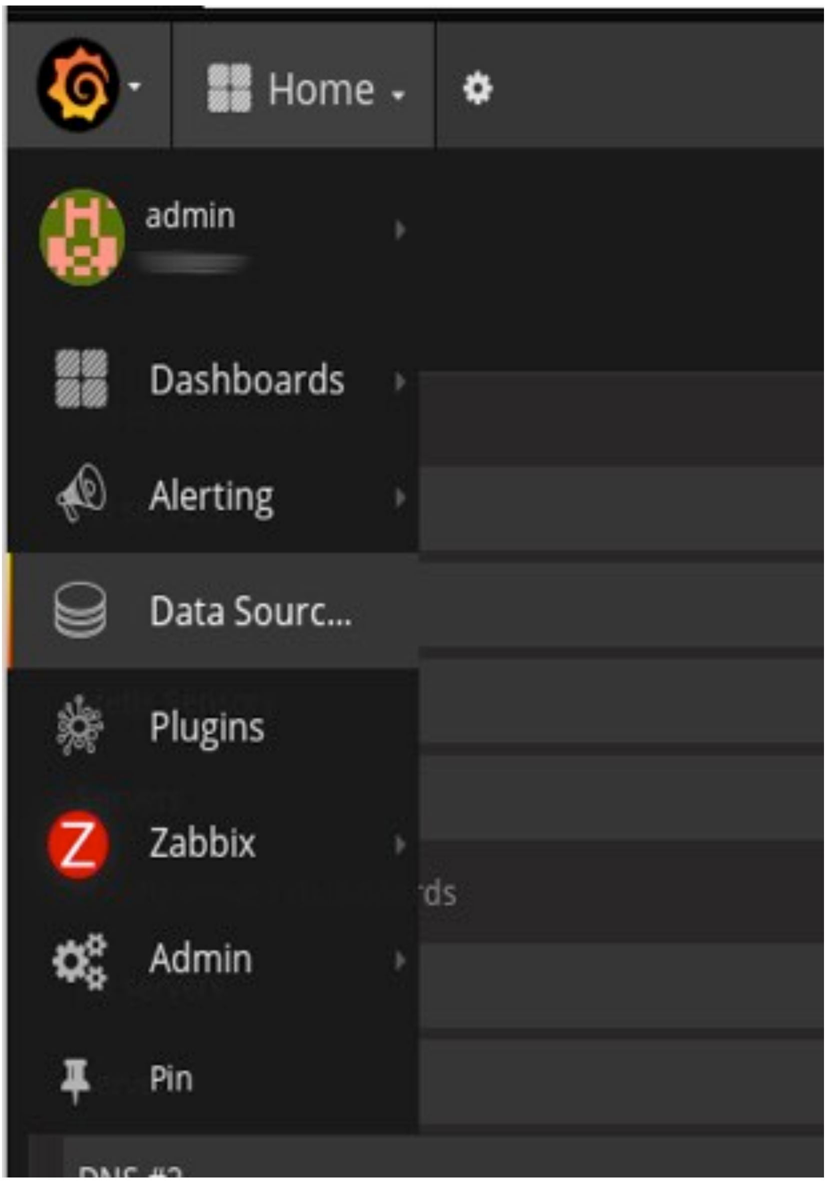
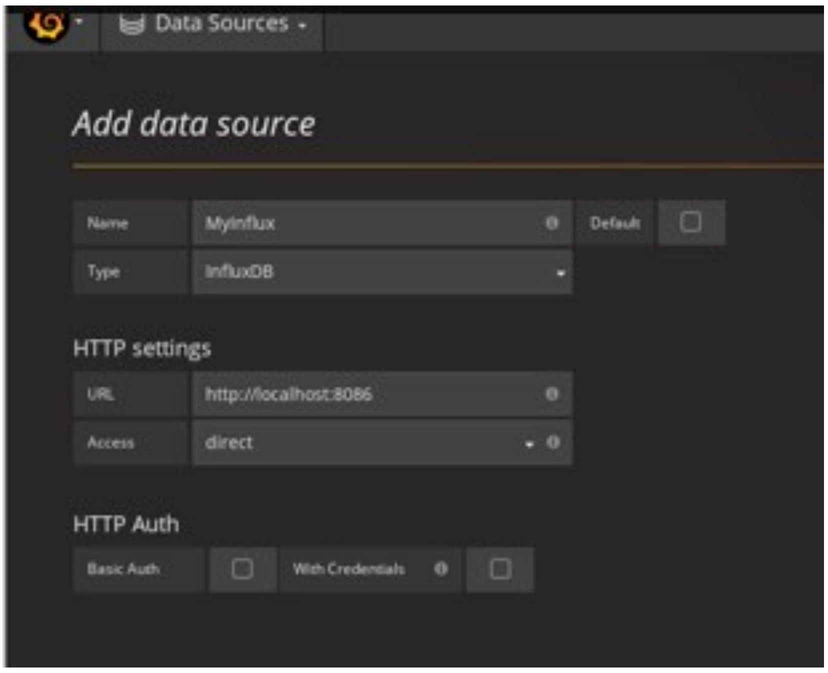


Figure 2. Data source

Click *Add data source* and choose *InfluxDB* from the menu. The Figure 3 shows it with the necessary data.



