MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS

Bitcoin Full Node On FreeBSD

OpenLDAP Directory Services in FreeBSD (II)

Applications on Centralized Management using NIS+

Page Checksum Protection in PostgreSQL

OPENBSD ROUTER WITH PF

My Switch to OpenBSD, First Impressions

Celebrating Our 100TH Issue

ISSUE 12/2017 (100)
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**

Copyright © 2017 iXsystems. FreeNAS is a registered trademark of iXsystems, Inc. All rights reserved.

DON'T DEPEND ON CONSUMER-GRADE STORAGE.

intel inside™

KEEP YOUR DATA SAFE!

Buy from amazon.com

USE AN ENTERPRISE-GRADE STORAGE SYSTEM FROM IXSYSTEMS INSTEAD.

The FreeNAS Mini: Plug it in and boot it up — it just works.

- Runs FreeNAS, the world's #1 software-defined storage solution
- Unifies NAS, SAN, and object storage to support multiple workloads
- Encrypt data at rest or in flight using an 8-Core 2.4GHz Intel® Atom® processor
- OpenZFS ensures data integrity
- A 4-bay or 8-bay desktop storage array that scales to 48TB and packs a wallop
- Backed by a 1 year parts and labor warranty, and supported by the Silicon Valley team that designed and built it
- Perfectly suited for SoHo/SMB workloads like backups, replication, and file sharing
- Lowers storage TCO through its use of enterprise-class hardware, ECC RAM, optional flash, white-glove support, and enterprise hard drives

And really — why would you trust storage from anyone else?

iX systems™

Call or click today! **1-855-GREP-4-IX** (US) | **1-408-943-4100** (Non-US) | **www.iXsystems.com/Freenas-Mini** or purchase on Amazon.

Intel, the Intel logo, Intel Inside, Intel Inside logo, Intel Atom, and Intel Atom Inside are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

Editor's Word

MAGAZINE BSD

Dear Readers,

Wow, it's a wrap for 2017! As you take stock of all the monthly issues, we hope that some, if not all of your dreams were fulfilled this year. Additionally, we would like to encourage you to start planning for 2018, in hopes for a better and more interesting year. I know that most of you are spending time with your family and friends this New Year's Eve. Although such an evening repeats itself yearly, it is undoubtedly an amazing experience for all. The evening is magical for each one of us. It brings hope and joy to our hearts. It shows us how we should live every day and what we should incorporate in our lives. I hope that you will have a great night, and may the same energy you feel today take you through the next year. Hence, allow me to make the following wishes to you:

The New Year is the time of unfolding horizons and the realization of dreams.

May you rediscover new strength and garner faith with you,

be able to rejoice in the simple pleasures

that life has to offer and put a brave front for

all the challenges that may come your way.

Wishing you a lovely New Year.

As usual, we have prepared a solid amount of good readings in this month tailored for you. You will not only meet new people who love the BSD world but also read mind-refreshing articles. Therefore, I invite you to check a list of the articles on the next page. Lastly, a big thank you to all our reviewers for their valuable input on how to make the articles better.

See you in 2018!

Enjoy reading,
Ewa & The BSD Team

Table of Contents

In Brief

PostgreSQL

FreeBSD

Blog Presentation

Column

In Brief

End-of-Life for FreeBSD 11.0

A few days ago, the FreeBSD Team announced END-of-LIFE for FreeBSD version 11.0.

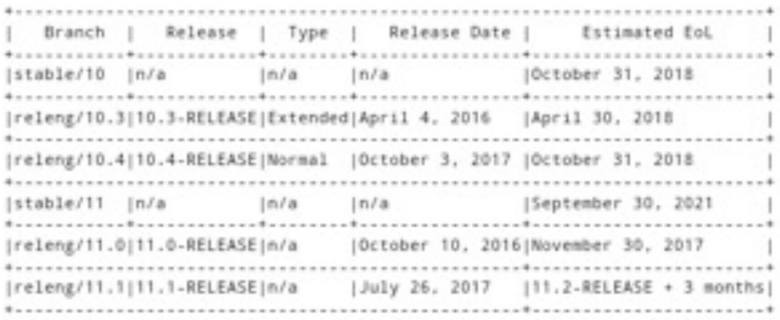
So if you are still on 11.0, you should consider upgrading to a newer release. This way, you will still be receiving updates.

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

Dear FreeBSD community,

As of Nov 30, 2017, FreeBSD 11.0 reached end-of-life and is no longer supported by the FreeBSD Security Team. Users of FreeBSD 11.0 are strongly encouraged to upgrade to a newer release as soon as
possible.

The currently supported branches, releases and their expected end-of-life dates are:



Branch	Release	Type	Release Date	Estimated EoL
stable/10	n/a	n/a	n/a	October 31, 2018
releng/10.3	10.3-RELEASE	Extended	April 4, 2016	April 30, 2018
releng/10.4	10.4-RELEASE	Normal	October 3, 2017	October 31, 2018
stable/11	n/a	n/a	n/a	September 30, 2021
releng/11.0	11.0-RELEASE	n/a	October 10, 2016	November 30, 2017
releng/11.1	11.1-RELEASE	n/a	July 26, 2017	11.2-RELEASE + 3 months

As a reminder, FreeBSD changed the support model as of 11.0-RELEASE.

For additional information, please see https://lists.freebsd.org/pipermail/freebsd-announce/2015-February/001624.html

Please refer to https://security.freebsd.org/ for an up-to-date list of
supported releases and the latest security advisories.

- --
The FreeBSD Security Team

Source:
https://www.mail-archive.com/freebsd-announce@freebsd.org/msg00822.html

Kernel ASLR on amd64

Maxime Villard has completed a Kernel ASLR implementation for NetBSD-amd64, making NetBSD the first BSD system to support such a feature. Simply said, KASLR is a feature that randomizes the location of the kernel in memory, making it harder to exploit several classes of vulnerabilities, both locally (privilege escalations) and remotely (remote code executions).

Source:
https://blog.netbsd.org/tnf/entry/kernel_aslr_on_amd64

arm64 Platform Supported

OpenBSD's ARM64 support is now considered officially supported. Theo de Raadt committed this change:

```
CVSROOT:        /cvs
Module name:    www
Changes by:
deraadt@cvs.openbsd.org  2017/12/07
12:00:12

Modified files:
      .                : plat.html

Log message:
graduate arm64 to supported; having
syspatch it is even beyond some
other systems
```

Source:
https://undeadly.org/cgi?action=article;sid=20171208082238

DTrace and ZFS Update

Chuck Silvers worked to update the DTrace and ZFS code. The code that has been used so far was originated from the OpenSolaris code-base.

Chuck Silvers worked to migrate over to FreeBSD's ZFS/DTrace code thanks to that many fixes and enhancements for the ZFS file-system, adds mmap() support to ZFS on NetBSD, and the DTrace code re-base can be done.

NetBSD 8.0 is the next major feature release currently under development.

Source: http://mail-index.netbsd.org/tech-kern/2017/12/07/msg022694.html

DragonFly 5.0.2 Released

DragonFly version 5 has been released, including the first bootable release of HAMMER2. Version 5.0.2, the current version, came out 2017/12/04.
DragonFly belongs to the same class of operating systems as other BSD-derived systems and Linux. It is based on the same UNIX ideals and APIs, and shares ancestor code with other BSD operating systems. DragonFly provides an opportunity for the BSD base to grow in an entirely different direction from the one taken in the FreeBSD, NetBSD, and OpenBSD series.
DragonFly includes many useful features that differentiate it from other operating systems in the same class.
The most prominent one is HAMMER, our modern high-performance file system with built-in mirroring and historic access functionality.
Virtual kernels provide the ability to run a full-blown kernel as a user process for the purpose of managing resources or for accelerated kernel development and debugging.
The kernel uses several synchronizations and locking mechanisms for SMP. Much of the work done since the project began has been in this area. A combination of intentional simplification of certain classes of locks to make more expansive subsystems less prone to deadlocks, and the rewriting of nearly all the original codebase using algorithms designed specifically with SMP in mind, has resulted in an extremely stable, high-performance kernel that is capable of efficiently using all CPU, memory, and I/O resources thrown at it.
DragonFlyBSD has virtually no bottlenecks or lock contention in-kernel. Nearly all operations can run concurrently on any number of CPUs. Over the years, the VFS support infrastructure (namecache and vnode cache), user support infrastructure (uid, gid, process groups,and sessions), process and threading infrastructure, storage subsystems, networking, user and kernel memory allocation and management, process fork, exec, exit/teardown, timekeeping, and all other aspects of kernel design, have been rewritten with extreme SMP performance as a goal.
DragonFly is uniquely positioned to take advantage of the wide availability of affordable Solid Storage Devices (SSDs), by making use of swap space to cache filesystem data and meta-data. This feature commonly referred to as "swapcache", can give a significant boost to both server and workstation workloads, with a minor hardware investment.
The DragonFly storage stack comprises of robust, natively written AHCI and NVME drivers, stable device names via DEVFS, and a partial implementation of Device Mapper for reliable

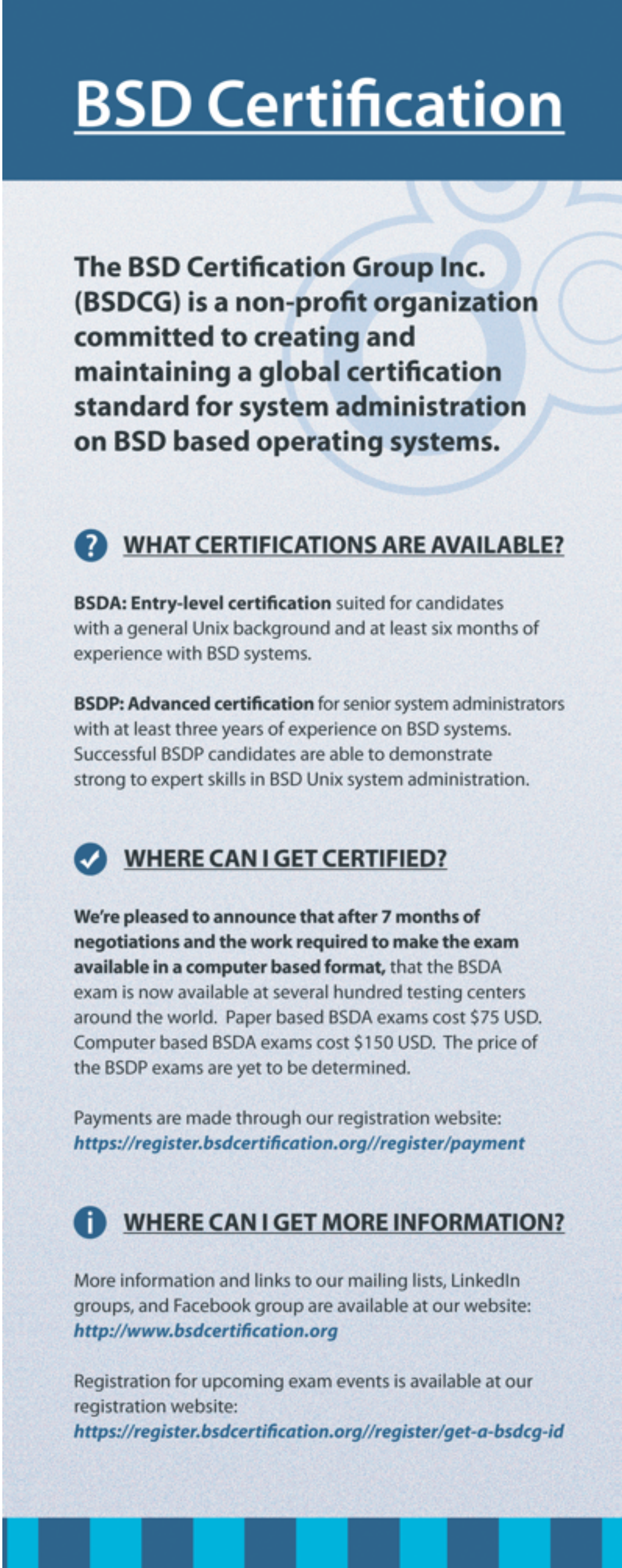
volume management and encryption.

Some other features that are especially useful to system administrators are a performant and scalable TMPFS implementation, an extremely efficient NULLFS that requires no internal replication of directory or file vnodes, our natively written DNTPD (ntp client) which uses full-bore line intercept and standard deviation summation for highly-accurate timekeeping, and DMA, designed to provide low-overhead email services for system operators who do not need more expansive mail services such as postfix or sendmail.

A major crux of any open-source operating system is third party applications. DragonFly leverages the dports system to provide thousands of applications in source and binary forms. These features and more band together to make DragonFly a modern, useful, friendly and familiar UNIX-like operating system.

