



BSD News

August 2019



REPORTS & UPDATES



LibreSSL 3.0.0 Released in OpenBSD

List: [openbsd-announce](#)
Subject: [LibreSSL 3.0.0 Released](#)
From: [Brent Cook <busterb \(\) gmail ! com>](#)
Date: [2019-08-05 12:43:56](#)
Message-ID: [20190805124356.GA38208 \(\) santo ! lan](#)
[Download RAW [message](#) or [body](#)]

We have released LibreSSL 3.0.0, which will be arriving in the LibreSSL directory of your local OpenBSD mirror soon.

This is the first development release from the 3.0.x series, which will eventually be part of OpenBSD 6.6. It includes the following changes:

- * Completed the port of RSA_METHOD accessors from the OpenSSL 1.1 API.
- * Documented undescribed options and removed unfunctional options description in openssl(1) manual.
- * A plethora of small fixes due to regular oss-fuzz testing.
- * Various side channels in DSA and ECDSA were addressed. These are some of the many issues found in an extensive systematic analysis of bignum usage by Samuel Weiser, David Schrammel et al.
- * Enabled openssl(1) speed subcommand on Windows platform.
- * Enabled performance optimizations when building with Visual Studio on Windows.
- * Fixed incorrect carry operation in 512 addition for Streebog.
- * Fixed -modulus option with openssl(1) dsa subcommand.
- * Fixed PVK format output issue with openssl(1) dsa and rsa subcommand.

The LibreSSL project continues improvement of the codebase to reflect modern, safe programming practices. We welcome feedback and improvements from the broader community. Thanks to all of the contributors who helped make this release possible.

- First 3.0.X release which will eventually be part of OpenBSD 6.6
- Completed the port of RSA_METHOD from OpenSSL 1.1
- Cleaned-up documentation
- Fixes in:
 - carry operation in Streebog
 - -modulus option for DSA
 - PVK format for DSA & RSA
- Build optimizations



OpenZFS on OS X 1.9.2 released

1.9.2 New Stable Release!

POSTREPLY Search this topic...

1.9.2 New Stable Release!

by **lundman** » Fri Aug 02, 2019 9:31 pm

OpenZFS_on_OS_X_1.9.2.dmg 2019-08-01

file size: 73400320

md5: fc82968e7a705e42ec2e0981f1037fb2 OpenZFS_on_OS_X_1.9.2.dmg

sha256: 125cfe675b6c65518184534f4822e44b3e59874e18828a35367d2467ce801d08 OpenZFS_on_OS_X_1.9.2.dmg

Get it here:

<https://openzfsonosx.org/wiki/Downloads#1.9.2>

New since 1.9.0 (1.9.1):

CODE: SELECT ALL

```
skip AVX512 vectorised functions (panic on modern CPUs)
include Mavericks.pkg
zfs rename could cause core dumps/abort
-1.9.1-

zfs commands "lost error messages"
zfs raw recv fix
zfs spill block recv fix
ZoL vectorized raidz / assembler for fletcher/aes port
zfs send deadlock in bqueue
```

Notarize can sometimes give installation error "kextcache code=71". Reason is unknown, but install completes successfully.

Last release was the officially the last release for 10.8 and 10.9. But due to popular demand, we included 10.9 anyway.

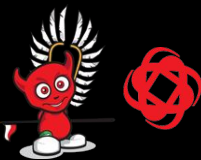
Clicking "Allow" button to load the KEXT on 10.13 can't be done over remote connections (VNC), but must be done on local console.