Add data source

Name: Myinflux
Default: □
Type: InfluxDB

HTTP settings
URL: http://localhost:8086
Access: direct

HTTP Auth
Basic Auth: □
With Credentials: □

Figure 3. Add data source to choose InfluxDB

After you've saved it, go to Grafana's *Getting Started* [7] guide to get acquainted with it. Also, visit *Grafana Dashboards* [8] to find one from hundreds of pre-made dashboards you can download and use.

Happy monitoring!

Meet the Author

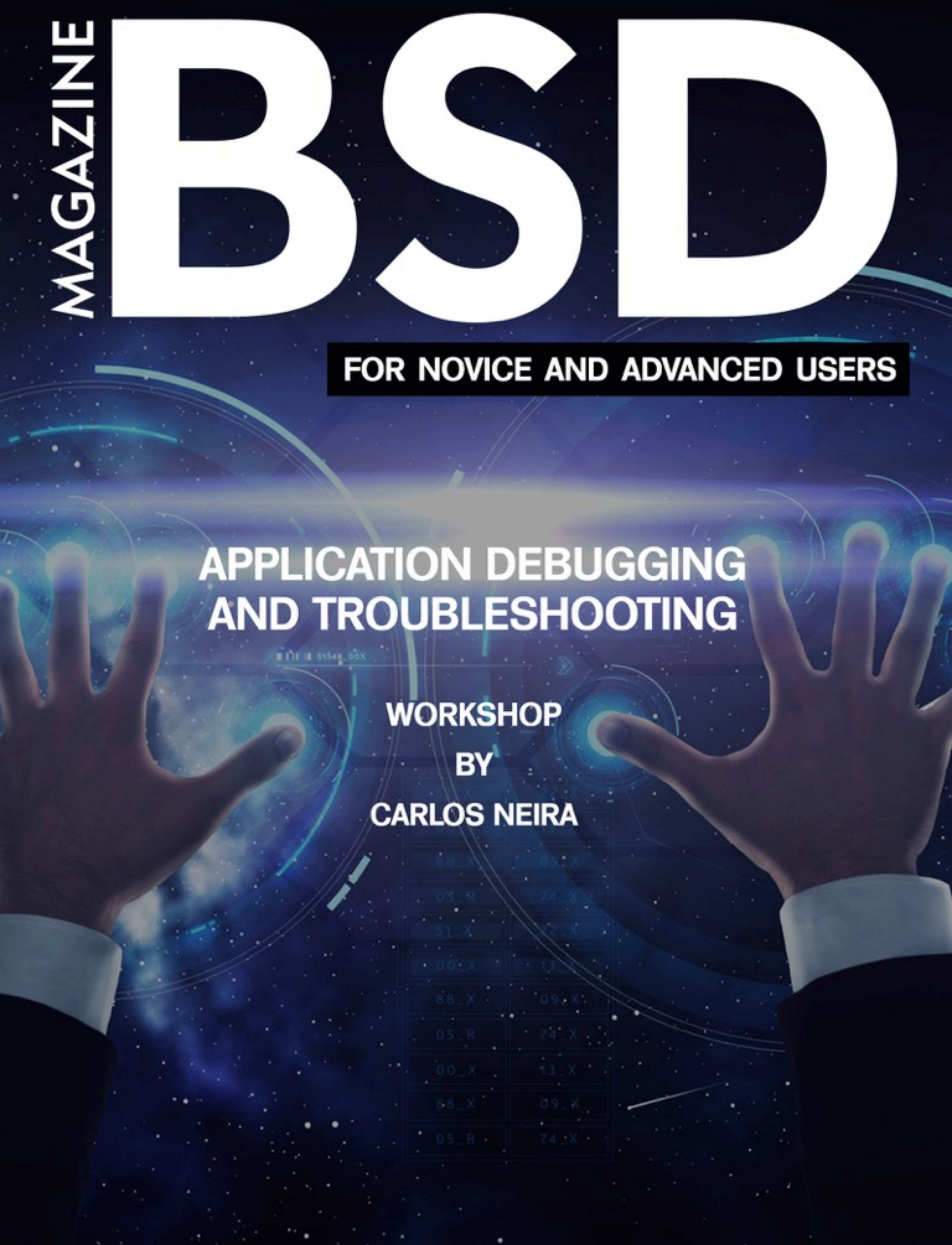
Sanel Zukan is founder and CEO of Hedron d.o.o. (https://hedron.cc), small consulting and data mining company. Sanel has been dealing with Linux and open source software since 2002, was one of the founders of Linux Users Group of Bosnia and Herzegovina, and is long time contributor to numerous open source projects.