The DragonFly BSD community is made up of users and developers who take pride in an operating system that maintains challenging goals and ideals. This community has no reservation about cutting ties with legacy when it makes sense, preferring a pragmatic, no-nonsense approach to development of the system. The community also takes pride in its openness and innovative spirit, applying patience liberally and always trying to find a means to meet or exceed the performance of our competitors while maintaining our trademark algorithmic simplicity.

Source: https://www.dragonflybsd.org/



BSD Certification

The BSD Certification Group Inc. (BSDCG) is a non-profit organization committed to creating and maintaining a global certification standard for system administration on BSD based operating systems.

? WHAT CERTIFICATIONS ARE AVAILABLE?

BSDA: Entry-level certification suited for candidates with a general Unix background and at least six months of experience with BSD systems.

BSDP: Advanced certification for senior system administrators with at least three years of experience on BSD systems. Successful BSDP candidates are able to demonstrate strong to expert skills in BSD Unix system administration.

✓ WHERE CAN I GET CERTIFIED?

We're pleased to announce that after 7 months of negotiations and the work required to make the exam available in a computer based format, that the BSDA exam is now available at several hundred testing centers around the world. Paper based BSDA exams cost $75 USD. Computer based BSDA exams cost $150 USD. The price of the BSDP exams are yet to be determined.

Payments are made through our registration website:
https://register.bsdcertification.org//register/payment

i WHERE CAN I GET MORE INFORMATION?

More information and links to our mailing lists, LinkedIn groups, and Facebook group are available at our website:
http://www.bsdcertification.org

Registration for upcoming exam events is available at our registration website:
https://register.bsdcertification.org//register/get-a-bsdcg-id

FreeNAS 11.1 is Now Available for Download!

by The FreeNAS Development Team

FreeNAS 11.1 Provides Greater Performance and Cloud Integration

The FreeNAS Development Team is excited and proud to present FreeNAS 11.1! FreeNAS 11.1 adds cloud integration, OpenZFS performance improvements, including the ability to prioritize resilvering operations, and preliminary Docker support to the world's most popular software-defined storage operating system. This release includes an updated preview of the beta version of the new administrator graphical user interface, including the ability to select display themes. This post provides a brief overview of the new features.

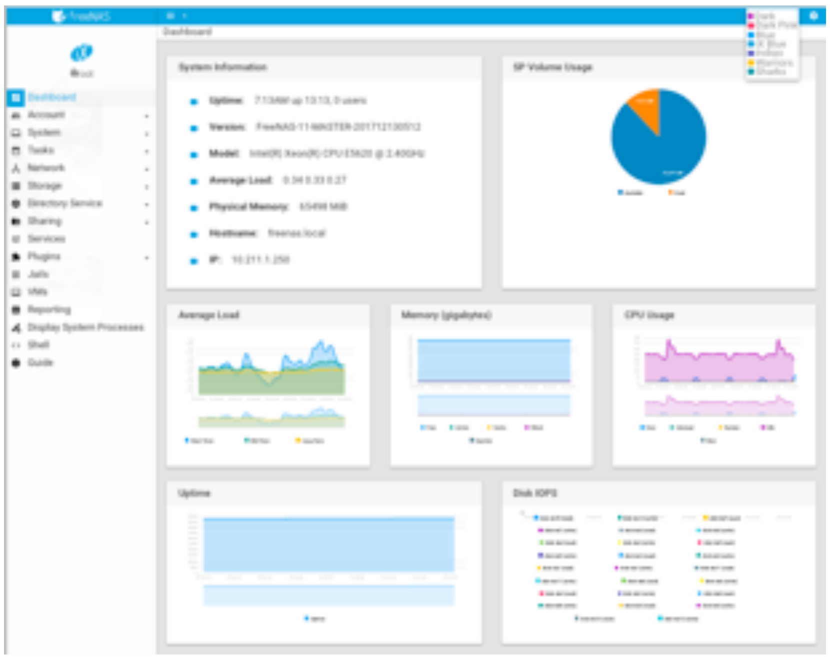
The base operating system has been updated to the STABLE version of FreeBSD 11.1, which adds new features, updated drivers, and the latest security fixes. Support for Intel® Xeon® Scalable Family processors, AMD Ryzen processors, and HBA 9400-91 have been added.

FreeNAS 11.1 adds a cloud sync (data import/export to the cloud) feature. This new feature lets you sync (similar to backup), move (erase from source), or copy (only changed data) data to and from public cloud providers that include Amazon S3 (Simple Storage Services), Backblaze B2 Cloud, Google Cloud, and Microsoft Azure.

OpenZFS has noticeable performance improvements for handling multiple snapshots and large files. Resilver Priority has been added to the Storage screen of the graphical user interface, allowing you to configure resilvering at a higher priority at specific times. This helps to mitigate the inherited challenges and risks associated with storage array rebuilds on very large capacity drives.

FreeNAS 11.1 adds preliminary Docker container support, delivered as a VM built from RancherOS. This provides a mechanism for automating application deployment inside containers, and a graphical tool for managing Docker containers. Please report any issues you encounter when beta testing this feature. This will assist the development team in improving it for the next major release of FreeNAS.

Finally, there are updates to the new Angular-based administrative GUI, including the addition of several themes. The FreeNAS team expects the new administrative GUI to achieve parity with the current one for the FreeNAS 11.2 release. To see a preview of the new GUI, click the BETA link on the login screen. Here is an example of the new GUI's main dashboard, with the available themes listed in the upper right corner.



System Information

- Uptime: 7:13AM up 13:13, 0 users
- Version: FreeNAS-11-MASTER-201712130512
- Model: Intel(R) Xeon(R) CPU E5620 @ 2.40GHz
- Average Load: 0.34 0.33 0.27
- Physical Memory: 65498 MiB
- Hostname: freenas.local
- IP: 10.211.1.250

The FreeNAS community is large and vibrant. We invite you to join us on the FreeNAS forum. To download **FreeNAS 11.1 RELEASE** and sign-up for the FreeNAS Newsletter, visit freenas.org/download.

TrueOS 17.12 Release

by Ken Moore

We are pleased to announce a new release of the 6-month STABLE version of TrueOS!

This release cycle focused on lots of cleanup and stabilization of the distinguishing features of TrueOS: OpenRC, boot speed, removable-device management, SysAdm API integrations, Lumina improvements, and more. We have also been working quite a bit on the server offering of TrueOS, and are pleased to provide new text-based server images with support for Virtualization systems such as bhyve. This allows for simple server deployments which also take advantage of the TrueOS improvements to FreeBSD such as:

Sane service management and status reporting with OpenRC

Reliable, non-interactive system update mechanism with fail-safe boot environment support.

Graphical management of remote TrueOS servers through SysAdm (also provides a reliable API for administering systems remotely).

LibreSSL for all base SSL support.

Base system managed via packages (allows for additional fine-tuning).

Base system is smaller due to the removal of the old GCC version in base. Any compiler and/or version may be installed and used via packages as desired.

Support for newer graphics drivers and chipsets (graphics, networking, wifi, and more)

TrueOS Version 17.12 (2017, December) is now available for download from the TrueOS website. Both the STABLE and UNSTABLE package repositories have also been updated in-sync with each other, so current users only need to follow the prompts about updating their system to run the new release.

We are also pleased to announce the availability of TrueOS Sponsorships! If you would like to help contribute to the project financially, we now can accept both one-time donations as well as recurring monthly donations which will help us advocate for TrueOS around the world.

Thank you all for using and supporting TrueOS!

~ *The TrueOS Core Team*

Notable Changes:

Over 1100 OpenRC services have been created for 3rd-party packages. This should ensure the functionality of nearly all available 3rd-party packages that install/use their own services.

The OpenRC services for FreeBSD itself have been overhauled, resulting in significantly shorter boot times.

Separate install of images for desktops and servers (server image uses a text/console installer)

Bhyve support for TrueOS Server Install

FreeBSD base is synced with 12.0-CURRENT as of December 4th, 2017 (Github commit: 209d01f)

FreeBSD ports tree is synced as of November 30th (pre-FLAVOR changes)

Lumina Desktop has been updated/developed from 1.3.0 to 1.4.1

PCDM now supports multiple simultaneous graphical sessions

Removable devices are now managed through the "automounter" service.

> Devices are "announced" as available to the system via *.desktop shortcuts in /media. These shortcuts also contain a variety of optional "Actions" that may be performed on the device.
>
> Devices are only mounted **while they are being used** (such as when browsing via the command line or a file manager).
>
> Devices are automatically unmounted as soon as they stop being accessed.
>
> Integrated support for all major filesystems (UFS, EXT, FAT, NTFS, ExFAT, etc..)
>
> NOTE: Currently, the Lumina desktop is the only one which supports this functionality.

The TrueOS update system has moved to an "active" update backend. This means that the user will need to start the update process by clicking the "Update Now" button in SysAdm, Lumina, or PCDM (as well as the command-line option). The staging of the update files is still performed automatically by default, but this (and many other options) can be easily changed in the "Update Manager" settings as desired.

Known Errata:

[VirtualBox] Running FreeBSD within a VirtualBox VM is known to occasionally receive non-existent mouse clicks – particularly when using a scroll wheel or two-finger scroll.

Quick Links:

TrueOS Forums

TrueOS Bugs

TrueOS Handbook

TrueOS Community Chat on Telegram

Become a Sponsor!

Versions of common packages:

NOTE: *The "STABLE" branch effectively locks 3rd-party package versions for its 6-month lifespan. The "UNSTABLE" branch provides rolling updates to all packages on a regular basis.*

Web Browsers:

Firefox: 57.0.1

Firefox-ESR: 52.5.0

Iridium: 58.0

Chromium: 61.0.3163.11

Palemoon: 27.6.2

QupZilla: 2.1.2 (Qt5) 1.8.9 (Qt4)

SeaMonkey: 2.49.1

Desktop Environments:

Lumina: 1.4.1

KDE: 4.14.3

MATE: 1.18.0

GNOME: 3.18.0

Cinnamon: 2.4.6

XFCE: 4.12

LXDE: 1.0

Databases:

PostgreSQL: 9.2, 9.3, 9.4, 9.5, 9.6, 10.1

MySQL: 5.5.58, 5.6.38, 5.7.20, 8.0.2

SQLite: 2.8.17, 3.21.0

Computing Languages:

Python: 2.7, 3.4, 3.5, 3.6

Ruby: 2.2, 2.3, 2.4

Perl: 5.22, 5.24, 5.26, 5.27

Rust: 1.22.1

Go: 1.4.3, 1.9.2

Compilers/Tools:

Clang: 3.3, 3.4.2, 3.5.2, 3.8, 6.0.d20171113

LLVM: 3.3, 3.4.2, 3.5.2, 3.8.1, 3.9.1, 4.0.1, 5.0.0, 6.0.d20171113

GCC: 4.6.4, 4.7.4, 4.8.5, 4.9.4, 5.5.0, 6.4.0, 6.4.1.s20171129, 7.2.0, 7.2.1.s20171123, 8.0.0.s20171126

Automated Deployment:

Puppet: 4.10.9, 5.3.3 (server: 2.7.2, 5.1.0 ; database: 4.4.0, 5.1.3)

Ansible: 1.9.6, 2.4.2.0

Salt: 2017.7.2

Other Applications/Utilities:

Libreoffice: 5.3.7

Apache OpenOffice: 4.1.4

Nginx: 1.12.2, 1.13.7

Apache: 2.4

Git: 2.15.1

GitLab: 10.1.4

Subversion: 1.8.19, 1.9.7

Page Checksum Protection in PostgreSQL

PostgreSQL does support a feature called page checksum that, if enabled at a cluster-wide level, protects the database from data corruption on the disk. The protection does not involve automatic data recover, rather a way to discard a piece of data that is considered no more reliable. In this short article, readers will see the effect of data corruption and how PostgreSQL reacts to such event.

You will learn

- **How to enable *page checksums***
- **What page checksum protects you from**
- **How to simulate a page corruption**
- **How to erase the damaged page**

You need to know

- **How to interact with a PostgreSQL (9.6) database**
- **How PostgreSQL sets up disk layout**
- **How to write and run a small Perl program**

PostgreSQL supports the *page checksum* feature since version 9.3; such feature allows the cluster to check for every *checked-in* data page to determine if the page is reliable or not. *A reliable page is a page that has not been corrupted during the path from memory to the disk (writing data to the disk) or the opposite (reading back the data)*.

As readers probably know, a data corruption can happen because of a bug or a failure in the disk controller, in the memory, and so on. What is the risk of a data corruption from a PostgreSQL point of view? A corrupted data page contains wrong tuples such that the data within the tuples is possibly wrong. Using such wrong data could make a single SELECT statement to report wrong data or, worst, such data can be used to build up other tuples and therefore "import" corrupted data within the database.