<https://openzfsonosx.org/wiki/Changelog>

- Fixed some core dumps & aborts after zfs rename
- Fixes in:
 - zfs raw recv
 - zfs spill block recv
 - zfs send deadlock in bqueue

Source: <https://openzfsonosx.org/forum/viewtopic.php?f=20&t=3296>

OpenBSD binary updates for security and other problems in Base OS



```
List: openbsd-announce
Subject: OpenBSD -stable binary packages
From: Solene Rapenne <solene () openbsd ! org>
Date: 2019-08-14 10:29:24
Message-ID: 9d24886e1145c157 () solene ! perso ! local
[Download RAW message or body]

The OpenBSD base system has received binary updates for security and
some other important problems in the base OS through syspatch(8) for the
last few releases.

We are pleased to announce that we now also provide selected binary
packages for the most recent release. These are built from the -stable
ports tree which receives security and a few other important fixes:

-release: fixed point in time, no update (6.3, 6.4, 6.5, ...).
-stable: conservative updates only. For ports, only the most recent
release is updated (currently 6.5).
-current: main development branch, receives bigger changes.

Initial updates for amd64 are already available at most mirrors (check
for the /pub/OpenBSD/6.5/packages-stable directory). i386 is currently
building and will follow soon. If the mirror you are using is not synced
yet, you will need to wait or use a different one.

pkg_add(1) already had the required heuristic to manage -stable
packages. It will be able to use the /packages-stable/ directory in the
following two cases:

1. you use /etc/installurl and the PKG_PATH environment variable is not
set (default installation case)
2. you use the PKG_PATH environment variable and it uses %c or %m

The two directories are separate because the "packages" directory holds
the packages built at the release time. They will not be updated.

The packages-stable directory will be empty at the time of a new
release. Its contents will grow during the release life cycle as
security fixes and other fixes are committed to the -stable ports tree.

If pkg_add(1) installs a new package and you meet the conditions for
using the packages-stable directory, detailed above, the version in
packages-stable will be chosen instead of the original supplied at
release time. This also applies when using 'pkg_add -u' to upgrade
packages.

This means that, in a default installation, pkg_add will automatically
pick the latest version available to you.

In the case of updating an installed package, this may require
restarting the running binaries to use the new code.

More info on the package system can be found at the following link:
https://www.openbsd.org/faq/faq15.html

Surprisingly, nobody saw the new directory show up on our mirrors, and
then report it on our mailing lists.
```

- **release:** fixed point in time, no update (6.3, 6.4, 6.5, ...).
- **stable:** conservative updates only. For ports, only the most recent release is updated (currently 6.5).
- **current:** main development branch, receives bigger changes

Source: <https://marc.info/?l=openbsd-announce&m=156577865917831&w=2>

OpenBSD Ada bindings for pledge & unveil



[Index](#) [Comments](#)

This library was written to help my easing into Ada programming, as I struggle with this language so different from all others I've known. This is a package I can be proud of, however. This defines a simple package for OpenBSD's extra facilities, namely pledge and unveil. There is an enumeration type common to both, two more enumeration types, two array types thereof, two exceptions, and then two procedures Pledge and Unveil. This is a simple and high-level interface to OpenBSD's pledge and unveil facilities that came to mind in my errant thought. The procedures are expected and designed to be used exclusively with array aggregates and not with the type names. Here is the example that inspired the library:

```
with OpenBSD; use OpenBSD;
...
Unveil("/tmp/", (Read | Write => Allowed, others => Disallowed));
...
Pledge((Stdio | Fattr => Allowed, others => Disallowed));
...
```

These procedures work by fixing Ada's enumeration types over the strings OpenBSD uses to approximate such a language facility. The pledge accepts a simple string of designated keywords separated by spaces and terminated according to the usual C convention. The unveil accepts a filename and string of designated characters terminated according to the usual C convention, as well. Failure cases are invalid pointers, malformed strings, invalid requests, attempts to increase permissions, and so on. Several of these failure cases simply won't occur in Ada; the permission errors are those that are particularly relevant and correspond to the exceptions, with unveil conflating some of the failure cases for simplicity, and this is a reasonable decision given that the precise error can be learned.