In recent years, after rediscovering Common Lisp, Scheme and Clojure, his focus shifted to LISP languages, functional programming, scalable mining and big data systems.

You can reach him via sanelz@gmail.com.

MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS



APPLICATION DEBUGGING AND TROUBLESHOOTING

WORKSHOP

BY

CARLOS NEIRA

Distributed Version Control with RaspberryPi 3, Fossil and FreeBSD

What is Distributed Version Control
What is Fossil
Fossil Features
RaspberryPi 3
How to Install FreeBSD 12 On RaspberryPi 3
How to Access to RaspberryPi Console
How to Install Fossil
Starting a New Project with Fossil
How to Manage Fossil By Using The CLI

What is Distributed Version Control?

Distributed version control is a type of version control where the full source and its history are mirrored on every developer's machine. This allows branching and merging to be managed automatically, increases speeds of most operations (except for pushing and pulling), improves the ability to work offline, and does not rely on a single location for storage of the entire repository.

What Is Fossil?

Fossil is a BSD-licensed, simple, highly-reliable, distributed software configuration management system capable of performing distributed version control, bug tracking, wiki services, and blogging. The software has a built-in web interface and issue tracker.

Content is stored using an SQLite database so that transactions are atomic even if interrupted by a power loss or system crash.

Fossil Features

Integrated Bug Tracking, Wiki, and Technotes – Fossil has issue tracker, wiki and technote.

Built-in Web Interface - Fossil has a built-in web server and can listen on a specific port.

Self-Contained - Fossil is a single self-contained, stand-alone executable.

Simple Networking – simply runs on HTTP or HTTPS.

CGI/SCGI Enabled - No server is required, but if you want to set one up, Fossil supports four easy server configurations.

Autosync - Fossil supports "autosync" mode which helps to keep projects moving forward by reducing the amount of needless forking and merging often associated with distributed projects.

Robust & Reliable - Fossil stores content using an enduring file format in an SQLite database so that transactions are atomic even if interrupted by a power loss or system crash. Automatic self-checks verify that all aspects of the repository are consistent before each commit.

Free and Open-Source - Uses the 2-clause BSD license.

Why FreeBSD?

FreeBSD is a modern and stable operating system. It has many bug fixes and new features for the ARM SOC platform used on the Raspberry Pi. These include:

- CPU frequency and voltage control
- NAND device support
- SMP support
- Stable SD cards detection
- ARM AArch64 architecture support
- Initial ACPI support
- 1-Wire devices support
- GPIO support

What is RaspberryPi 3 and Why RaspberryPi 3?

The Raspberry Pi 3 is the third generation Raspberry Pi **SOC**. It replaced the Raspberry Pi 2 Model B in February 2016.

A **single-board computer** (**SBC**) is a complete computer built on a single circuit board, with microprocessor(s), memory, input/output (I/O) and other features required of a functional computer. Single-board computers were made as demonstration or development systems, for educational systems, or for use as embedded computer controllers. Many types of home computers or portable computers integrate all their functions onto a single printed circuit board.

Compared to the Raspberry Pi 2, it has:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)
- Like the Pi 2, it also has:
- 1GB RAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port

We choose RaspberryPi 3 because of better CPU clock and WIFI support.

How To Install FreeBSD 12 On RaspberryPi 3?

All you need is a 2GB MicroSD and a MicroSD reader. Let's suppose that after you connect your MicroSD to your PC, and FreeBSD names it **da0** (you can find it by **dmesg** command), then:

```
#dd
if=FreeBSD-12.0-CURRENT-arm64-aarch64-RPI3-201
80110-r327788.img of=/dev/da0 bs=1m conv=sync
```

You can download FreeBSD 12 from the official site, and the name of your ".img" can be different.

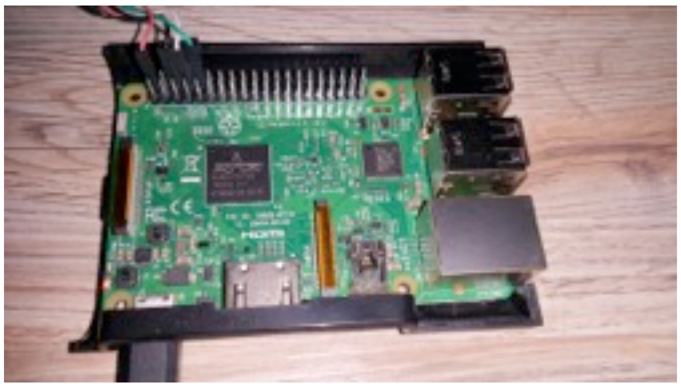
How To Access RaspberryPi Console?