It is important to note that the *page checksum* feature does not enforce the database consistency: the latter is provided by the *Write Ahead Logs (WALs)* that have always been strongly protected from corruption with several techniques including a checksum on each segment. But while WALs ensure that data is made persistent, they don't protect you from a silent data corruption that hits a tuple (or alike), and again this "silent" corruption will be checked in the database in future.

What can the database do when a corruption in a data page lays around? There are two possible scenarios:

The data page is checked in and used as if it was reliable (i.e., the corruption is not detected at all);

The data page is discarded. Therefore, the data contained in it is not considered at all. Without *page checksums* , PostgreSQL will default to scenario 1), that is the detection is not perceived. Hence, possible corrupted data (e.g., a tuple, a field, a whole range of an index or table) is used in live operations and can corrupt other data.

MAGAZINE BSD

Editor in Chief:

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

Contributing:

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

All trademarks presented in the magazine were used only for informative purposes. All rights to trademarks presented in the magazine are reserved by the companies which own them.

With *page checksums* enabled, PostgreSQL will discard the page and all the data within it. For instance all the tuples of a table stored in such a page. Is it the administrator’s duty to decide what to do with such a data page? However, there is nothing PostgreSQL can do automatically since it is unknown what the real corruption is and what caused it.

Enabling page checksums

This feature can be enabled only at cluster initialization via `initdb: the` `--data-checksum` option instruments the command to enable data pages from the very beginning of the database. It is worth noting that a page checksum means a little more resource consumption to compute, store, and check the checksums on each page. More resource consumption means fewer throughputs.

Therefore, the database administrator must decide what is more important: performance or data reliability. Usually, the latter is the right choice for pretty much any setup. Therefore, it is important to understand that there is no protection at all against external data corruption without page checksum.

Consequently, to enable page checksums, initdb has to be run with the `--data- checksum` option. For instance a new cluster can be initialized as follows in Table 1:

```
$ initdb --data-checksum

-D /mnt/data1/pgdata/
```

Table 1. A new cluster can be initialized

Once the database has been instrumented as shown above, the user can interact with it in the same way as if page checksums was disabled. *The whole feature is totally transparent to the database user or administrator*.

Forcing a corruption

Readers must not execute the following steps in production!

The main aim of this section is to provide an insight on what happens when data is corrupted, but readers must understand that these four steps will deliberately destroy their data!

First of all, find out a table to corrupt. The following query will show you all the user tables order by descending page occupation. Hence the first table that will show up is a "large" table (see Table 2).

```
# SELECT relname, relpages,
reltuples, relfilenode FROM
pg_class

WHERE relkind = 'r'

AND relname NOT LIKE 'pg%' ORDER
BY relpages DESC;

-[ RECORD 1
]------------------------

relname      | event

relpages     | 13439

reltuples    | 2.11034e+06

relfilenode  | 19584

...
```

Table 2. All the user tables order by descending page occupation

As readers can see, the event tables have 13439 data pages, and a two millions tuple, so it is a large enough table to play with.

In order to find out the real file on the disk, it is important to get the path of the database which can be obtained with the following query (see Table 3).

```
# SELECT datname, oid

   FROM pg_database;


datname    |  oid

-----------+-------

postgres   | 12758

template1  | 1

template0 | 12757

luca       | 16389

foodb      | 18936

testdb     | 19554
```

Table 3. The path of the database

Since the event table is within the `testdb` database, the file on disk will be in

`$PGDATA/baase/19554/19584`. The utility `oid2name(1)` can be used to extract the very same information for databases and tables.

Corrupting a data page

The following simple Perl script will corrupt a data page (see Table 4).

```
#!env perl


open my $db_file, "+<", $ARGV[ 0
]

|| die "Impossibile aprire il
file!\n\n"; seek $db_file, ( 8 *
1024 ) + $ARGV[ 1 ], 0;


print { $db_file } "Hello
Corrupted Database!"; close
$db_file;
```

Table 4. The simple Perl script

The idea is simple:

- Open the specified data file (the one named relname in the previous SQL query);
- Move to the specified data page (please note that data pages are usually 8kb in size for a default PostgreSQL installation);
- Print out a string to corrupt the data;
- Close the file and flush to disk.

To perform the corruption, you have to launch the program with something like you can see on Table 5.

```
% sudo perl corrupt.pl
/mnt/data1/pgdata/base/19554/195
84 20
```

Table 5. To launch the program

The above will corrupt the 20th page of the event table. This can be done when the database is running or is stopped.

See the corruption

When you try to access the relation, PostgreSQL will clearly state that there is a corruption in the data page (see Table 6).

```
> SELECT * FROM event;

...

ERROR:  invalid page in block 20
of relation base/19554/19584
```

Table 6. PostgreSQL will clearly state that there is a corruption in the data page

So far, the database has no chance to recover the data, but at least *it is not checking in wrong data*!

Cleaning the damaged page

Since PostgreSQL can do nothing about data recovery, the only choice it has is to *zero* the damaged page. In other words, unless you really need the page to inspect the corruption, you can instrument PostgreSQL to *clean* the page and make it reusable (as a fresh new page). Data will still be lost, but at least you will not waste space on the disk. PostgreSQL provides the `zero_damaged_pages` option that can be set either in the configuration file *postgresql.conf* or in the running session. For instance, if a session performs the same extraction from the table with `zero_damaged_pages` enabled, PostgreSQL will not warn on anything (see Table 7).

```
# SET zero_damaged_pages TO 'on';
# SELECT * FROM event;
...
-- the query runs to the end
```

Table 7. PostgreSQL will not warn on anything

But in the cluster logs, there will be a notice about the cleanup of the page (see Table 8).

```
WARNING: page verification
failed, calculated checksum 61489
but expected 61452

WARNING: invalid page in block 20
of relation base/19554/19584;
zeroing out page
```

Table 8. The cleanup of the page

Moreover, the relation will have a page less than it was before (see Table 9).

```
# SELECT relname, relpages,
reltuples, relfilenode FROM
pg_class
WHERE relkind = 'r'
AND relname NOT LIKE 'pg%' ORDER
BY relpages DESC;

-[ RECORD 1]--------------------

relname  | event
relpages | 13438
reltuples    | 2.11015e+06
relfilenode | 19841

...
```

Table 9. The relation will have a page less

The number of pages is now 13438,a page less than the original size, 13439. *PostgreSQL did find out a page was not reliable and discarded it*.

Vacuum and autovacuum

The same effect would have taken place in the case where a vacuum was run against the table (see Table 10).

```
# SET zero_damaged_pages TO 'on';



# VACUUM FULL VERBOSE event;

INFO:    vacuuming "public.event"

WARNING: page verification
failed, calculated checksum 22447
but expected 19660

WARNING: invalid page in block 1
of relation base/19554/19857;
zeroing out page

INFO:    "event": found 0
removable, 2109837 nonremovable
row versions in 13437 pages
```

Table 10. A vacuum was run

However, do not expect autovacuum to work the same: it is a design choice to not allow autovacuum to clean up damaged pages, as you can read in the source code of the autovacuum process (see Table 11).

```
/*
     •    Force zero_damaged_pages OFF
in the autovac process, even if it is
set

     •    in postgresql.conf. We don't
really want such a dangerous option
being

     •    applied non-interactively.

*/

SetConfigOption("zero_damaged_pages",
"false", PGC_SUSET, PGC_S_OVERRIDE);
```

Table 11. The autovacuum process

As you can see, the option `zero_damaged_pages` is always set to false, so that an autovacuum process will not zero (or clean) a page. The idea is that such an operation is so important that an administrator should be notified and decide manually to perform a cleanup. In fact, a page corruption often means there is a problem with hardware (or filesystem or other software) that requires more investigation, and also a recovery from a reliable backup.

Conclusions

The *page checksum* feature allows PostgreSQL to detect silent data corruption that happened outside the WALs, i.e., on real data pages. The database cannot decide automatically how to recover such data. Therefore, the only choice left to the administrator is to clean up the wrong page or not. However, once a corruption is detected, PostgreSQL will refuse to *check-in* such a page thus protecting the other data pages from being polluted.

References

PostgreSQL website: www.postgresql.org
PostgreSQL Doc: www.postgresql.org/docs/

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 the only truly sane way of interacting with software and services. His website is available at http://fluca1978.github.io

OpenLDAP Directory Services in FreeBSD (II)

Applications on Centralized Management using NIS+

What you will learn:

- Installation and configuration methods for OpenLDAP 2.4 under FreeBSD
- Basic foundations of the new LDAP on-line configuration (OLC)
- Hardening LDAPv3 with SASL and TLS/SSL protocols
- Embedding of NIS+/YP into an LDAP server to provide centralized NIS+ support for UNIX computers
- Administration and basic tuning principles for LDAP servers

What you should already know:

- Intermediate UNIX OS background as end-user and administrator
- Some knowledge of UNIX authentication systems and NIS+/YP
- Experience with FreeBSD system package and FreeBSD ports
- Good taste for command-line usage

In the first part of this article, the main basic concepts around installation and configuration using the new dynamic online configuration (OCL) for FreeBSD systems have been presented. At this point the reader will understand the importance and benefits that the use of directory services provided by LDAP protocol. For the sake of simplicity, the second part is going to present a direct application to encapsulating a NIS+/YP centralized user authentication and management schema for an arbitrary number of servers and clients connected to a TCP/IP network. Additionally, we'll show a web-based administration tool that will make administering the OpenLDAP server easier.

LDAP Administration

Assuming our LDAP server was configured correctly, some typical operations to search for interesting values and attributes of the configuration are shown in illustration 4 are:

a) Finding LDAP Administrator Entry

In addition to the database directories, the BaseDN, RootDN, and user's password, the retrieved information also contains the name of indexes created by LDAP database to speed up the queries:

```
root@laertes:~ # ldapsearch -H ldapi:// -Y EXTERNAL -b "cn=config" "(olcRootDN=*)"
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
# extended LDIF
#
# LDAPv3
# base <cn=config> with scope subtree
# filter: (olcRootDN=*)
# requesting: ALL
#

# {0}config, config
dn: olcDatabase={0}config,cn=config
objectClass: olcDatabaseConfig
olcDatabase: {0}config
olcAccess: {0}to * by dn.exact=gidNumber=0+uidNumber=0,cn=peercred,cn=external
 ,cn=auth manage by * break
olcRootDN: cn=admin,cn=config

# {1}mdb, config
dn: olcDatabase={1}mdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcMdbConfig
olcDatabase: {1}mdb
olcDbDirectory: /var/db/openldap-data
olcSuffix: dc=bsd-online,dc=org
olcAccess: {0}to attrs=userPassword,shadowLastChange by self write by anonymou
 s auth by * none
olcAccess: {1}to dn.base="" by * read
```

```
olcAccess: {2}to * by * read
olcLastMod: TRUE
olcRootDN:
cn=admin,dc=bsd-online,dc=org
olcRootPW:
{SSHA}OaucFEx3RGbjVc+9JLXkfbDP8QBsRM
S1
olcDbCheckpoint: 512 30
olcDbIndex: objectClass eq
olcDbIndex: cn,uid eq
olcDbIndex: uidNumber,gidNumber eq
olcDbIndex: member,memberUid eq
olcDbMaxSize: 1073741824

# search result
search: 2
result: 0 Success

# numResponses: 3
# numEntries: 2
```

b) Modules and backends

Modules are widely used to extend LDAP functionality:

```
root@laertes2:~# ldapsearch -H
ldapi:// -Y EXTERNAL -b "cn=config"
-LLL -Q "objectClass=olcModuleList"
SASL/EXTERNAL authentication started
SASL username:
gidNumber=0+uidNumber=0,cn=peercred,
cn=external,cn=auth
SASL SSF: 0
# extended LDIF
#
# LDAPv3
# base <cn=config> with scope
subtree
# filter:
(objectClass=olcModuleList)
# requesting: ALL
#

# module{0}, config
dn: cn=module{0},cn=config
objectClass: olcModuleList
cn: module{0}
olcModulePath:
/usr/local/libexec/openldap
olcModuleLoad: {0}back_mdb

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
```

c) Backends included

```
root@laertes2:~# ldapsearch -H
ldapi:// -Y EXTERNAL -b "cn=config"
-LLL -Q
"objectClass=olcBackendConfig"

SASL/EXTERNAL authentication started

SASL username:
gidNumber=0+uidNumber=0,cn=peercred,
cn=external,cn=auth

SASL SSF: 0

# extended LDIF

#

# LDAPv3

# base <cn=config> with scope
subtree

# filter:
(objectClass=olcBackendConfig)

# requesting: ALL

#



# {0}mdb, config

dn: olcBackend={0}mdb,cn=config

objectClass: olcBackendConfig

olcBackend: {0}mdb



# search result

search: 2

result: 0 Success



# numResponses: 2

# numEntries: 1
```

d) Databases

```
root@laertes2:~# ldapsearch -H
ldapi:// -Y EXTERNAL -b "cn=config"
-LLL -Q "olcDatabase=*" dn

SASL/EXTERNAL authentication started

SASL username:
gidNumber=0+uidNumber=0,cn=peercred,
cn=external,cn=auth

SASL SSF: 0

# extended LDIF

#

# LDAPv3

# base <cn=config> with scope
subtree

# filter: (olcDatabase=*)

# requesting: dn

#



# {-1}frontend, config

dn:
olcDatabase={-1}frontend,cn=config



# {0}config, config

dn: olcDatabase={0}config,cn=config



# {1}mdb, config

dn: olcDatabase={1}mdb,cn=config
```

```
# search result
search: 2
result: 0 Success

# numResponses: 4
# numEntries: 3
```

By default, LDAP servers have three databases numbered from -1, 0, and 1. Their description is given now:

- olcDatabase={-1}frontend,cn=config: This entry is used to define the features of the special "frontend" database. This is a pseudo-database used to define global settings that should apply to all other databases (unless overridden).
- olcDatabase={0}config,cn=config: This entry is used to define the settings for the cn=config database that we are now using. Most of the time, this will be mainly access control settings, replication configuration, etc.
- olcDatabase={1}hdb,cn=config: This entry defines the settings for a database of the type specified (mdb in this case). These will typically define access controls, details of how the data will be stored, cached, and buffered, and the root entry and administrative details of the DIT.

