

happened to him during the past year. It is the personal touch that makes this special every year.

FREEBSD/MIPS, EMBEDDING FREEBSD

Summarized by Bjoern A. Zeeb (bz@FreeBSD.org)

Warner Losh

Warner Losh began talking about the long history of FreeBSD/mips starting in the late 1990s with FreeBSD 3.x. The second try to bring it into the mainstream started in 2002 and the third one at BSDCan 2006, which led to more community success in getting to single users on real hardware. In 2007 Juniper released code that was later merged with the mips branch and gets to multiusers. FreeBSD/mips is self-hosting now. Today mips32/r2 and mips64/r2 are supported and FreeBSD runs on at least four SoC families: ADMTek ADM5120, IDT RC32432, Broadcomm MIPS, and the MIPS 4Ke core. More are to come soon. Currently the work is merged from the Perforce repository into mainline CVS.

Warner Losh then gave an update on the embedded FreeBSD world. Two Google Summer of Code students will work on a PowerPC port and on further reducing the footprint size of embedded FreeBSD. Both the PowerPC and ARM support develop well and there is more and more support in the repositories. He closed with an outline of future projects.

RESOURCE-LIMITING ON THE VIRTUAL PRIVATE SERVER

Summarized by Mathieu Arnold (mat@FreeBSD.org)

Fred Clift, Verio

NTT-Verio uses FreeBSD extensively on its virtual hosting services. They started at about the same time as jail(8), but they added a lot of things to it, such as limits, similar to rlimits, with io, network, mail inject, and syscall rate limit. Those limits were needed to have numerous jails without one taking all the resources. All those limits have clever algorithms that allow bursts, so that the virtual servers feel responsive when needed. They're waiting on the lawyers to release the code, which they have on FreeBSD 4, 6, and 7. Slides with nice graphs are available at <http://clift.org/fred/bsdcan2008.pdf>.

A CLOSER LOOK AT THE ZFS FILE SYSTEM/ZFS, THE INTERNALS

Summarized by Mathieu Arnold (mat@FreeBSD.org)

Pawel Jakub Dawidek

Pawel started by peeling ZFS like an onion, explaining how the components talk to each other and what they do. The best way to get an idea is to look at his slides at http://www.bsdcan.org/2008/schedule/attachments/58_BSDCan2008-ZFSInternals.pdf. He then explained

BSDCan: The BSD Conference

*Ottawa, Canada
May 16–17, 2008*

OPENING SESSION

Summarized by Bjoern A. Zeeb (bz@FreeBSD.org)

“BSDCan 2008, welcome back” was Dan Langille’s first slide. But before telling you about all the conference talks let’s go forward to the closing session to tell you one reason why these summaries were written.

Dan’s Rules of Conference:

1. You do talk about conference.
2. You DO talk about conference.
3. You shall not stand in a direct line between TV and Dan during an NHL game at conference.

In case you ignore rule #3 you’ll find out about #4, but that is left as an exercise to the reader.

So the opening talk started with a screensaver of lots of Nigeria s(clp)am asking for letters of invitation. Would you have imagined this happens to an organizer of a conference? Dan continued thanking all the sponsors, talked about the organizational things, and gave his talk, a summary of what

how RAID-Z is similar to RAID5/RAID6 but so different. The problem is that ZFS is both the filesystem and the RAID system, so it knows what to write and in what order. It's also self-healing because everything on disk is checksummed, so if it reads parity that's bad or data that's bad, it will rewrite what should be back to the disk. Also, every write to the disk has a transaction number, so a hard drive that's only been temporarily unavailable will only need to synchronize changes. As for snapshots, well, ZFS does copy on write; so it already does snapshots behind your back: taking a snapshot is O(1). Pawel fielded several questions, mainly about "how do I do that and get it all right," stating that adding disks is easy and replacing disks with bigger ones is almost as easy.