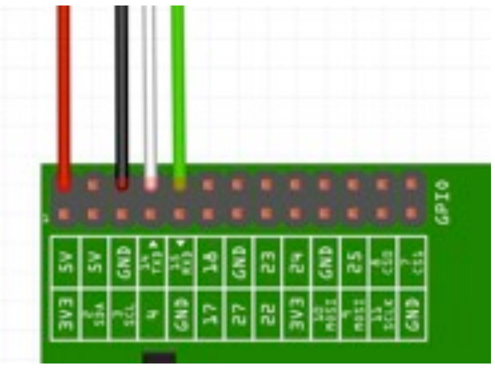
There are three ways to access RaspberryPi :

- HDMI cable and keyboard
- SSH
- Console cable

Access to RaspberryPi with Console Cable is more technical and Geek-Style. So, I will cover it.

Connect RaspberryPi 3 power supply and USB cable. Plug the TTL Serial cable as shown in the pictures below:





Access to RaspberryPi with cu command:

The call unix (cu) utility establishes a full-duplex connection to another machine, giving the appearance of being logged in directly on the remote CPU. It goes without saying that you must have a login on the machine (or equivalent) to which you wish to connect.

FreeBSD will name the USB serial cables like “ttyUX” where X can be a number like 0 or 1

```
#cu -l /dev/ttyU0 -s 115200
```

-l Specify the line to use

-s Set the speed of the connection. The default is 9600

Hit enter, and you will see the login prompt

```
user: root
```

Password is not required.

As you can see, the command prompt shows us “rpi3”. You can issue **“uname -a”** to see more details about arch and FreeBSD version.

How to Install Fossil?

Fossil package is about 1 MB, and you can install it easily using the following command:

```
# pkg install fossil
```

There are no dependencies, and because of that, it’s suitable for RPI3 or any other low-spec computer.

Starting a New Project with Fossil

Let’s create a new repository called bsdmag:

```
# fossil init bsdmag
```

The command above will return something like this:

```
project-id:
af7d78e8d0d1347cdb7d45e1f4573b9c8185
c17a
server-id:
16cee8655038eef551c101354e227d65c0b8
8d00
admin-user: root (initial password
is "c341da")
```

When you create a new repository, you usually want to do some local configuration. This is easily accomplished using the web-server that is built on fossil.

Fossil can act as a stand-alone web-server using one of these commands:

```
#fossil server repository-filename
#fossil ui repository-filename
```

```
# fossil ui bsdmag
```

Unnamed Fossil Project /

The ui command is intended for accessing the web interface from a local desktop. The ui command binds to the loopback IP address only (and thus makes the web interface visible only on the local machine). Also, it automatically starts your web browser pointing at the server. For cross-machine collaboration, use the server command, which binds on all IP addresses and does not try to start a web browser.

```
# fossil server bsdmag --port 80
```

And you can configure your new repository using the Fossil UI in your browser.

Manage Fossil By Using The CLI

There are some tasks you cannot do through a web interface. Like how to reset root password or to convert git repository to fossil-scm compatible repository. All you need is root access.

You can list all fossil commands by:

```
# fossil -help -a
```

Listing Repository Users

To list all bsdmag repository users, issue this command:

```
# fossil user list -R bsdmag
```

Creating a New User

Sometimes it's easier to create a user from CLI:

```
# fossil user new USERNAME
CONTACT-INFO PASSWORD
```

Reset Password

Forgetting the root password is a nightmare. However, you can change root or any user's password by:

```
# fossil user password USERNAME
PASSWORD
```

Convert Git to Fossil

To import a Git repository into Fossil, issue these commands:

```
# cd git-repo

# git fast-export --all | fossil
import --git new-repo.fossil
```

Conclusion

Beside Fossil's advanced features, installing Fossil and FreeBSD on RPI3 is minimalistic. Fossil lets you create your own professional Distributed Version Control server that can compete with something like Git.