The latter one numbered 1 is the database used to store all data for our BaseDN **dc=bsd-online,dc=com** and corresponds to the red-coloured box in Illustration 4.

Populating LDAP Custom Database

Once the preliminary setup of slapd(8) server configuration is complete, we can now populate our database to handle our BaseDN whose DN is dc=bsd-online,dc=org associated to olcDatabase{1}. The information we need to provide corresponds to NIS+ services, more specifically focused on password, group and hosts management although it can be expanded to support other of NIS+ tables described in nsswitch.conf(5) file.

a) Querying operational attributes for an entry

```
root@laertes:~# ldapsearch -H
ldap:// -x -s base -b
"dc=bsd-online,dc=org"
# extended LDIF
#
# LDAPv3
# base <dc=bsd-online,dc=org> with
scope baseObject
# filter: (objectclass=*)
# requesting: ALL
#

# bsd-online.org
dn: dc=bsd-online,dc=org
objectClass: top
objectClass: dcObject
objectClass: organization
o: BSD Online
dc: bsd-online

# search result
search: 2
result: 0 Success
```

```
# numResponses: 2

# numEntries: 1
```

If you need more information, add the options -LLL "+" to the ldapsearch(1) command.

Adding NIS+ tables data to the domain

Now, our OpenLDAP server is password-protected, and another LDIF file named `base.ldif` file shall be incorporated with the following contents:

```
dn: ou=People,dc=bsd-online,dc=org

ou: People

objectClass: top

objectClass: organizationalUnit



dn: ou=Group,dc=bsd-online,dc=org

ou: Group

objectClass: top

objectClass: organizationalUnit



dn: ou=Hosts,dc=bsd-online,dc=org

ou: Hosts

associatedDomain: bsd-online.com

objectClass: top

objectClass: organizationalUnit

objectClass: domainRelatedObject
```

This file must be inserted by issuing the command below with the RootDN password defined by our custom database associated to our domain dc=bsd-online,dc=org:

```
root@laertes:~# ldapadd -x -W -D
cn=admin,dc=bsd-online,dc=org -f
/etc/openldap/base.ldif

Enter LDAP Password:

adding new entry
"dc=bsd-online,dc=org"



adding new entry
"ou=People,dc=bsd-online,dc=org"



adding new entry
"ou=Group,dc=bsd-online,dc=org"



adding new entry
"ou=Hosts,dc=bsd-online,dc=org"
```

These entries whose RDN are ou=People, ou=Group and ou=Hosts correspond to the NIS+ entries in nsswitch.conf(5) file passwd, group, and hosts respectively.

Adding NIS+ users and hosts data to the domain

Eventually, we need to populate the database with our known hosts and users' account settings. There are two possible approaches:

- Migrate local /etc/{hosts,passwd,group,shadow} data to LDAP server.
- The fastest way to proceed with such a migration is to install a copy of PADL Migration Tools which are a set of Perl scripts available at http://www.padl.com . The tools require LDAP

client tools ldapadd(1) for online migration to LDAP and ldiff2dbm(1) if you prefer an offline migration to LDAP.

These scripts do not only allow migrating hosts and users accounts but also the following local files:

/etc/alias

/etc/networks

/etc/services

/etc/protocols

/etc/netgroup

/etc/rpc

Nevertheless, these scripts use the passwd/shadow approach used by GNU/Linux and SunOS 5.x systems. However, they do not support the FreeBSD password approach which uses /etc/master.password rather than /etc/shadow file to store the encrypted passwords of users.

Once these scripts have been downloaded in our FreeBSD system running OpenLDAP server, uncompress the gzipped-tarball:

```
root@laertes:~# tar xvfz
MigrationTools.tgz
```

Thereafter, replace the default path for Perl executable (for FreeBSD OS, it is placed at /usr/local/bin/perl.). The easiest way to achieve it is by the use of soft links:

```
# ln -sf /usr/local/bin/perl
/usr/bin/perl
```

Then, edit the migrate_common.ph file and set up the following variables:

```
# Default DNS domain

$DEFAULT_MAIL_DOMAIN =
"bsd-online.org";



# Default base

$DEFAULT_BASE
dc=bsd-online,dc=org

$EXTENDED_SCHEMA=1
```

Notice that the Perl variable is required to add organisationalPerson and inetOrgPerson among others to all user accounts.

There are also environment variables that may alter the normal behaviour of PADL Migration Tools. However, for simplicity; we do not reproduce it here as they are detailed in the official documentation.

Now, as a super-user, execute the following shell script:

```
root@laertes:~/MigrationTools-47#
./migrate_all_offiline.sh
```

As a result, a set of LDIF files containing the data are released:

- `base.ldif`, which contains the domain data, which has been defined.
- `group.ldif`, which contains /etc/group data in LDIF format
- `passwd.ldif`, which contains /etc/passwd data in LDIF format

- `hosts.ldif`, which contains /etc/hosts data in LDIF format

Now is the moment to feed up the OpenLDAP server database with our hosts, passwd, and other databases using these files mentioned above:

- If you are more ambitious, you can also use migrate_all_nis_offline.sh script to migrate all NIS/YP files.
- Add data step by step using `ldapadd`(1) or `ldapmodify`(1).

In such a case, we can generate individual LDIF files for each of the /etc files we plan to migrate to LDAP. The safest approach is the use of PADL Migration Tools scripts. Hence, to generate the LDIF files corresponding to hosts, passwd/shadow, and group local data files at /etc/, execute the following shell scripts:

```
root@laertes:~/MigrationTools-47#
./migrate_hosts.pl /etc/hosts >
hosts.ldif

root@laertes:~/MigrationTools-47#
./migrate_group.pl /etc/group >
group.ldif

root@laertes:~/MigrationTools-47#
./migrate_passwd.pl /etc/passwd >
passwd.ldif
```

These three LDIF files can be uploaded to our LDAP server by issuing the ldapadd(1) commands:

```
root@laertes:~# ldapadd -x -W -D
cn=admin,dc=bsd-online,dc=org -f
group.ldif

root@laertes:~# ldapadd -x -W -D
cn=admin,dc=bsd-online,dc=org -f
hosts.ldif

root@laertes:~# ldapadd -x -W -D
cn=admin,dc=bsd-online,dc=org -f
passwd.ldif

root@laertes:~# ldapadd -x -W -D
cn=admin,dc=bsd-online,dc=org -f
hosts.ldif
```

Once again, recall that we can use these scripts in another computer running Unix-like OS such as GNU/Linux to migrate its local /etc/ files to our LDAP server running on FreeBSD.

Tuning LDAP Access. Indexes Usage

Eventually, to improve LDAP performance, the indexes set up during the OLC configuration phase may be rebuilt by stopping our `slapd`(8) daemon and using the `slapindex`(8) command

```
root@laertes:~#
/usr/local/etc/rc.d/slapd stop

root@laertes:~# slapindex

root@laertes:~#
/usr/local/etc/rc.d/slapd start
```

To add new indexes, just use the following OCL file, new_indexes.ldif, with ldapmodify(1) command:

```
dn: olcDatabase={2}hdb,cn=config

changetype: modify

add: olcDbIndex
```

```
olcDbIndex: uid pres,sub,eq

-

add: olcDbIndex

olcDbIndex: displayName pres,sub,eq

-

add: olcDbIndex

olcDbIndex: default sub

-

add: olcDbIndex

olcDbIndex: uidNumber eq

-

add: olcDbIndex

olcDbIndex: gidNumber eq

-

add: olcDbIndex

olcDbIndex: dc eq
```

And execute the command:

```
[root@mcn241 ~]# ldapmodify -Y
EXTERNAL -H ldapi:/// -f
new_indexes.ldif

SASL/EXTERNAL authentication started

SASL username:
gidNumber=0+uidNumber=0,cn=peercred,
cn=external,cn=auth

SASL SSF: 0

modifying entry
"olcDatabase={2}hdb,cn=config"
```

To recreate a given index, stop slapd process and issue the `slapindex`(8) command for our database which is numbered 1 with the argument "-n 1". Otherwise, you will index both, configuration database and the custom database to allocate data from BaseDB **dc=bsd-online,dc=org**.

```
root@laertes:~#
/usr/local/etc/rc.d/slapd stop

root@laertes:~# slapindex -n 1 uid

root@laertes:~#
/usr/local/etc/rc.d/slapd start
```

Notice that if you do not specify the database number to be used by all slap-commands, the main configuration database numbered "0" will be used by default. For further details, remember the already shown `olcDatabase` attribute objects shown in previous sections.

Indexes recreation is an essential task to speed up the search operations in LDAP server databases, and shall be performed on a regular basis.

Useful administration tools for LDAP

At this point, the reader is aware that dealing with LDIF files to add, modify or delete data is a tricky task. There are a bunch of tools to provide user-friendly interfaces for LDAP administration such as PHPldapadmin to make things easier. The interface is nothing other than a web front-end built in PHP 5.6 to interact with the `slapd`(8) daemon.

To get a fully working LDAP-based front-end, download the package into our FreeBSD host running LDAP server using portmaster(8) command:

```
root@laertes:/usr/ports # portmaster
net/phpldapadmin
```

The above command installs phpldapadmin-1.2.3_7,1 and php-ldap for FreeBSD 11.1 distribution at `/usr/local/www/phpldapadmin/` directory. Moreover, it adds the following entry to `/usr/local/etc/php/ext-20-ldap.ini` configuration file to automatically load the installed extension:

```
extension=ldap.so
```

To make phpLDAPadmin available through your web site, install Apache 2.4 server and the required PHP 5.6 module:

```
root@laertes:/usr/ports # portmaster
www/apache24

root@laertes:/usr/ports # portmaster
www/mod_php56
```

The commands install www/libnghttp2 (libnghttp2-1.26.0), mod_php56-5.6.31, www/apache24 (apache24-2.4.27_1), and mod_php56-5.6.31 packages together with the required PHP module so that the server can interact with PHP files. Next, add the entry:

```
apache24_enable="YES"
```

in `/etc/rc.conf` file and start the httpd(8) server:

```
root@laertes:/usr/local/etc/rc.d #
./apache24 start
```

My suggestion for HTTP configuration is to add something like the following to httpd.conf:

```
    Alias /phpldapadmin/
"/usr/local/www/phpldapadmin/htdocs/
"

    <Directory
"/usr/local/www/phpldapadmin/htdocs"
>

        Options none

        AllowOverride none



        Order Deny,Allow

        Deny from all

        Allow from 127.0.0.1
.bsd-online.org

    </Directory>
```

in your HTTP server configuration file, and edit configuration file `config.php` at `/usr/local/www/phpldapadmin/config/ directory` with the following settings:

```
$servers->newServer('ldap_pla');

$servers->setValue('server','name','
LDAP Server');

$servers->setValue('server','host','
127.0.0.1');

$servers->setValue('server','port',3
89);

$servers->setValue('server','base',a
rray(''));

$servers->setValue('login','auth_typ
e','cookie');

$servers->setValue('login','bind_id'
,'');

$servers->setValue('login','bind_pas
s','');

$servers->setValue('server','tls',fa
lse);
```

```
$servers->setValue('login','auth_type','sasl');

$servers->setValue('sasl','mech','GSSAPI');

$servers->setValue('sasl','realm','BSD-ONLINE.ORG');

$servers->setValue('sasl','authz_id',null);

$servers->setValue('sasl','authz_id_regex','/^uid=([^,]+)(.+)/i');

$servers->setValue('sasl','authz_id_replacement','$1');

$servers->setValue('sasl','props',null);


$servers->setValue('appearance','password_hash','md5');

$servers->setValue('login','attr','dn');

$servers->setValue('login','fallback_dn',false);

$servers->setValue('login','class',null);

$servers->setValue('server','read_only',false);

$servers->setValue('appearance','show_create',true);


$servers->setValue('auto_number','enable',true);

$servers->setValue('auto_number','mechanism','search');

$servers->setValue('auto_number','search_base',null);
```