I'm, of course, still rather green to Ada and so it can be expected the body of this package will be improved over time. I likely won't change any of the names used in the specification, but this is a reason you should only use array aggregates with it. This library is not currently robust to change as I believe it should be, but I intend to correct this. It has been a valuable learning experience and I expect to update this library as OpenBSD changes the semantics of pledge and unveil or adds new facilities that would be worthwhile to expose to Ada.

Here is [the package specification](#), [the package body](#), and [the documentation](#).

Written in 2019 by Prince Trippy.

This page is available under the CC0 Public Domain Dedication.

Source: <http://verisimilitudes.net/2019-07-27>



New firewall test suite on FreeBSD

Revision 350586

Jump to revision:



Author:

thj

Date:

Mon Aug 5 11:47:34 2019 UTC (2 weeks, 2 days ago)

Changed paths: 7

Log Message:

Add common firewall test suite

Add a common test suite for the firewalls included in the base system. The test suite allows common test infrastructure to test pf, ipfw and ipf firewalls from test files containing the setup for all three firewalls.

Add the pass block test for pf, ipfw and ipf. The pass block test checks the allow/deny functionality of the firewalls tested.

Submitted by: Ahsan Barkati

Sponsored by: Google, Inc. (GSoC 2019)

Reviewed by: kp

Approved by: bz (co-mentor)

MFC after: 2 weeks

Differential Revision: <https://reviews.freebsd.org/D21065>

Test suite for pf, ipfw and ipf;

Source: <https://svnweb.freebsd.org/base?view=revision&revision=350586>

OpenBSD ported Electron



Bryan Steele 

@canadianbryan

Follow

robert@ has ported Electron to OpenBSD by integrating it into our chromium port, it also uses the current stable version and not some older copy, avoiding duplicating some 600+ local patches.

Henry, shield your eyes! @qb1t

Sadly no pledge/unveil.. yet.

Source: <https://twitter.com/canadianbryan/status/1164632273519611904?s=19>



More updates for using OpenBSD on Lenovo X1

OpenBSD Support Log

2019-07-26: First boot of the OpenBSD installer panicked fairly early with an AML error "Not Integer". Booting a non-ramdisk kernel with `ACPI_DEBUG` enabled showed this was due to a problem with the touchpad's `_INI` method. I eventually [tracked this down](#) to a problem with OpenBSD's implementation of `ToHexString`, a fix for which has been [committed](#).

2019-07-29: Now that the system boots, I noticed that key repeating on the console was broken, and that `date; sleep 1; date` showed it taking 3 or 4 seconds. In the past this has been due to an unsynchronized TSC, but even with `acpihp0` being the new default `kern.timecounter.hardware`, it still showed this problem. Some further debugging pointed to `cpu0: apic clock running at 100MHz when it should be 24Mhz`. By default, the X1 Carbon ships with its BIOS option "8254 Timer Clock Gating" enabled, and OpenBSD uses the 8254 for APIC clock calibration. This was [fixed](#) by fetching the CPU frequency directly from the CPU instead of timing it, in order to avoid relying on the 8254.

Despite [fixing](#) the `ihidev` polling issue, the X1C7's touchpad uses GPIO interrupts and requires a new Cannon Lake GPIO driver. I'll need to work on this.

2019-08-07: I realized that the sound from the speakers was lacking bass on OpenBSD but sounded fine on Linux. A tweak with `mixerctl` will properly hook up the `speaker2` outputs to the proper DAC, enabling proper sound on OpenBSD.

```
echo outputs.spkr2_source=dac-0:1 >> /etc/mixerctl.conf
```

2019-08-13: I [committed](#) a quirk to the `azalia` driver to do the speaker routing by default without having to use `mixerctl`.

- lacking bass on OpenBSD;
- fix for msrs & pat on main CPU -- slow console on x1r7.