Useful Links

https://www.fossil-scm.org/index.html/doc/trunk/www/index.wiki

https://www.fossil-scm.org/xfer/doc/trunk/www/inout.wiki

https://www.fossil-scm.org/xfer/doc/trunk/www/fossil-v-git.wiki

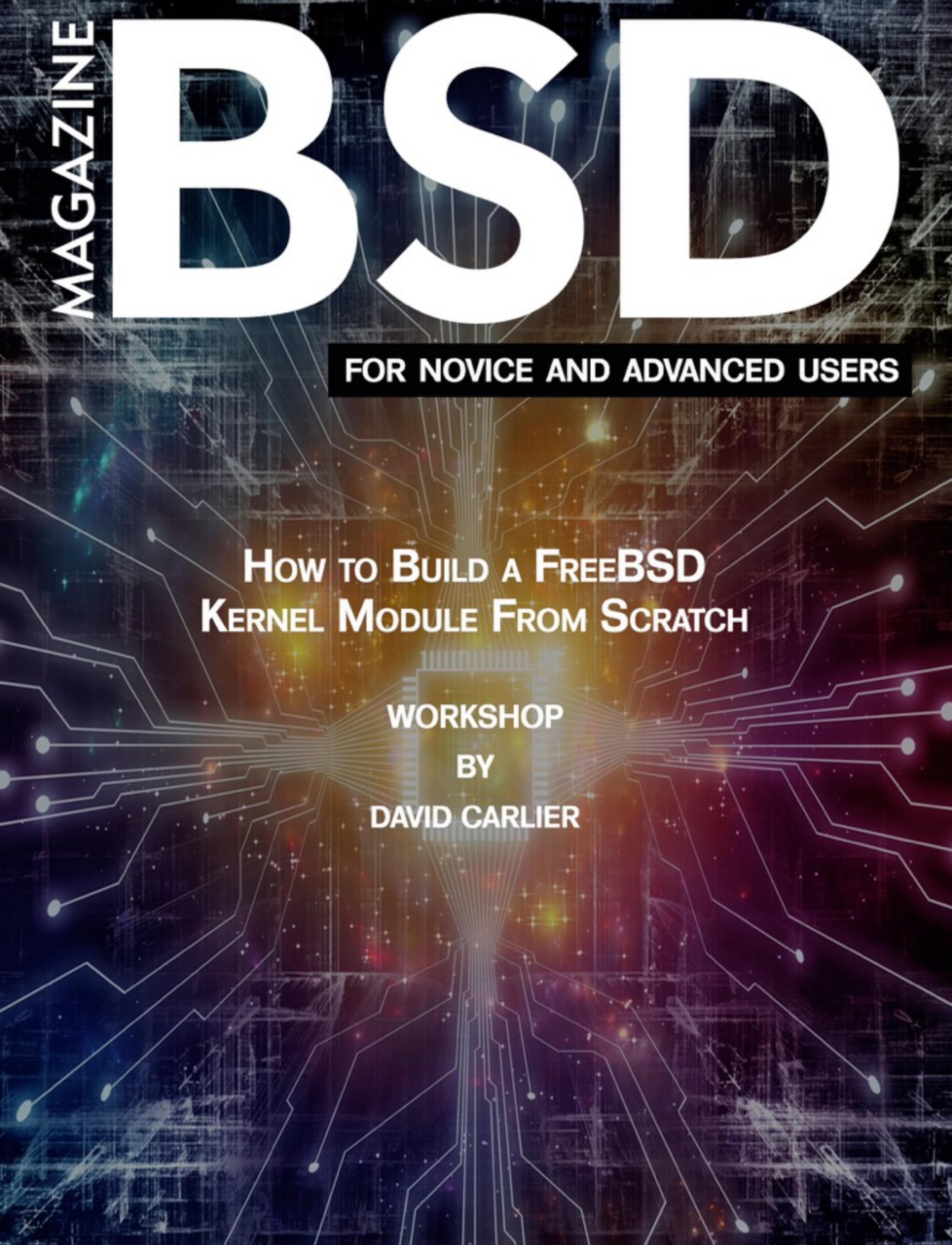
http://chiselapp.com/

Meet the Author

Abdorrahman Homaei has been working as a software developer since 2000. He has used FreeBSD for more than ten years. He became involved with the meetBSD dot ir and has performed serious training on FreeBSD. He started his company (etesal amne sara tehran) in Feb 2017, and it is based in Iran Silicon Valley.

Full CV: **http://in4bsd.com**

His company: **http://corebox.ir**



MAGAZINE

BSD

FOR NOVICE AND ADVANCED USERS

How to Build a FreeBSD Kernel Module From Scratch

WORKSHOP

BY

DAVID CARLIER

Expert Speak by E.G.Nadhan

What Blockchain Can Do Depends on Who You Are

Well, you know how excited I am about Blockchain. Hence, when I met with a distinguished group of CIOs at the Gartner ITXPO 2017, I engaged in a meaningful discussion on what Blockchain could mean to them in the context of their enterprise. The group of CIOs and IT leaders came from a diverse spectrum of industries. While it is true that several cogent points emerged from the open discussion that ensued, I was intrigued by the unique perspectives that each leader brought to the meeting. While Blockchain is being viewed as a technology that has the potential across several industries, what you do with Blockchain depends upon the entities that are most germane to the business need of the enterprise. Or simply put, **what Blockchain can do depends on who you are!**

Early adopter. There was certainly a fair share of skepticism about how real the application of this technology is today. This was why the CIO from the financial sector stepped up and asserted that Blockchain is almost an effective settlement of trades in a secure, expeditious fashion. They are already using it and clearly need to do it. Welcome to the domain where this concept originated in the first place.

Game-on! The CIO of a company that owns one of the football clubs in the National Football League was all about the legitimacy of the tickets being used. Are the tickets falling into the wrong hands through improper channels? Can Blockchain technologies enable the integrity of the system as a whole so that the right number of tickets is available for their customers?