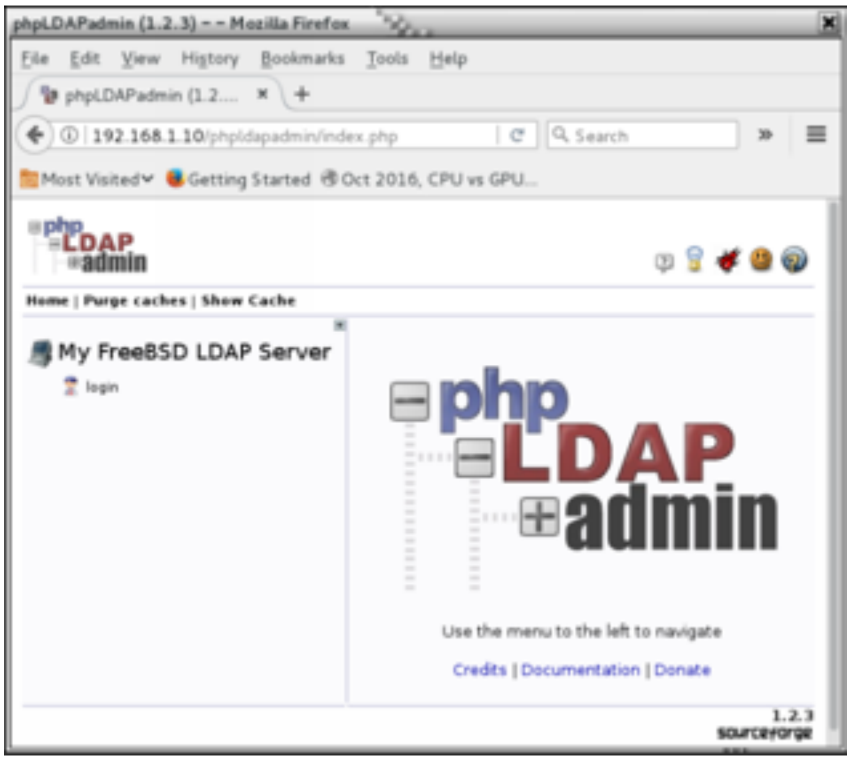
```
$servers->setValue('auto_number','min',array('uidNumber'=>1000,'gidNumber'=>500)
);

$servers->setValue('auto_number','dn',null);

$servers->setValue('auto_number','pass',null);


$servers->setValue('login','anon_bind',true);

$servers->setValue('custom','pages_prefix','custom_');

$servers->setValue('unique','attrs',array('mail','uid','uidNumber'));

$servers->setValue('unique','dn',null);

$servers->setValue('unique','pass',null);


$servers->setValue('server','visible',true);

$servers->setValue('login','timeout',30);

$servers->setValue('server','branch_rename',false);

$servers->setValue('server','custom_sys_attrs',array('passwordExpirationTime','passwordAllowChangeTime'));

$servers->setValue('server','custom_attrs',array('nsRoleDN','nsRole','nsAccountLock'));
```

```
$servers->setValue('server','force_may',array('uidNumber','gidNumber','sambaSID'
));
```

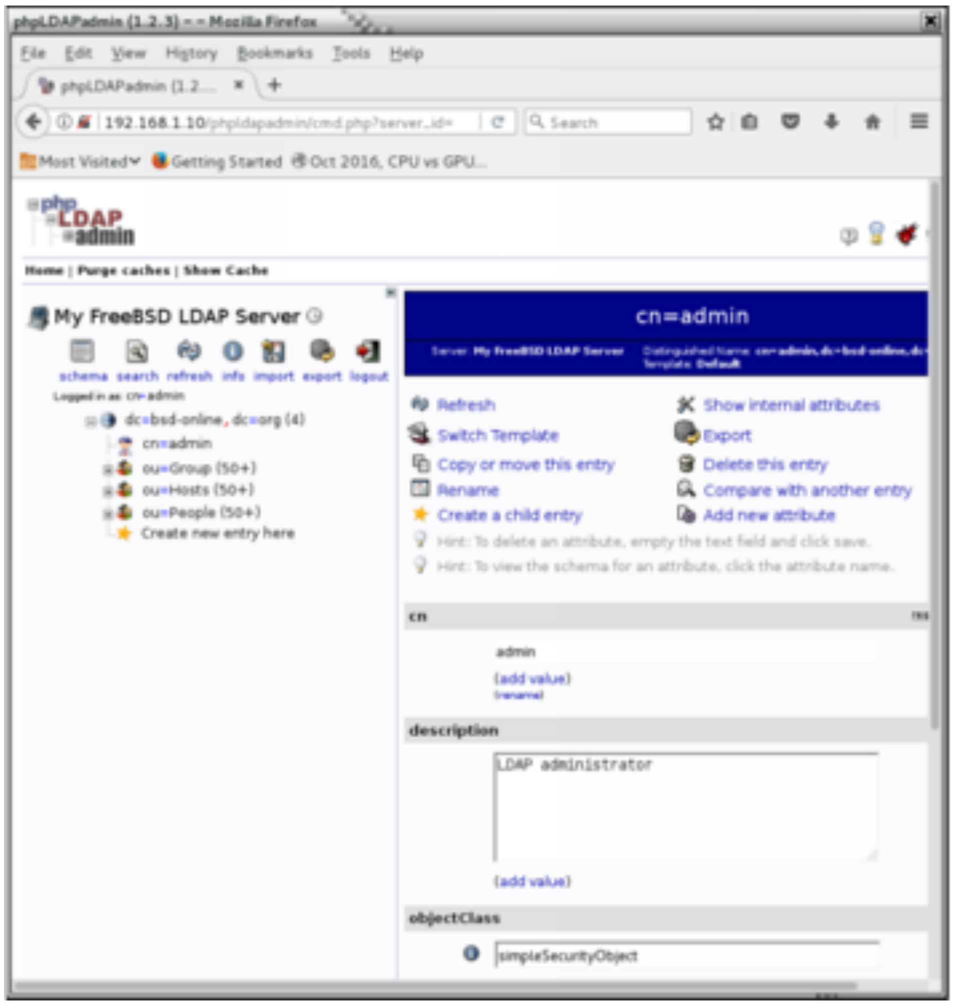
It is prudent to remark that the following setting:

```
$servers->setValue('login','attr','dn');
```

will allow you to log into HTTP/S interface using DN cn=admin,dc=bsd-online,dc=org and the BindDN password defined earlier by the slappasswd(8) command. Now, open a session with your favourite web browser and point out the URL whose IP address matches the one for our FreeBSD OpenLDAP server. Then, authenticate yourself using rootDN user data before getting the page shown in Illustration 5.


phpLDAPadmin (1.2.3) - - Mozilla Firefox
File Edit View History Bookmarks Tools Help
phpLDAPadmin (1.2....
192.168.1.10/phpldapadmin/index.php
Search
Most Visited
Getting Started
Oct 2016, CPU vs GPU...
php LDAP admin
Home | Purge caches | Show Cache
My FreeBSD LDAP Server
login
php LDAP admin
Use the menu to the left to navigate
Credits | Documentation | Donate
1.2.3
sourceforge

Also, try to navigate across the hierarchy of database associated to the BaseDN example shown in Illustration 6 which chose dc=bsd-online,dc=org . You will discover how easy it is now to add, remove or modify these values and attributes stored on it.



Hint: To delete an attribute, empty the text field and click save.

Hint: To view the schema for an attribute, click the attribute name.

cn: admin

description: LDAP administrator

objectClass: simpleSecurityObject

A Use Case. NIS+ Management using LDAP Server

Suppose we encounter a scenario in which we have an OpenLDAP server on a FreeBSD host with the data migrated using the procedure described in the previous section. This LDAP server will be used to provide centralised NIS+ data for an arbitrary farm of GNU/Linux servers running DEB-based and RPM-based distributions.

These GNU/Linux servers will play the role of LDAP clients contacting with our LDAP server using NSS PAM LDAP services according to the procedures described in the following paragraphs.

Client Configuration (DEB-based distributions)

By client, I mean the machine that connects to LDAP server to get users and authorize. It can also be the machine on which the LDAP server runs. In both cases, we have to edit three files:

`/etc/ldap.conf`, `/etc/nsswitch.conf` and `/etc/pam.d/system-auth`:

Let's start with `ldap.conf`, the ldap's client configuration file.

```
BASE    dc=bsd-online,dc=org

URI     ldap://192.168.1.10

HOST    192.168.1.10


scope sub

suffix
"dc=bsd-online,dc=org"


## when you want to change user's
password by root


rootbinddn
cn=admin,dc=bsd-online,dc=org


## there are needed when your ldap
dies


timelimit 5

bind_timelimit 5


pam_password exop


ldap_version 3

pam_filter objectclass=posixAccount

pam_login_attribute uid

pam_member_attribute memberuid


nss_base_passwd
ou=Computers,dc=bsd-online,dc=org

nss_base_passwd
ou=People,dc=bsd-online,dc=org

nss_base_shadow
ou=People,dc=bsd-online,dc=org

nss_base_group
ou=Group,dc=bsd-online,dc=org

nss_base_hosts
ou=Hosts,dc=bsd-online,dc=org
```

Now, modify the entries in nsswitch.conf so that they can appear as shown below:

```
passwd: files ldap

shadow: files ldap

group:  files ldap
```

Thereafter, ensure the following entries are present in /etc/pam.conf file:

```
auth       required     pam_env.so

auth       sufficient   pam_unix.so
likeauth nullok

auth       sufficient   pam_ldap.so
use_first_pass

auth       required     pam_deny.so


account    sufficient   pam_unix.so
account    sufficient   pam_ldap.so

account    required     pam_ldap.so



password   required
pam_cracklib.so difok=2 minlen=8
dcredit=2 ocredit=2 retry=3



password   sufficient   pam_unix.so
nullok md5 shadow use_authtok

password   sufficient   pam_ldap.so
use_first_pass

password   required     pam_deny.so



session    required
pam_limits.so

session    required     pam_unix.so

session    optional     pam_ldap.so
```

Eventually, test the LDAP configuration using the best available tool, `getent`(1):

```
$ getent passwd

$ getent hosts

$ getent group
```

Getent tool enables all data present in our FreeBSD LDAP server is available at our GNU/Linux client. Notice that you should get the result twice in case you did not remove the local data if nss_ldap is working fine. Also, PAM might be tested by deleting a user from /etc/passwd file, and trying to log in through SSH.

Client Configuration (RPM-based distributions)

For clarity, suppose a GNU/Linux computer is running CentOS 7 or RHEL 7 distribution release. First, you must install the following packages:

```
# rpm -i
openldap-clients-2.4.40-8.el7.x86_64
.rpm

# rpm -i
nscd-2.17-105.el7.x86_64.rpm

# rpm -i
nss-pam-ldapd-0.8.13-8.el7.x86_64.rp
m
```

If SElinux is enabled, it squashes client authentication connectivity and users will fail to log in. To avoid this undesired inconvenience, issue the command below in server/clients:

```
# setsebool -P allow_ypbind=0
authlogin_nsswitch_use_ldap=0
```

In such a case, there exist two options instead of the deprecated `authconfig-tui` to configure LDAP clients: `nslcd` and `sssd`, whether you want to use NSS or SSS approach, which provides NSS and PAM modules support. In case of NSS, the process of configuring an LDAP client to use centralised LDAP repository is summarised below:

```
# authconfig --enableldap
--enableldapauth –
ldapserver=192.168.1.10
--ldapbasedn="dc=bsd-online,dc=org"
--enablemkhomedir --update
```

Also, check the accuracy of the configuration by typing the following command:

```
# authconfig --test

caching is disabled

nss_files is always enabled

nss_compat is disabled

nss_db is disabled

nss_hesiod is disabled

 hesiod LHS = ""

 hesiod RHS = ""

nss_ldap is enabled

 LDAP+TLS is disabled

 LDAP server = 
"ldap://192.168.1.10/"

 LDAP base DN = 
"dc=bsd-online,dc=org"
```

Ultimately, if you want to enable LDAP+TLS support to increase security on LDAP communication, use the OpenSSL package to generate the X.509v3 self-signed certificate.:

```
# scp 
root@192.168.1.10:/etc/openldap/cert
s/cert.pem /etc/openldap/cacerts

# authconfig --enableldaptls 
--update

getsebool:  SELinux is disabled
```

Then, check if everything is working well:

```
# getent passwd c20395

c20395:x:10452:5100:ALOS ALQUEZAR;
Jose 
Bernardo:/homesun/c20395:/bin/tcsh
```

In case you prefer SSSD option, the following package should be installed before running `authconfig-tui` tool:

```
# rpm -i 
sssd-1.13.0-40.el7.x86_64.rpm
```

If you encounter any trouble, pay attention to the system logs. In both cases (SSS/NSLCD), use `authconfig-tui` command to disable the default use of TLS certificates for Secure LDAP connection.