FreeBSD April-June 2019 Status Report

FreeBSD Team Reports

- [Continuous Integration](#)
- [FreeBSD Core Team](#)
- [FreeBSD Foundation](#)
- [FreeBSD Graphics Team status report](#)
- [IRC Admin](#)
- [Ports Collection](#)
- [Release Engineering Team](#)

Projects

- [bhyve - Live Migration](#)
- [bhyve - Save/Restore](#)
- [BIO_DELETE support for the swap pager](#)
- [ENA FreeBSD Driver Update](#)
- [FreeBSD SDIO and Broadcom FullMAC WiFi Support](#)
- [FUSE](#)
- [Fuzzing FreeBSD with syzkaller](#)
- [Kernel ZLIB Update](#)
- [Linux compatibility layer update](#)
- [Lock-less delayed invalidation for amd64 pmap](#)
- [Locking changes for vnodes during execve\(2\)](#)
- [Mellanox Drivers Update](#)
- [NFSv4.2 client/server implementation for FreeBSD](#)
- [NUMA awareness in the FreeBSD kernel](#)

Architectures

- [Broadcom ARM64 SoC support](#)
- [NXP ARM64 SoC support](#)

Third-Party Projects

- [Aberdeen Hackathon](#)
- [Bring more Security Intelligence to FreeBSD](#)
- [libyds - QOW2 implementation](#)
- [nsysctl 1.0](#)

Example reports for anyone:

- not a programmers:
 - link to a presentation of the 2019 FreeBSD survey results at BSDCan 2019
- more technical:
 - news about git in FreeBSD;
 - status of some error detection on security tools,
 - announce the sysctl clone
- experienced:
 - updates to the linux compatibility layer;
 - much low level work on graphics;
 - many new bhyve features;
 - more user-friendly experience with trackpoints and touchpads enabled by default

Source: <https://www.freebsd.org/news/status/report-2019-04-2019-06.html>



BUGS



Reported CVE in BSD

- Reference count overflow in mqueue (FreeBSD 11.2 - 12.0) [CVE-2019-5603]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:24.mqueueufs.asc>
- Kernel memory disclosure from /dev/midistat (FreeBSD 11.2 - 12.0) [CVE-2019-5612]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:23.midi.asc>
- IPv6 remote Denial-of-Service (FreeBSD 11.2 - 12.0) [CVE-2019-5611]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:22.mbuf.asc>
- Insufficient validation of guest-supplied data (e1000 device) (FreeBSD 11.2 - 12.0) [CVE-2019-5609]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:21.bhyve.asc>
- Insufficient message length validation in bsnmp library (FreeBSD 11.2 - 12.0) [CVE-2019-5610]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:20.bsnmp.asc>
- ICMPv6 / MLDv2 out-of-bounds memory access (FreeBSD 11.2 - 12.0) [CVE-2019-5608]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:19.mldv2.asc>
- Multiple vulnerabilities in bzip2 (FreeBSD 11.2 - 12.0) [CVE-2016-3189, CVE-2019-12900]
<https://www.freebsd.org/security/advisories/FreeBSD-SA-19:18.bzip2.asc>



Reported errata

- ipfw(8) jail keyword broken prior to jail startup
<https://www.freebsd.org/security/advisories/FreeBSD-EN-19:17.ipfw.asc>
- Bhyve instruction emulation improvements (opcode 03H and F7H)
<https://www.freebsd.org/security/advisories/FreeBSD-EN-19:16.bhyve.asc>
- Incorrect exception handling in libunwind (FreeBSD 11.2, 12.0)
<https://www.freebsd.org/security/advisories/FreeBSD-EN-19:15.libunwind.asc>
- Incorrect locking in epoch(9)
<https://www.freebsd.org/security/advisories/FreeBSD-EN-19:14.epoch.asc>
- IPv6 neighbor cache leak on expiration (NetBSD 8.1 - current)
<https://ftp.netbsd.org/pub/NetBSD/security/advisories/NetBSD-SA2019-004.txt.asc>




CURIOSITIES



New GSoC update for incorporating the memory-hard Argon2 hashing scheme into NetBSD