Controlled transparency. I could not help bring up the metaphor of the Open-Source community where there are multiple pairs of eyes that constantly monitor new code segments being injected, sort of the Wikipedia for software. Blockchain -- with a distributed ledger that no single individual controls -- is similar in many respects to the collaborative culture of the Open-Source ecosystem of innovation.

Self-monitoring service. Given the ability of the Blockchain ecosystem to proactively prevent and thus, control fraud, the question came up about the ongoing need for a formal auditing service. Well, there is always going to be the human element. Computers can never be data scientists. Auditors can serve more as reviewers and validate what is detected by the Blockchain solution. No, this technology is not going to put auditing firms out of business! In a similar vein, checkout this article: Germany has more industrial robots than US but they haven't caused job losses!

A healthy outlook. Healthcare industry is rich with scenarios where blockchain can help. Checkout this futuristic perspective. This is more so because it has the classic intersection of two key entities, the **drug** impacting the wellbeing of the **patient**. Both entities are rich with data that should be secured, but available to the right parties at the right time. Blockchain technologies could be used to ensure that this type of critical information, whether it is related to the composition of the drug itself or the confidential data of the patient, does not fall into the wrong hands. It should not take patients to open data sharing!

There you have it.

The business of the enterprise determined the manner in which each CIO perceived the application of Blockchain in their industry. The fact that such a diverse array of CIOs took the time to converge and share their thoughts at the Gartner symposium is indicative of a high level of interest in this technology.

What about you? Where is your enterprise with Blockchain? What are some of the areas where you see it being applied, and how?

What is your way of applying Blockchain in your domain?

Chime in with your thoughts and let’s meet-up right here!

About Me

E.G.Nadhan is Chief Technology Strategist for the Central Region at Red Hat. He provides thought leadership on various concepts including Cloud, Big Data, Analytics and the Internet of Things (IoT) through multiple channels including industry conferences, Executive Round tables as well as customer specific Executive Briefing sessions. With 25+ years of experience in the IT industry selling, delivering and managing enterprise solutions for global corporations, he works with the executive leadership of enterprises to innovatively drive Digital Transformation with a healthy blend of emerging solutions and a DevOps mindset. Follow Nadhan on Twitter and LinkedIn.

Interview with Rob Somerville

Can you tell our readers about yourself?

I am a long-haired, bearded, Scotsman who has a passion for technology, chess, writing, cookery, backgammon, and my family. The first three passions I have had since a teenager, the latter three have come along on the journey. I was the typical bespectacled, geeky kid, taking everything apart and building electronic projects out of scrap recycled parts from when I was young. My father inspired my interest in electronics from when I was 5 when he demonstrated how a bulb and battery worked. My best electronics creation was when I was 13 - a high voltage generator which threw out enough Kv to light a fluorescent tube placed on the bald head of my father! I started off working as an electronics technician when I left school, and moved into IT when digital electronics started eclipsing analogue systems in the '80's. Professionally though, I didn't enter the field until DOS 3.0 and the IBM XT/AT generation machines were unveiled.

When was your first contact with a computer and what attracted you at first ?

Difficult to say really, as it was such a long time ago. I messed around with friend's BBC Micro's, whenever I got the chance as I was not wealthy enough at that time to actually own my own computer. Eventually I purchased a Sinclair ZX Spectrum - which I treasured - when I was 20 (over 36 years ago). Programming always appealed to me, and the power and graphics available on the BBC were a world apart from the limited programming functionality available on the HP25 RPN calculators that I was always experimenting with at work. The attraction was I could translate my ideas into reality via a keyboard.

How you first got involved with programming and the FreeBSD world?

My first job in IT was as a systems administrator / developer in the mid '80's, responsible for a Novell Netware system running 5 clients and a server. Part of that role was developing an in-house database which was a Pick based system using Advanced Revelation. I had to learn RBASIC from the ground up after a 5 day course, which was a steep learning curve. Through the years, I have dabbled with everything from Assembler, Basic, DOS batch files and server scripts through to Pascal, C, Bash, Perl, PHP and SQL to name but a few. It wasn't until the late '90's I became involved with the Open-Source movement with the rise in popularity of the Internet and the World Wide Web. I was immediately struck by the potential of Linux and the other *BSD's. While I had always been a supporter of Public Domain software, Redhat Linux and *BSD was the turning point for me, as it was the first time a complete O/S and access to a huge range of software that was available on a set of floppies or a CD or two. It was love at first sight.

What is your preferred operating system to program on, and why?

For me, it has just got to be Unix or a derivative, be it Linux or *BSD. I love the power a developer has at the command line to access O/S tools across the family, and while there may be a few differences in switches and directory structure, the principles remain pretty much identical. The plethora of tools and software available “out of the box” at the time was in a different league from anything Microsoft had to offer. Multiple programming languages, a built-in compiler and world class systems tools and diagnostics in those days were rare and expensive beasts. Microsoft has caught up now to some extent, but the command line and the Unix philosophy will always remain close to my heart.

Do you remember your very first development? How do you consider it now?