Conclusions and Remarks

Anyway, a professional deployment of LDAP servers requires redundancy and replication mechanisms provided by `syncrepl`(8). `syncrepl`(8) is a replacement of the former `slurpd`(8) which was provided by the previous OpenLDAP releases due to its lack of reliability. In as much as replication has not been introduced here, it should be taken into account to ensure high-availability to provide centralized NIS+ services for a set of Unix-like servers. Similarly, the use of referrals in LDAP servers has not been discussed due to limited time and space which as you have noticed, were dedicated to focus on the main goal of this article.

It is important to remark that the process of building and populating LDAP server is not free from mistakes. For this reason, it is imperative to get familiar with the debugging options to identify the cause of errors that arise during the process, and indeed the official documentation for OpenLDAP available at www.openldap.org.

References and Bibliography

http://www.nlc-bnc.ca/publications/1/p1-244-e.html	Directories and X.500: An Introduction
Timothy A. Howes, Gordon S. Good, Mark Smith; Macmillan Publishing, USA	Understanding and Deploying LDAP Directory Services
https://tools.ietf.org/search/rfc1558	RFC1558. A String Representation of LDAP Search FIlters
https://tools.ietf.org/search/rfc2222	RFC 2222. Simple Authentication and Security Layer (SASL)
https://tools.ietf.org/search/rfc6101	RFC6101. The Secure Sockets Layer (SSL) Protocol Version 3.0
http://www.openldap.org/	OpenLDAP Home Page
http://www.penldap.org/docs/admin24/	OpenLDAP 2.4 Administration's Guide
http://phpldapadmin.sourceforge.net	PhpLDAPadmin Home Page
https://www.freebsd.org/releases/11.1R/announce.html	FreeBSD 11.1 Release Home Page

Acronyms and Abbreviations

DSA	Directory Specific Agent
DIT	Directory Information Tree
DSE	DSA Specific Entry
OCL	OpenLDAP Online Configuration
DN	Distinguished Name
RDN	Relative Distinguished Name
LDIF	LDAP Data Interchange Format
LDAP	Lightweight Directory Access Protocol
NSS	Name Service Switch
PAM	Pluggable Authentication Modules
SSL	Secure Sockets Layer
TLS	Transport Layer Security
SASL	Simple Authentication and Security Layer
OID	Object Identifier
MDB	Memory-Mapped Database
LMDB	Lighting Memory-Mapped Databases
YP	Yellow Pages

Meet the Author

José B. Alós has developed an important part of his professional career since 1999 as an EDS employee, as a UNIX System Administrator, mainly focused on SunOS/Solaris, BSD and GNU/Linux and High-Availability solutions for industry, communications services and banking. In 2007 he joined EADS Defense and Security, as the person responsible for providing support for end-users in aircraft engineering departments for long-term projects. These days his professional career has moved to High-Performance Computing and Simulation area within Airbus group, He was also Assistant Professor in the Universidad de Zaragoza (Spain), and his academic background includes a PhD in Nuclear Engineering and three MsC in Electrical and Mechanical Engineering, Theoretical Physics and Applied Mathematics.

Bitcoin Full Node on FreeBSD

What is a Bitcoin?
What is a Bitcoin Wallet?
What is a Blockchain?
What is Mining?
What is Pooled Mining?
What is a Full Node?
What is a Bitcoind?
How To Start Bitcoind To Be Full Node?

What is a Bitcoin ?

Bitcoin is a valuable popular open-source cryptocurrency that was invented by Satoshi Nakamoto in 2009. Bitcoins have value because they possess same characteristics like money (durability, portability, fungibility, scarcity, divisibility, and recognizability), but based on the properties of mathematics rather than on physical properties (like gold and silver) or trust in central authorities (like fiat currencies). In short, Bitcoin is backed by mathematics.

Bitcoin is the first decentralized peer-to-peer cryptocurrency that is controlled by its users.

Transactions take place directly between users, and are later verified by network nodes with digital signature and then placed in a public distributed ledger called a blockchain. Bitcoin is unique in that only 21 million bitcoins will ever be created. The unit of the bitcoin system is *bitcoin or mBTC.*

What is a Bitcoin Wallet ?

A *wallet* is nothing more than a pair of public and private keys that are created by a client to store the digital credentials for your bitcoin.

There are several types of wallets:

- Desktop Wallet
- Token Wallet
- Online Wallet
- Mobile Wallet

A token wallet is the safest way to work with bitcoin network, but you can use your mobile or pc as a bitcoin wallet.

What is a Blockchain?

A *blockchain* is a ledger that records bitcoin transactions. The blockchain is a distributed database that achieves independent verification of the chain of ownership. Each network node stores its own copy of the blockchain. Transactions will broadcast on the bitcoin network, and about 2400 transactions create a block. These blocks are building blocks of the blockchain.

What is Mining?

Mining is the process of dedicating computing power to process transactions, secure the network, and keep everyone in the system synchronized together. It has been designed to be fully decentralized.

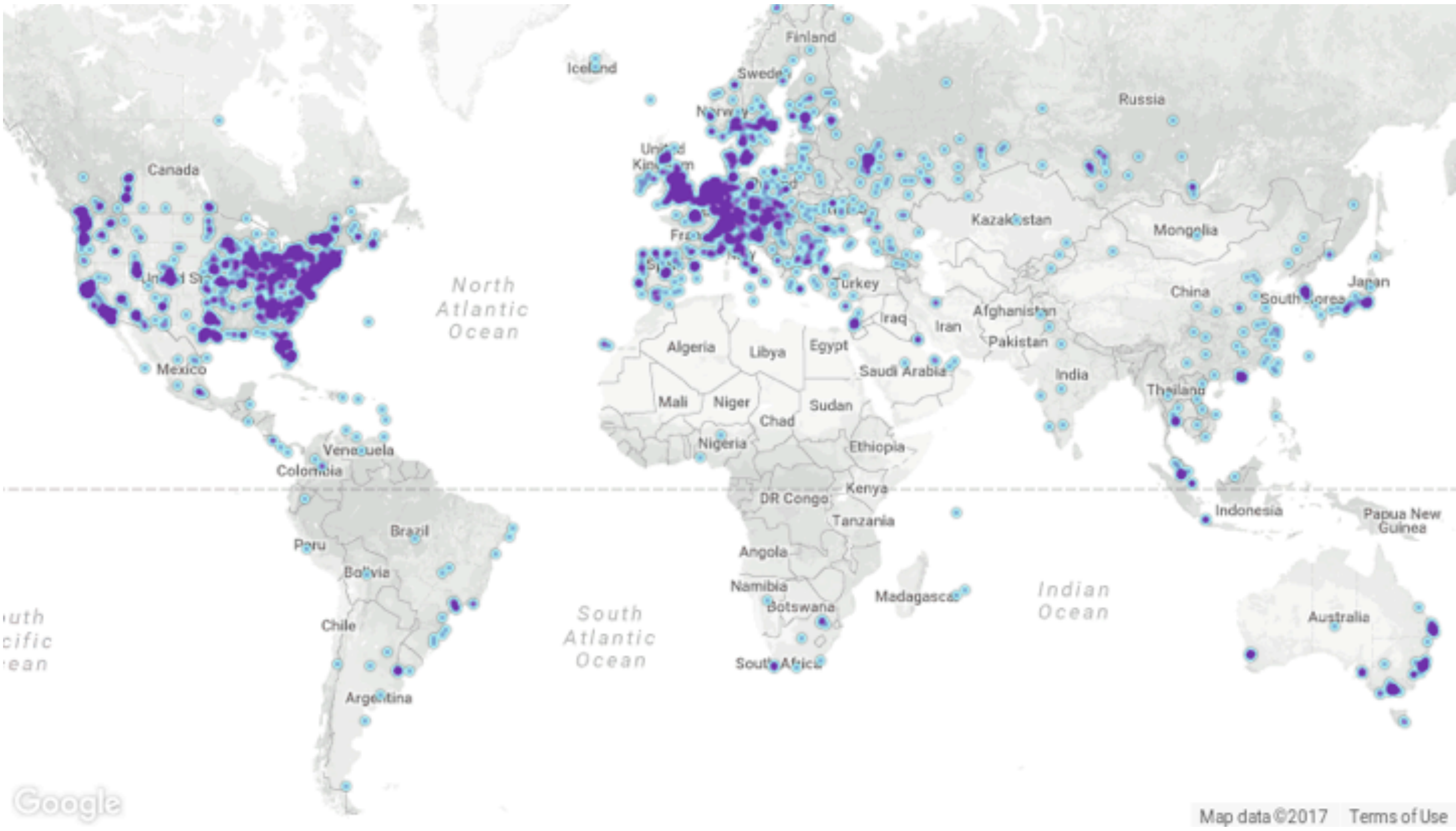
Miners need mining software with specialized hardware. Mining software listens for transactions broadcasted through the peer-to-peer network and performs appropriate tasks to process and confirm these transactions. Bitcoin miners perform this work because they can earn transaction fees paid by users for faster transaction processing.

New transactions have to be confirmed then be included in a block along with a mathematical proof of work. Such proofs are very hard to generate because there is no way to create them other than by trying billions of calculations per second. Hence, miners are required to perform these calculations before their blocks are accepted by the network and before they are rewarded. As more people start to mine, the difficulty of finding valid blocks is automatically increased by the network to ensure that the average time to find a block remains equal to 10 minutes. As a result, mining is a very competitive business where no individual miner can control what is included in the blockchain.

The proof of work is also designed to depend on the previous block to force a chronological order in the blockchain. This makes it exponentially difficult to reverse previous transactions because it would require the recalculation of the proofs of work of all the subsequent blocks. When two blocks are found at the same time, miners work on the first block they receive and switch to the longest chain of blocks as soon as the next block is found. This allows mining to secure and maintain a global consensus based on processing power.

What is Pooled Mining?

You have more chances if you participate with others to create a block. In a pool, all participating miners get paid every time a participating server solves a block. The payment



depends on the amount of work an individual miner contributed to help find that block.

What is a Full Node?

A full node is a client that fully validates transactions and blocks. Full nodes also help the network by accepting transactions and blocks from other full nodes, validating those transactions and blocks, and then relaying them to further full nodes.

Many people and organizations volunteer to run full nodes using spare computing and bandwidth resources.

What is a Bitcoind?

bitcoind is a Bitcoin client under the MIT license in 32-bit and 64-bit versions for Windows, GNU/Linux-based OSes, Mac OS X, OpenBSD and FreeBSD as well.

How To Start Bitcoind To Be Full Node?

Install bitcoind by PKG:

```
#pkg install bitcoin-daemon
```

Install bitcoind by source:

```
#fetch https://github.com/UASF/bitcoin/archive/v0.14.2-uasfsegwit1.0.tar.gz

#tar xzvf v0.14.2-uasfsegwit1.0.tar.gz

#cd bitcoin-0.14.2-uasfsegwit1.0
```

Install dependencies:

```
#pkg install autoconf automake libtool pkgconf boost-libs openssl libevent gmake
```

Then config build and install Bitcoind:

```
#./autogen.sh
```

```
#./configure –without-gui
```

Since we are in command line, GUI is not required and –without-gui will disable it.

```
#gmake
```

```
#gmake install
```

Start Bitcoind client and wait full-sync with other nodes:

```
#bitcoind -daemon
```

bitcoind will download database that is about 150GB. You can check your node status by clicking on this URL:

https://bitnodes.earn.com

Conclusion

Cryptocurrencies are replacement for banking we know today, and bitcoin is the game changer. Mining bitcoin with typical hardware is not a good idea. It needs specialized devices like ASIC, but you can create a full node and help the bitcoin network.

Useful Links

https://en.wikipedia.org/wiki/Cryptocurrency

https://bitcoin.org/en/faq

Meet the Author

Abdorrahman Homaei has been working as a software developer since 2000. He has used FreeBSD for more than ten years. He was involved with the meetBSD dot ir and performed serious trainings 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**

Blog Presentation

Interview with

Carlos Klop

SOLRAC

Please tell us about yourself?
I am Carlos Klop. I live in Netherlands and have been working in IT for 14 years now. When it's windy outside, I am either out kitesurfing or behind my computer.

How you first got involved with programming? What was your path?
In 2000, I started to create a forum for a group of friends with PHP 3 and MySQL server. From there, I became interested and figured out myself how to program. Later in 2005, I learnt how to program in Java at school. From there, I did some small C++ and Object-C projects. The last 6 years, I worked for companies that had Microsoft C#/.NET developers, and that was when I primarily got involved in programming.