MEASURED (ALMOST) DOES AIR TRAFFIC CONTROL

Summarized by Mathieu Arnold (mat@FreeBSD.org)

Poul-Henning Kamp

Poul-Henning Kamp explains how he helped the Danish Air Traffic Control monitor 30-year-old navigation devices (DME, VOR, etc.). He set up devices based on NanoBSD (stripped down FreeBSD configured to run on a compact flash card) to run the MeasureD daemon. The daemon gets information in many different ways, mostly through obscure serial protocols (almost reverse engineered) from all the devices, transmitters, fuel gauges, battery voltage levels, and so on. MeasureD multiplexes all it gets and incorporates a small HTTP daemon in which you can get all the data you want. (Another MeasureD can also get the data, say, from a central location.) There is also a bsnmp plug-in so you can get MeasureD data from SNMP. All that data has to be logged somewhere, and with the storage device being a flash card it has to be done in a sensible manner. Kamp wrote fifolog, which time-stamps and multiplexes its input, compresses it, and stores it regularly (with padding if the compressed size does not fill a data cell). It can be grepped through pretty easily with the `fiforead` command. See http://www.bsdcn.org/2008/schedule/attachments/64_BSDCan2008-AirTrafficControl.pdf for more details.

SCTP

Summarized by Florent Parent (florent.parent@beon.ca)

Randall Stewart

SCTP, or Stream Control Transmission Protocol, is a new transport protocol that sits above IPv4 and IPv6, at the same layer as TCP and UDP. SCTP has been standardized by the IETF in 2000 and is available in FreeBSD 7.0.

In this talk, Randall Stewart presented an overview of SCTP. New features offered by SCTP are four-way handshakes (which help reduce DoS attacks), framing (which

is used to preserve message boundaries), multistreaming, multihoming, and reachability.

SCTP multistreaming allows logical separation of data within an association. Stewart demonstrated video multistreaming using a Web page download containing multiple pictures. The download was done over a path exhibiting packet loss. SCTP took about half the time that TCP took to do the same thing.

The SCTP socket API was also presented. The API offers two socket types: one-to-one (STREAM) and one-to-many (SEQPACKET). The one-to-one interface offers backward compatibility with TCP sockets. The one-to-many offers a UDP-like interface and is the type of interface a peer-to-peer application would use.

Overall, this talk was an excellent introduction to the benefits of SCTP. The Web site <http://www.sctp.org> is SCTP equipped.

PORTING FREEBSD/ARM TO MARVELL ORION SYSTEM-ON-A-CHIP

Summarized by Mathieu Arnold <mat@FreeBSD.org>

Rafal Jaworowski, Semihalf

System-on-a-Chip (SoC) is an integrated chip with many things integrated, such as the main CPU, GPIO (General Purpose I/O), Ethernet, UART, PCI, USB, SATA, Crypto, and SPI. The FreeBSD/ARM port to that SoC family is working pretty well, with new chips being added every now and then and much work still not committed to the FreeBSD source tree. For more information see http://www.bsdcn.org/2008/schedule/attachments/50_2008_marvell_freebsd.pdf.

GOOGLE SUMMER OF CODE

Summarized by Mathieu Arnold (mat@FreeBSD.org)

Leslie Hawthorne, Google

Leslie Hawthorne is the Project Manager of the Google Summer of Code program. She came here to explain how the idea came into being at Google—"Here, we have too much money; let's spend some"—and also why they're not evil—"We're giving you money and ask nothing in return." Since the beginning, the Summer of Code project has involved participation from 1500 students, 2000 mentors, 175+ open source projects, and 98 countries, and more than 10 million U.S. dollars have been spent on students and projects. Details can be found at http://www.bsdcn.org/2008/schedule/attachments/52_LeslieHawthorn_bsdcn2008.pdf.

UP CLOSE AND PERSONAL WITH TCP IN FREEBSD

Summarized by Florent Parent (florent.parent@beon.ca)

Lawrence Stewart

Lawrence Stewart presented a look at a new modular TCP congestion control framework. The talk started with a review of the current state of TCP. (RFC4616 is a good summary of TCP-related RFCs.) Today, NewReno TCP is the de facto standard and is used in BSD and many other OSes. But there are still open issues in high-speed networks, where improvements to current congestion control protocols are required.

Many new TCP congestion control protocols have been proposed, and some of them are being used in Linux (CUBIC) and Windows Vista (CTCP). A new modular framework for TCP congestion control (CC) protocols has been developed in FreeBSD. This framework allows the support for new CC protocols in addition to NewReno.

Lawrence Stewart showed a demo where the TCP congestion control protocol selection was done using the `sysctl` command.

The tool SIFTR (Statistical Information For TCP Research) was presented. SIFTR is a kernel module that logs to a file different statistics on TCP connections. This tool helps the development and testing of new protocols. Many other tools are used, such as `dummysnet`, `iperf`, `tcpstat`, `R`, and `ns-2`.