The Guide | Manual pages | Mailing lists and Archives | CVS repository | Report or query a bug | Software Packages

**The NetBSD Project**
"Of course it runs NetBSD"

Home | RSS | Release engineering | Development | The NetBSD Foundation | Networking | General | Ports | Security | Events | Packages | Login

Bookmarks
The NetBSD Project
NetBSD Wiki

Feeds
All
/Release engineering
/Development
/The NetBSD Foundation
/Networking
/General
/Ports
/Security
/Events
/Packages
Comments

GSoC 2019 Report Update: Incorporating the memory-hard Argon2 hashing scheme into NetBSD
August 06, 2019 posted by Kamil Rytarowski
This report was prepared by Jason High as a part of Google Summer of Code 2019
Introduction
As a memory hard hashing scheme, Argon2 attempts to maximize utilization over multiple compute units, providing a defense against both Time Memory Trade-off (TMTD) and side-channel attacks. In our first post, we introduced our GSoC project's phase 1 to integrate the Argon2 reference implementation into NetBSD. Having successfully completed phase 1, here we briefly discuss parameter tuning as it relates to password management and performance.
Parameter Tuning
Both the reference paper [1] and the forthcoming RFC [2] provide recommendations on how to determine appropriate parameter values. While there are no hard-and-fast rules, the general idea is to maximize resource utilization while keeping performance, measured in execution run-time, within a tolerable bound. We summarize this process as follows
1. Determine the Argon2 variant to use
2. Determine the appropriate salt length
3. Determine the appropriate tag length
4. Determine the acceptable time cost
5. Determine the maximum amount of memory to utilize
6. Determine the appropriate degree of parallelism
Step 1
All three Argon2 variants are available in NetBSD. First, *argon2i* is a slower variant using data-independent memory access suitable for password hashing and password-based key derivation. Second, *argon2d* is a faster variant using data-dependent memory access, but is only suitable for application with no threats from side-channel attacks. Lastly, *argon2id* runs *argon2i* on the first half of memory passes and *argon2d* for the remaining passes. If you are unsure of which variant to use, it is recommended that you use *argon2id* [1][2]
Step 2-3
Our current implementation uses a constant 32-byte hash length (defined in *crypt-argon2.c*) and a 16-byte salt length (defined in *pw_gensalt.c*). Both of these values are on the high-end of the recommendations.
Steps 4-6
We parameterize Argon2 on the remaining three variables: time (*t*), memory (*m*), and parallelism (*p*). Time *t* is defined as the amount of required computation and is specified as the number of iterations. Memory *m* is defined as the amount of memory utilized, specified in Kilobytes (KB). Parallelism *p* defines the number of independent threads. Taken together, these three parameters form the *knobs* with which Argon2 may be tuned.

Report contains some recommendations and descriptions for:


- Argon2 variants;
- appropriate salt length;
- appropriate tag length;
- appropriate degree of parallelism;
- maximum amount of memory to utilize;
- acceptable time cost;

Source: https://blog.netbsd.org/tnf/entry/gsoc_2019_report_update_incorporating

Fuzzing NetBSD filesystems via AFL [Part 2]



[» The Guide](#) | [Manual pages](#) | [Mailing lists and Archives](#) | [CVS repository](#) | [Report or query a bug](#) | [Software Packages](#)

 **The NetBSD Project**
"Of course it runs NetBSD"

Home | [RSS](#) | [Release engineering](#) | [Development](#) | [The NetBSD Foundation](#) | [Networking](#) | [General](#) | [Ports](#) | [Security](#) | [Events](#) | [Packages](#) | [Login](#)

Bookmarks
The NetBSD Project
NetBSD Wiki

Feeds
All
/Release engineering
/Development
/The NetBSD Foundation
/Networking
/General
/Ports
/Security
/Events
/Packages
Comments

Fuzzing NetBSD Filesystems via AFL. [Part 2]

August 11, 2019 posted by *Kamil Rytarowski*

This report was written by Maciej Grochowski as a part of developing the AFL+KCOV project.