Reading your blog, we can see that you have a wide field of expertise. Please tell us which is your favorite area?
Automating workflows is what I like the most. At this moment, I pick a programming language that fits the job and that I have experience with. In the feature, maybe I did focus on specific areas, but for now, I like the diversity. But if I have to pick a favorite one for the last year, about it would be Microsoft Azure. I started to work with Microsoft Azure from the beginning and hence reserve a lot of questions about it. From last year, I think it has been my favorite since it has enabled me to create ARM templates, make C# scripts to work with data in blob storage, and assist customers migrate from on-premise Windows environments to Microsoft Azure platforms.

What is the most interesting programming issue you've encountered, and why was it so amazing?
One of the web shops we hosted was crashing all the time on the eight front-end server we had running. It was a C# / .NET Framework application and it was running on Windows Server. We discovered that it used more than 15GB of memory before it crashed. The memory usage was astonishingly more than 4GB we had expected during normal operations. The problem was that the high season for this web shop had commenced, and we had one week to fix this. None of the developers could tell us where the problem was. So, we started the investigation by cutting the Web Application in pieces and running them in separated servers with a reverse proxy in front. We could separate the web application with the URL used. Hence, I created a few rewrite rules for it. Splitting the application was costing us already a few days since it was one big monolithic application. After identifying the crashing pieces, we started debugging and we found out that the XML converter tool was causing high memory usage. Because the XML converter was not used all the time, we had troubles to find exactly where the issue emanates. Also, that I was not the developer of this code and only was there to troubleshoot it, it

was interesting to find the solution in a short amount of time.

What tools do you use most often, and why?
Mostly, I use the internet browsers because new application that I develop are mainly with a web interface. For programming, I rely on NetBeans since it is easy and fast to use. Also, I like to use TMUX in the terminal to split screens.

What was the most difficult and challenging implementation you've done so far? Could you give us some details?
We had to create a search engine based on availability for accommodation, flight, and car rental as a package for a travel company. Each data set was separated from different databases or connected with Web API's. Loading the data in Elasticsearch indices and partitioning them for optimal search speed was one task. Together with Elasticsearch engineers, we tried to figure out how to query this data at an acceptable response time. During the Proof of Concept, things were more challenging than we thought in the beginning. Because of the combination of three items, there were so many combinations with different outcomes. Hence, it was difficult to make the data flat so that the search engine can optimize it.

Can you tell us about your favourite features in the new releases of your favourite OS?
For hardware, I always pick Apple with Mac OS. I like the stability of Unix/Linux compared to Windows. My favorite feature is probably PF, but it's already been there for a long time.

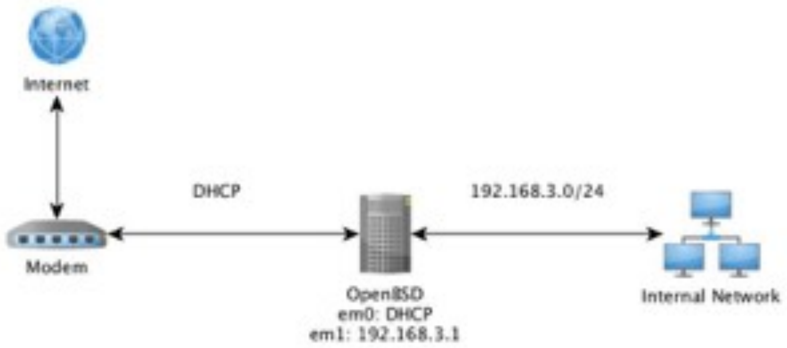
Do you have any specific goals for the rest of this year?
Technically, I am learning from .Net Core and started with OpenCV. My goal is to learn more from both and see how I can use them.

What's the best advice you can give to the BSD magazine readers?
Keep track of what you do and how you figured out by writing down. My blog is also a note to myself, so that I can go back, read and gauge my progress.

Thank you

OPENBSD ROUTER WITH PF

OpenBSD is an operating system which has been used widely for network routing and firewall. Also, it can easily install for you Virtual Machine lab environment. In this blog post, you will learn how to turn an OpenBSD installation quick in router and NAT with PF for your environment.



This blog post will focus on the basic steps and settings.

Install OpenBSD

Make sure you create an new VM with at least two network interfaces. One connected to the internet and the other one to the internal.

For my test machine, I used OpenBSD version 6.0 that is available on the OpenBSD website (http://www.openbsd.org/), or directly via the following link from Amsterdam: ftp://mirror.meerval.net/pub/OpenBSD/6.0/amd64/install60.iso

The installation can be done with default settings. My preference was not to install the X Windows system.

Configure network

First, we will start with the configuration of the network. Run the following command to show the available interfaces:

```
ifconfig -a

lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu

32768

        index 4 priority 0 llprio 3

        groups: lo

        inet6 ::1 prefixlen 128

        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x4

        inet 127.0.0.1 netmask 0xff000000

em0:

flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST>

mtu 1500

        lladdr 00:00:00:00:00:00

        index 1 priority 0 llprio 3

        groups: egress

        media: Ethernet autoselect (1000baseT

full-duplex,master)

        status: active

        inet 192.168.1.40 netmask 0xffffff00

broadcast 192.168.1.255

em1:

flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST>

mtu 1500

        lladdr 00:00:00:00:00:00

        index 2 priority 0 llprio 3

        media: Ethernet autoselect (1000baseT

full-duplex,master)

        status: active
```

Now that we have the interface names, em0 and em1, we can configure them individually by creating (or editing if they exist) the files hostname.em0 and hostname.em1. In OpenBSD, you can use vi or bash. In this example, I will use a bash command to overwrite the files:

```
echo "dhcp" > /etc/hostname.em0

echo "inet 192.168.3.1 255.255.255.0 192.168.3.255"
> /etc/hostname.em1
```

We set interface em0 to DHCP, assuming this is our external interface that is coupled to an DHCP scope. And we have to set em1 interface to 192.168.3.1 that will be our internal network.

Enable IP forwarding

This command is the same for most Linux distributions. We will enable the IP forwarder in the kernel, so we are allowed to receive and forward network packages that are not for this host. We can check if it's enabled with the following command:

```
sysctl | grep forward
net.inet.ip.forwarding=0
net.inet.ip.mforwarding=0
net.inet6.ip6.forwarding=0
net.inet6.ip6.mforwarding=0
```

To enable this, we set the following in the sysctl.conf file:

```
echo net.inet.ip.forwarding=1 >> /etc/sysctl.conf
echo net.inet6.ip6.forwarding=1 >> /etc/sysctl.conf
```

Rebooting the OpenBSD installation enables IP forwarding configuration for IPv4 and IPv6.

With the IP forwarding enabled, we can use the host as an IP router, configure the internal connected server to use this server as the gateway, and it will work. But wait, not everything will work. If you send a PING command, it will arrive at the destination. However, can it trace its way back? Not if it's an internet router, it only knows the way to our router and not the hosts.

Configure NAT with PF

You need to configure Network Address Translation so that the session is saved on our host, and the packet is forwarded with a new IP number. To turn our OpenBSD installation in a NAT router, we will use the integrated PF (Packet Filter) configuration. Open the configuration file /etc/pf.conf with a text editor;

```
vi /etc/pf.con
```

To start with a working example, make sure you empty the file and add the following configuration:

```
# Create blocks that are variable

ext_if="em0"

int_if="em1"

icmp_types="echoreq"


# Skip all loopback traffic

set skip on lo


# Perform NAT on external interface

match out on $ext_if from $int_if:network to any
nat-to $ext_if


# Define default behavior

block in

pass out keep state


# Allow inbound traffic on internal interface

pass quick on $int_if


# Protect against spoofing

antispoof quick for { lo $int_if }


# Allow other traffic

pass in on $ext_if proto tcp to ($ext_if) port ssh
flags S/SA keep state
```

In the example on configuration, we created an variable for em0 = ext_if and em1 =int_if. If we need to change the interface later, it's easier only to change those variables. Further, in the configuration, you will find the NAT rule set so that any internal network traffic will go through the NAT. If you have a host connected to the internal and sends traffic out, you can see this in the PF state table;

```
pfctl -s state
```

Session states will now be held on the OpenBSD host we have just configured, and NAT is done.

Inbound NAT

But wait, we can send traffic from our internal network to external. However,what if we want to forward a network port to our internal network? For example, we have web servers running behind our NAT router. We can enable TCP port 80 by adding the following configuration to /etc/pf.conf :

```
pass in on $ext_if proto tcp from any to any port
80 rdr-to 192.168.1.5
```

The configuration shown above will pass traffic through TCP port 80 from our external interface to the internal host with IP 192.168.1.5. If we change the *any* to an specific IP number, we can limit the source addresses that are allowed to visit our web server. For example:

```
pass in on $ext_if proto tcp from 86.82.0.0/16 to
any port 80 rdr-to 192.168.1.5
```

In the last example, we only allowed IP numbers from KPN ADSL in the Netherlands to our web server. This configuration can be useful, for instance, in test environments where you can specify an smaller IP range from your home IP number .

More about Packet Filter can be found here: http://www.openbsd.org/faq/pf/filter.html

Blog Presentation

Interview with

Eduardo Lavaque

Can you tell our readers about yourself and your role nowadays?
I'm a 20-year-old-guy that started being interested in programming and computers when I was 12, I think. Born in Argentina, raised in Mexico, and my great grandma is Italian. Although my culture is Mexican, I live in Switzerland and have three passports.. I was homeschooled so I had a lot of time to look into Windows, OS X/macOS, the Linux distros, and then embark on a short adventure with the BSDs. Therefore, by consequence, I am quite familiar with the innards of Linux-running machines and so on.

Nowadays, I am a professional JavaScript developer. I write for both the front end and for the backend with Node.js. I'm sure at some point I would work with something else in the backend, probably Elixir.

How you first got involved with programming?
I was a kid, and my dad was editing the HTML of websites. I liked how the colors looked from the syntax highlighting in the text editor. I always had some sort of affinity for code.

My first actual programming was with PHP when I was 12 or so. My dad taught me my first hello world program.

While having a wide field of expertise, please tell our readers on which area you put the much emphasis, and why?
Yes, web development is wide, and it keeps getting wider, which is exciting if you're into new stuff. Personally, I put my emphasis on the back end. The front end is exciting and all, and there are new libs and approaches all the time, but it's exhausting. Pixel perfection is exhausting. Limitations and workarounds are exhausting.

Back end is always interesting. Every product has a different solution. With the freedom, you can do all kinds of trash as long as it works as it should. No one is the wiser, but that doesn't mean what I do is hogwash.

My emphasis is to try to follow Robert C. Martin's Programmer's Oath. It's almost impossible, but an excellent thing to strive towards. http://blog.cleancoder.com/uncle-bob/2015/11/18/TheProgrammersOath.html

What was your best work? Can you tell what was the idea behind it? What was its purpose?
I'm always bad at answering this question because I never get anywhere with my side projects. Something I have to fix it myself. So I will deviate a bit from your question.

My best work is the one which am currently doing as my full-time job. I'm proud of what I do since it works well. Of course, the purpose is whatever the company is trying to solve.

Currently, I am interested in making a Moneywiz alternative. I use it, but I'd love to have an open-source version. I've been brainstorming that, and coding on it will start soon.

What tools do you use most often, and why?
Neovim, tmux, bash and mksh shells, Firefox, Chrom(e|ium), Atom, Sublime Text, GitHub, Git, ag (the_silver_searcher), off the top of my head.

All the standard stuff. I use all of them. I am not shy of using whatever tool that fits the task best. Right now, I'm feeling the vim nostalgia, so my main editor is vim. But for merge conflicts, I use Atom (this conflict resolver is awesome).

As to why I use them is because I like them. They fit the task best. If I had to use an IDE, I'd also do it. However,t I'd first try to find a way to do it with the command line.

What was the most difficult and challenging implementation you've done so far? Could you give us some details?
Besides that one time when I had to do a crazy SQL migration, it would be the webhooks system which we have implemented in the product I'm working on. I was doing this for the first time, and we're using RabbitMQ, also something which I had never interacted with earlier. The hard part, though, was the fact that our webhooks tell you differences from the new data set and the old one. Therefore, figuring out differences when implementing these difference detectors had me stumped longer than I would've liked.

Do you have any specific goals for the rest of this year?
Not much left of this year, so not really. Next year looks interesting though. One of my goals is to become stable in Switzerland and in my new job I am currently in. December is my fifth month in Switzerland.

If I can finish that financial app, there would be a big cherry on top. I'm very slow paced with my goals.

What's the best advice you can give to the BSD magazine readers?
Oh, hi BSD users!

Well, I am probably not qualified to give you life advice, but I would like to say take it slow. For professional advice, Elixir and Clojure look like good investments for your profession, if you're into web dev. However, neither is specifically designed for web.