As a developer, you are always learning by your mistakes so I can honestly admit - whatever it was – probably, it was very crude and nowhere near a well-structured and tight code, something I would write today. Professionally, most of my coding started with RBASIC and I know I was very pleased with the first project I put together unaided for the AREV system I managed – an online help system   that I managed to sell to a fellow developer at a small profit.

What was your best work? Can you tell us the idea behind it? What was its purpose?

Conceptually, it was a system to integrate Advanced Revelation with Reuters Textline to allow researchers to enter queries as a search term in the same way Google works today. I didn’t have enough programming knowledge at the time so I needed a lot of help from a fellow developer to build the skeleton. It was a typical integration / automation scenario regularly encountered by developers, how to make a system more efficient and streamlined for the users. It brought greater efficiencies to the desktop, rather than manually writing out requests for me to enter into Textline, the system would store them, send them down-line when the telco charges were cheap, and distribute them in time via the database for the researcher coming to work in the morning. It also saved on mountains of paper, as prior to the development of the system the results were printed out on a noisy and earth-shaking line printer. This inspired me, and I have been fascinated in “glueing” disparate systems together ever since.

Which kind of development and programming languages are your preferred?

PHP and Perl are my favourites, as well as shell scripts and SQL. I have never managed to get my head around object oriented code, and while I see the benefits, I still think very procedurally. Multi-threading and race conditions give me nightmares.

What tools do you use most often, and why?

Grep, Midnight Commander and a hex editor are the essential tools for me. I would be lost without them. As far as programming IDE is concerned, I like Komodo. Although I am just as much at home with VI (or at a push Emacs) provided debugger support is available. A lot of my work involves data analysis, and Grep is fantastic at extracting textural information from files e.g. Apache server logs. MC is a great shell to use when copying and updating files. Also, a hex editor is invaluable when investigating security incidents and infected files etc. I use Git on big projects for version control, and frequently MySql and Postgres clients, packet sniffers and Diff as required. The Man pages are also invaluable, as with the passing years, I forget more and more.

What was the most difficult and challenging implementation you've done so far? Could you give us some details?

Working in government and integrating the various disparate systems has probably been the most complex challenge I've encountered. I built a large Geographical Information System for my previous employer entirely on an Open-Source platform, with a fellow Open Source developer who wrote the custom front-end. The software was the least of our problems. It worked pretty much flawlessly as expected, and integrating this with Drupal was very straightforward. The biggest issue was the sheer size of the datasets and the amount of bandwidth required for this to work smoothly for the end users. BSD and Linux have consistently risen to every technological challenge, and I have managed to reliably integrate some of the most closed systems imaginable using these platforms. More often than not, data size, quality and proprietary systems pose many problems for me.

Do you use your own development works professionally or are they hobbies ?

Both. If I need to write some ad-hoc code, I will do it. I needed to recover a friend's hard disk recently, and using a combination of Photorec, Testdisk and a custom PHP script, I managed to recover all of their data and sort it all into various logical categories by file type etc. I was rather pleased with the results, as the hard disk was unmountable and sounded like a loose bag of rocks inside a cement mixer! It took a few days though.

What future do you see for FreeBSD and other OSes? Can you tell us about your favourite features in the new releases?

I think it will be the continuing focus on security which will be the top priority. The debates, divisions and forks over such issues as to how an init system should be designed, the best desktop etc. will carry on as normal. But I think there will be less "revolutionary" innovation since developers grapple with the implications of legacy libraries and code, the growth of more automated, widespread and sophisticated attacks, the implications of the cloud and the widespread virtualisation etc. This will lead, in turn, to a more "OpenBSD" approach to security, focussing on the code rather than bells and whistles and user features.

I really like the move to the new binary pkg system in the 10.x version of FreeBSD. As to the Linux distros, Timeshift is a great new feature that is rapidly maturing, allowing users to make snapshots of all system files.

Do you have any specific goals for the rest of this year?

I am starting to write a book based on my life experiences, very much a philosophical commentary. A few friends have also suggested I write and design a cookbook. Therefore, expanding my writing skills is very much on the cards for me in 2018, be it for clients as a professional coder or an author.

What's the best advice you can give to the BSD magazine readers?

Consider what your passion and dreams are, and live for them. I have been truly fortunate in that I managed to leave school at a time in history where I could enter my chosen profession with ease, which opened doors for me into the IT industry, and eventually writing. It will be hard work, and not always easy, but provided you constantly commit yourself to learn from your mistakes, it will always be highly rewarding.

Thank you

MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS

UNIX

HOW TO OPTIMALLY USE THE RESOURCES OF YOUR MACHINE TO MINIMIZE THE RUN TIME OF YOUR PROGRAM

MARK SITKOWSKI
C.ENG, M.I.E.E DESIGN SIMULATION SYSTEMS LTD

ISSN 1898-9144

Spectre and Meltdown have hit the lead, copper, silver, and gold streets of the Internet highway. We knew this was on the cards. What now?