Recently I started working on Fuzzing Filesystems on NetBSD using AFL. You can take a look at the [previous post](#) to learn more details about background of this project. This post summarizes the work that has been done in this area, and is divided into 3 sections:

1. Porting AFL kernel mode to work with NetBSD
2. Running kernel fuzzing benchmark
3. Example howto fuzzing particular Filesystem

☞AFL Port for NetBSD

AFL is a well known fuzzer for user space programs and libraries, but with some changes it can be used for fuzzing the kernel binary itself.

For the first step to fuzz the NetBSD kernel via AFL, I needed to modify it to use coverage data provided by the kernel instead of compiled instrumentations. My initial plan was to replace the coverage data gathered via `afl-as` with that provided by `kcov(4)`. In this scenario, AFL would just run a wrapper and see the real coverage from the kernel.

I also saw previous work done by [Oracle](#) in this area, where instead of running the wrapper as a binary, the wrapper code was included in a custom library (.so object).

Both approaches have some pros and cons. One thing that convinced me to use a solution based on the shared library with initialization code was the potentially easier integration with remote fork server. AFL has some constraints in the way of managing fuzzed binary, and keeping it on a remote VM is less portable than fuzzing using a shared library and avoiding introducing changes to the original binary fuzzing. Porting AFL kernel fuzzing mode to be compatible with NetBSD kernel mainly relied on how the operating system manages the coverage data. The port can be found currently on [github](#).

☞Writing a kernel fuzzing benchmark

Performance is one of the key factors of fuzzing. If performance of the fuzzing process is not good enough, it's likely that the entire solution won't be useful in practice. In this section we will evaluate our fuzzer with a practice benchmark.

One exercise that I want to perform to check the AFL kernel fuzzing in practice is similar to a password cracking benchmark. The high level idea is that a fuzzer based on coverage should be much smarter than bruteforce or random generation.

To do this, we can write a simple program that will take a text input and compare it with a hardcoded value. If the values match, then the fuzzer cracked the password. Otherwise, it will perform another iteration with a modified input.

Instead of "password cracker", I called my kernel program "lottery dev". It's a character device that takes an input and compares it with a string.

Report contains information about:

- AFL port for NetBSD;
- plans to write a kernel fuzzing benchmark
- step by step examples how to fuzzing particular filesystem

Source: http://blog.netbsd.org/tmf/entry/fuzzing_netbsd_filesystems_via_afl



Half-Life 1 on OpenBSD -- confirmed



Source: <https://twitter.com/canadianbryan/status/1158512880217731079>

OpenBSD Olive: Adobe Premiere-like clone

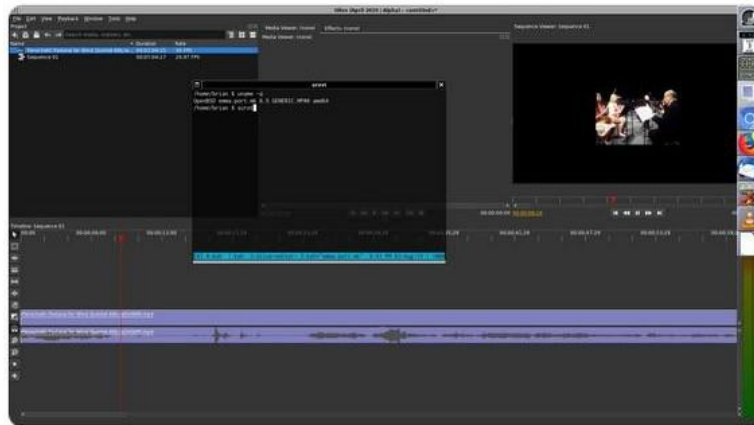


(((Dr. Brian Robert Callahan)))

@__briancallahan

Follow

Must be a video editing weekend. Got Olive, an Adobe Premier-like clone, working on #OpenBSD too!