In general, the best advice I can give is. Find out about it with your own eyes and form your own opinions and conclusions. Don't get on the bandwagon just because everyone is. Get on the bandwagon only when you, with your own eyes, see that it's a good idea to do so.

I hope you have enjoyed this interview.

Thank you

My Switch to OpenBSD,

First Impressions

This blog post is about my first impressions when I switched to OpenBSD. Probably, it won't be my last blog post on the subject. So that you can understand how I use my distros, "ricer" is a term used mostly to refer to people that change the look of their setup to make it look very attractive, outside of the defaults of whatever environment they have. Take a look at /r/unixporn for many good examples of ricing.

An under-the-bonnet ricer means the ricer only looks to improve the workflow or the commands and stuff available to them, not the looks. I am an under-the-bonnet ricer to the core. Because of my nature, I've had to reinstall Arch 3 times because I broke it and have been using CRUX for a while since it's a fun distro to play with. OK, now on to BSD.

Why?

Why OpenBSD? Why not FreeBSD,NetBSD,DragonflyBSD or any other BSD? Why BSD in the first place?

I've been a Linux user for several years.Recently, I've been getting into being all POSIX-compliant and stuff, but GNU's coreutils have been grinding on my nerves with that stuff. Though Linux is awesome and compiling it is fun, I didn't like the OS on top of it.. So I wanted to switch to something better, that something was BSD.

Sidenote: Why does the GNU `sort` command have an `-R` flag which randomises the result? You can't sort something into being random. That's an oxymoron (with a particular choice of definitions).

Now, why OpenBSD instead of another BSD? First of all, my friends at Nixers.net prefer OpenBSD (those that use a BSD). It's good to switch to a system which is known to several people you can freely interact with.. It makes the switch much more fun.

Secondly, I did try to switch to FreeBSD in December. It was a chance I had to switch. But I had trouble getting X to work, and at that point, I needed a working OS. This time I didn't want to deal with the X stuff. Therefore, I just went ahead and installed OpenBSD which I had heard had excellent X support out of the box. To my surprise, it did..

And thirdly because of the security orientation of the whole project. That, for me, is a really attractive feature.

First impressions

Short version: I'm loving it.

Keep reading for the long version.

The install

Getting the USB stick ready was unique. I downloaded the `install56.iso` , but that didn't work when I `dd`'d it into the USB stick. Thereafter, I read the `INSTALL.amd64` file. It uses the `.fs` file for the USB stick and not the `.iso` file. Hence, I downloaded that, `dd`'d it, and it worked. So that was new.

The install was certainly "weird" for me since I was more into manual Linux distros where I could format the hard drives, mount the partitions, write the `fstab`, etc. all manually. It was pleasant though. Somehow, I don't feel dirty with a clean install of OpenBSD as I do with a clean install of any Linux. Probably, that could be as a result of the lack of GNU. But yeah, I was expecting a slightly more graphical install since I had experienced the FreeBSD install. Anyway, I'm fine with text prompts. It's still simple.

X and hardware support

The X support is extremely incredible and simple. To start with, I enabled xdm. However, I quickly disabled it because I had my `.xinitrc` file.

Simply put, if I don't mention it, it's because it worked perfectly.

The only thing that isn't supported is my wifi card, a dreaded BCM4315. That would have been a deal breaker some months ago, but now I have an extra long ethernet cable, so it's fine. Since this is a laptop, I need to buy that wifi dongle.

I encountered a bit of trouble with the lid. Closing it suspended all programs , which was fine, but when I opened it, the screen was all black. I pressed buttons and stuff and it didn't turn on again.Hence I'm guessing my monitor doesn't wake up for some reason. I don't know what's up with that.

Sidenote: I've a Dell Vostro 1500 from 2007, with an Intel Core 2 Duo 2.0GHz.

Ports/packages system

The ports/packages system is something I like in OpenBSD. It is rather sad that CVS is still preferred over Git for the ports. However, that isn't going to stop me from liking it. Seriously, why CVS not Git?

I love how it's decentralised. A ton of mirrors counts as decentralised for me.

I like how the `$PKG_PATH` variable works. I'd be fine with this setup if it was in some Linux distro.

The `pkg_add` command also works very well as. It lets me know the all the stuff that it installs, and when it installs dependencies, it lets me know what package requires that dependency. As a matter of fact, you can easily identify what piece of software is installing a ton of dependencies you don't want.

Being productive again

First of all, I would like to thank the BSD Now and their tutorials, especially this one: http://www.bsdnow.tv/tutorials/the-desktop-obsd

As a Node.js dev, it can be a little hard to get started if you're used to using `nvm` because of a little bug which I already reported. Anyway, worry not since by the time you read this, it'll have been fixed most probably.

Though Node v0.10 is included in the packages, it is not the latest and greatest of the packages. I am glad I can still work with it, but I expect that to get updated to v0.11 or v0.12 in OpenBSD v5.7.

Other than that, everything has been very smooth so far. Some software I like isn't in the packages. Nevertheless, I can compile that myself so there's no issue.Also, I'm not too worried since I expect that it will be included in the next release.

On the first day, I installed OpenBSD in the evening. The next day, I spent some time figuring out how stuff worked, basics here and there etc. and getting up to a working productive state again. On the third day, I was completely productive again. Therefore, I only experienced downtime while I was installing it and figuring out basics.

Honestly, I'd say that it will take less than half an hour from the moment you plug in the USB to when you get back to work again.

Final verdict

If you're considering switching to OpenBSD, totally go for it. There is nothing stopping you but yourself.

Also, I spent an hour trying to figure out why the wifi didn't work because I assumed it would have worked like in Linux.Therefore, make sure your hardware is supported.

Column

The year of 2017 went by so quickly, and we are now entering the season of goodwill, parties, and family gatherings. It is a time to look back, look forward, and take stock. What might 2018 bring to the technology table?

by ***Rob Somerville***

2017 has been an interesting year for me. Since being made redundant earlier this year, I have spent a considerable amount of time reading, researching, and writing. This has meant that I have been "plugged in" to the internet probably more times than even my teenage daughter, which is truly saying something. Some things don't change though. My trusty Linux box (probably getting on for ten years old now) still runs flawlessly, rarely disappoints, and despite having less than 10% disk free (Santa, will you please put another 1TB external SATA drive in my stocking, thank you), carries on regardless. My cheap Lenovo tablet has suffered premature death due to the dog knocking it out my hand. The device landed on the micro USB plug, fracturing the motherboard so that it will not charge. The battery of my daughter's Samsung tablet would not charge. After forking out £30 for a replacement battery and the kit to disassemble the unit, I discovered the tablet had suffered a similar internal fate. After six months of "repossessing" my wife's Lenovo tablet (a larger version of my own which was rarely used), it began to crash randomly despite a few rebuilds. All I do is read my emails on it, watch a few YouTube videos, and go to a few news sites. All the various Linux and BSD boxes scattered around the house just carry on despite abuse by coffee, roll up tobacco, cigarette ash, dust, dog, and cat hair. I give them a clean every so often, the occasional software upgrade as required, and it is business as usual. Even my flashy Microsoft illuminated keyboard survived getting a glass of wine poured over it. Some things never change.

What has clearly happened though in 2017 is that the commercial mood towards the implications of technology is becoming much less favourable. The larger internet players and mainstream media organisations are apparently going on a censorship rampage, with the cries of "Fake news" and "Fact checking" being the latest buzzwords thrown around. My local newspaper (which will remain nameless to save them from embarrassment) has their website adorned with "Say no to fake news" logos, and even goes as far as to solicit financial donations from readers. Their resolve would be fine, except that I don't know a single person in the area that trusts a word they say, as it is neither objective nor does it attempt to redress this imbalance by performing some credible investigative journalism. It has got to the point that a number of friends just don't buy the paper on principle anymore, not only is it so anodyne (while maintaining clear political bias, a feat truly to behold), but also the website is crammed full of adverts and click-bait that would insult the intelligence of a 5-year old. It is the pinnacle of a journal that is driven by its advertisers for its advertisers, and the readers are getting somewhat disillusioned since the demographic where I live leans towards the more mature edge of the spectrum.

This same mantra “Trust us because we are big and established” is permeating all the social media platforms now as well. Numerous commentators, especially those with controversial views, are having their streams demonetized or rated as adult content. As a result, there will be an inevitable mass migration to other providers from the biggest players in 2018 if the trend continues. That is if the giant global AI bot doesn’t get there first. The biggest upset in 2017 is the realisation that technology, especially AI and Robotics, will decimate traditional safe employment sectors such as driving, journalism, the law, and indeed prostitution. My guess is the biggest upset in 2018 will be a major clampdown on crypto-currencies, as the banks and law enforcement are already very concerned about the business and criminal implications. I heard someone the other night (at a party of all places) say to a friend “Don’t buy BitCoin unless you want to support money laundering”. Hmm, if that is the case, I am not sure how I am going to survive for long without cash unless I can arrange some form of barter system with the supermarket, the taxman, and the local garage. I’ve forgotten now exactly how much of the traditional economy is funded by dirty money, but it is a sufficient amount that if it was taken out of circulation, the economy would collapse overnight. Maybe my friend has a point. If the criminals do migrate to crypto-currencies, it could trash the current financial system. Hence my prediction is that there will be a clampdown, maybe some form of licensing, taxation or a foolish attempt or two to outright ban it. It is definitely on the list of feats of the “unintended consequences of technological advance” that some are deeply concerned about.

As an ageing dinosaur, I eschew Facebook as much as humanly possible. However, it is sadly becoming more and more essential to have a Facebook account. Recently, I was performing some research and the only way I could access the information was via that specific platform. More and more employers are advertising their job vacancies on Facebook, and as a consequence, their HR departments regularly want to see a valid profile and a sensible completed timeline as part of their “hidden” recruitment process. Thanks, but no thanks. If I really wanted to share with you my likes and dislikes, I’m sure convention would have dictated that these matters would be added to my CV as a matter of course.

The cyber-attacks will carry on, and the level of sophistication will increase, as will the collective stupidity of a few organisations that really get caught out. Will 2018 be the first year when fatalities can be directly attributed to hackers? We came very close in the UK this year when our National Health Service was the victim of crypto-malware.

As for you, your family, friends, colleagues and associates, I wish you a peaceful, prosperous and healthy Christmas and New Year. What is clear is that 2018 will hold many more surprises than 2017, and as always, IT will be on the front line.

HEY GOLIATH...

MEET DAVID

TRUENAS® PROVIDES MORE PERFORMANCE, FEATURES, AND CAPACITY PER-DOLLAR THAN ANY ENTERPRISE STORAGE ARRAY ON THE MARKET.

Introducing the TrueNAS X-Series: Perfectly suited for core-edge configurations and enterprise workloads such as backups, replication, and file sharing.

- ★ **Unified:** Simultaneous SAN, NAS, and object protocols to support multiple applications
- ★ **Scalable:** Up to 120 TB in 2U and 720 TB in 6U
- ★ **Fast:** Leverages flash and the Intel® Xeon® CPU with AES-NI for blazing performance
- ★ **Safe:** High Availability ensures business continuity and avoids downtime
- ★ **Reliable:** Uses OpenZFS to keep data safe
- ★ **Trusted:** TrueNAS is the Enterprise version of FreeNAS®, the world's #1 Open Source SDS
- ★ **Enterprise:** Enterprise-class storage including unlimited instant snapshots and advanced storage optimization at a lower cost than equivalent solutions from Dell EMC, NetApp, and others

The TrueNAS X10 and TrueNAS X20 represent a new class of enterprise storage. Get the full details at iXsystems.com/TrueNAS.

iX systems™

intel XEON inside™

Copyright © 2017 iXsystems. TrueNAS and FreeNAS are registered trademarks of iXsystems, Inc. All rights reserved. Intel, the Intel logo, Xeon, and Xeon Inside are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

ServerU

Rack-mount networking server

Designed for BSD and Linux Systems

DESIGNEDFOR FreeBSD | DESIGNEDFOR GNU / Linux | DESIGNEDFOR PRO apps FreeBSD Enterprise Appliance | DESIGNEDFOR pfSense

Designed. Certified. Supported

Up to **5.5Gbit/s** routing power!

KEY FEATURES

- 6 NICs w/ Intel igb(4) driver w/ bypass
- Hand-picked server chipsets
- Netmap Ready (FreeBSD & pfSense)
- Up to 14 Gigabit expansion ports
- Up to 4x10GbE SFP+ expansion

PERFECT FOR

- BGP & OSPF routing
- Firewall & UTM Security Appliances
- Intrusion Detection & WAF
- CDN & Web Cache / Proxy
- E-mail Server & SMTP Filtering

contactus@serveru.us | www.serveru.us
8001 NW 64th St. Miami, LF 33166 | +1 (305) 421-9956