Rob Somerville

Unlike the many players in this scenario, I am going to start with an apology. The next 1000 words or so are going to be fiery and passionate, unlike a rant. The Oxford English Dictionary defines a rant as follows – to speak or shout at length in an angry, impassioned way. I refuse to be angry here, and 1000 words can hardly be considered lengthy once we have to consider all the variables, but more specifically, the context requires more than a few words.

There is a phrase we have in the UK, used when a spoilt child eats too much sugar or spends too much time in the sun. “There will be tears before bedtime” is the parental refrain, and such was my viewpoint in the late ‘80’s, early ‘90’s when peer-to-peer networking took off on the Microsoft platform. I said this coming from a strong engineering background, understanding how germanium and silicon work. Part of the reason I bailed from working in the electronics industry to IT was simple–I realized that the analogue age was dying and the future was zeros and ones. Basically, if I stayed on the same track not only would I become unemployed fairly quickly, but the skills I had achieved–understanding all the “bad things” that come with analogue territory, would be as naught. I realized at this point that the genie was out the bottle, but pulled down a visor of cynicism just in case. While the benefits of technological expansion were clear, the siren calls of “let's slow this down a bit” were steamrollered with IPO’s, PR hype and the worst form of patronization any engineer or scientist will encounter - “Trust the crowd”.

So here we are. We have a design flaw, going back possibly 20 years, which exposes all of a CPU’s dirty laundry to the world, and as developers, we are expected to patch by coding around with a potentially significant hit to system performance. Cross-platform. Cross O/S. You really couldn’t hand the black hats a bigger gift unless you gave them a free pass to Area 51 or whatever. Those clued up, as far as the guts of technology, are both more aware and concerned. The Spectre and Meltdown are just mistakes (well until at least proven in a court otherwise). We have a bigger torrent coming on-stream–failure due to ignorance or motive.

I don't honestly believe the designers of the relevant CPU's affected allowed this bug to get past them deliberately. Most engineers I have encountered to date have been scrupulous individuals. Some chastised mercilessly for their attention to detail while others sadly fell into the "good enough" brigade. The latter was immediately identified by the former and suitably ostracised. I remember the conversation I had with a senior engineer who was slating in his opinion of a junior engineer regarding his choice of a capacitor in a piece of critical circuitry. The only reason I was party to this potentially fatal career decision was that I identified a critical issue during testing of the prototype. I cannot remember what happened to the engineer in question.

The whole issue boils down to one single point of failure – trust. The bigger issue, while the IT industry scrambles for excuses, best PR and the like – is how far we can depend on the supply chain. If Intel *et al* can push the responsibility over to software developers, manufacturers, integrators, and distributors to fix this problem, the worse the problem will get in the long term. Every chip fabricator and designer on the planet will have *carte blanche* to say, "Oh well, our chips had a major error flaw but 99.999% of the time they worked OK". We are looking at a broadside attack on Western civilization and values here. If the chip designers are allowed to walk away from this, anything goes. While Intel is working with software developers to integrate solutions into newer chips, this hardware patch will, in reality, only be available to the few. Forget Joe Q PC. Forget the chip recall on the exposure of the FDIV bug. Software cures all ills.

Human beings make mistakes. This is a fundamental understanding in the bedrock of our society, our law, our religion, and our way of life. We work our ways around things, the embarrassing moment, the dropped glass of red wine on the carpet, the missed appointment. Even in our most intimate moments, we can embarrass ourselves. That is part of what being human is all about – we work our way around these embarrassments. However, this scenario is different in that we are not talking about human scale, we are talking global implications. Even with the best PR in the world, a 5-year chip recall would damage the major CPU manufacturers to the point of no return. Once you get the lawyers involved, exponential are the fees, bureaucracy, delays, and relentless arguments. So, it is no surprise that the hardware industry says "not guilty" and flaunts a license or legal agreement in your face saying "we are not responsible".

So, who should carry the can? Personally, I hold my greatest wrath for those who don't get it. Those who worship at the altar of technology. Those who don't understand the implications. The people who are willing to commit to crap projects purely on the basis of remuneration, without having enough character to see through to the medium or long-term repercussions. Guns for hire that will not be seen again.

I have a great deal of respect for Intel and others in this very unfortunate boat. Unlike Microsoft, and others in the software sector, they have had to put their money where their mouth is to develop their business. Chip fabrication and production is closer to satellite technology than telling your mouse where to go. The skills and engineering disciplines that are required are quite dissimilar but meet at the nexus of innovation, creativity, and focus. And that is where this tragedy will be played.

We need a Ralph Nader moment. Unsafe at Any Speed, The Designed-In Dangers of the American Automobile, was a best seller in 1966. It changed cars forever. More importantly, it changed an industry.

ServerU Netmap-L800
www.ServerU.us
Server U
CONSOLE
USB
RESET
0
1
2
3
4
5