3:46 PM - 3 Aug 2019

Source: https://twitter.com/__briancallahan/status/1157784738129752067



OpenBSD 6.5 booted from floppy disk on machine from 1996

- Successful network installation of OpenBSD 6.5 + LibreSSL + Perl.
- Machine specification:
 - Pentium 200 (Socket 7, no MMX),
 - 128 MB RAM, 4x SIMM,
 - 1 MB DRAM for integrated graphic S3 Trio64+.
- Installer size: ~ 1.44 MB.

Source:

<https://www.dobreprogramy.pl/wielkipiec/Instalator-OpenBSD-mikroskopijny-wyjatek-w-morzu-softwareowej-opuchlizny,103141.html>

Instalator OpenBSD: mikroskopijny wyjatek w morzu software'owej opuchlizny

wielkipiec · 11.08.2019 00:05



Jak byś może niektórzy pamiętają, poza zmywaniem naczyń w Redakcji, jestem też administratorem-wdrożeniowcem. Zajmuję się przygotowywaniem i obsługą platform automatycznych instalacji systemów Windows Server i Red Hat Enterprise Linux. Dotyczy to zarówno modeli bezstanowych (mądra nazwa na tępą kopię zawartości dysków), jak i - przede wszystkim - "bardzo stanowych", czyli nienadzorowanych instalacji sieciowych. Obrazy WIM, pliki odpowiedzi Kickstart, WSUS, Cobbler, pakiety MSI oraz RPM i tak dalej.

Relaks ze starociami

Gdy jakimś cudem mam jakiś strzęp wolnego czasu i nie piorę mi od monitora oczy, zajmuję się swoim hobby, którym jest składowanie starych pecetów. Jestem jak ten kierowca ciężarówki, który po pracy idzie się odprężyć grając w "Truck Simulator". Niestety, moje ziomowe hobby sprawia mi przyjemność na tyle, że koszty oddalenia się od racjonalnego zarządzania czasem i pulę tematyczną w życiu schodzą na dalszy plan. Dlatego niezmieennie cieszę mnie, gdy znajdę jakiś artefakt, który w zabawny sposób łączy oba te światy.



Intelowskie dziwadło NLX z 1996



New VCS on OpenBSD: Game of Trees



OpenBSD Journal

[Home](#) | [Archives](#) | [About](#) | [Submit Story](#) | [Create Account](#) | [Login](#)

Game of Trees

Contributed by [rueda](#) on 2019-08-10 from the got-to-do-things-properly dept.

Stefan Sperling (stsp@) is developing a version control system, "[Game of Trees](#)". From <https://gameoftrees.org/>:

Game of Trees (Got) is a version control system which prioritizes ease of use and simplicity over flexibility.

Got is still under development; it is being developed exclusively on OpenBSD and its target audience are OpenBSD developers.

Got uses [Git](#) repositories to store versioned data. At present, Got supports local version control operations only. Git can be used for any functionality which has not yet been implemented in Got. It will always remain possible to work with both Got and Git on the same repository.

GoT has been [added](#) to the ports tree as [devel/got](#).

It is the subject of a [talk](#) at [EUROBSDCON 2019](#).

Stefan has been involved in the [discussion on Lobste.rs](#).

Source: <https://undeadly.org/cgi?action=article;sid=20190810123007>



EVENTS



BSD Events in August

August 2019

- **COSCON 2019**, Taipei, Taiwan
17 - 18 August, 2019
- **Open Source Summit North America**, San Diego, United States
21 - 23 August, 2019



BSD Events in September

September 2019

- **vBSDcon 2019**
September 5-7, 2019, Reston, VA, USA.
- **EuroBSDCon 2019,**
September 19-22, 2019, Lillehammer, Norway.