There is still work and testing to be completed, and there is a plan to share the congestion control protocols between TCP and SCTP. The home page for this work (including code) is found at <http://caia.swin.edu.au/urp/newtcp/>.

OPENBSD HARDWARE SENSORS FRAMEWORK

Summarized by Constantine A. Murenin (cnst@openbsd.org)

Constantine A. Murenin

Constantine is a math graduate student at the University of Waterloo, a committer at OpenBSD, and a Google Summer of Code 2007 (SoC2007) student at FreeBSD. During this Invited Talk, an overview of the sensors framework was presented, including the recent developments in the driver arena and the Google SoC2007 experience of the speaker.

The talk started with a brief introduction to the main ideas behind a unified sensor framework and followed with some numbers representing the pervasiveness of the framework within OpenBSD: At the end of March 2008, a 64th driver utilizing the interface was committed into the code tree, representing an anniversary of some kind, and at the time of the presentation the number of drivers calling `sensor_dev_install(9)` was as high as 67. Some of these drivers are unique to OpenBSD and are not yet available elsewhere; for example, Theo de Raadt has recently added support for the JEDEC JC-42.4 SO-DIMM temperature sensors, and Constantine has provided support for the temperature sensors embedded in AMD Phenom and Opteron Barcelona

processors (neither of which is yet available in the Linux `lm-sensors` package).

The rationale behind the framework design was explained, with the primary objective of the API being “simple, secure, and usable.” An example was given on how the voltage sensors work in the hardware monitoring modules of most popular Super I/O solutions, where it is often the case that it is impossible to know the true relationship between the sensors and the power lines of the power supply unit, so an overengineered framework isn't likely to be beneficial for most simple drivers aimed at being usable by default. An overview of the API and of the userland utilities was presented. Userland utilities include `sysctl`, `sensorsd`, `ntpd`, `systat`, `snmpd`, and `ports/sysutils/symon`.

Constantine then proceeded to describe his experience with porting the framework to FreeBSD, sponsored by the Google SoC2007 program. Most popular parts of the framework were ported, and a complete patchset was publicly released on September 13, 2007, but the FreeBSD CVS HEAD tree was frozen at the time because of the then-upcoming RELENG_7 branching, limiting the integration of the new components into the tree.

In the meantime, the framework was committed to DragonFly BSD by Hasso Tepper, who within a few days of the posting adapted Constantine's patch for inclusion into DragonFly. A few days later, Constantine's patch was approved by the FreeBSD Release Engineering team to be committed into FreeBSD after RELENG_7 was branched, and on October 14, 2007, it was committed into the FreeBSD CVS repository by Alexander Leidinger. The commit has generated a lot of attention in the FreeBSD community, and some people suddenly felt that the framework itself was designed only with the OpenBSD architecture in mind and didn't have a FreeBSD feel. The framework was then backed out from the FreeBSD CVS tree within a few days upon a request from Poul-Henning Kamp, who perceived it as not being architecturally fit for FreeBSD, even if it may be appraised in OpenBSD, as FreeBSD and OpenBSD have different architectural goals in mind, although Poul-Henning specifically clarified on the mailing lists that the SoC2007 porting itself was done to his satisfaction.

Poul-Henning was present in the audience, and during the comments-and-questions portion of the talk he clarified that he doesn't want the code being imported into FreeBSD so that FreeBSD could have a clear space in the area and someone could implement a framework more suitable for FreeBSD sometime in the future. Given that the framework in question was based on a framework that was available in NetBSD since 1999, Constantine and Poul-Henning agreed that designing and implementing a sensors framework perfect for FreeBSD with usable drivers may not be easy (especially considering the often-inadequate hardware specifications in the area).

X.ORG

Summarized by Constantine A. Murenin (cnst@openbsd.org)

Matthieu Herrb

Matthieu Herrb is an OpenBSD X Window System developer and a member of the Board of Directors of the X.Org Foundation. In this Invited Talk, Matthieu provided some historical insights into the development of the X Window System, from the late 1980s and the time of the monochrome and 8-bit color framebuffers, to the present time of Direct Rendering Infrastructure (DRI) and anti-aliasing support.

Major administrative milestones were briefly covered (e.g., the XFree86 license fiasco and the creation of the X.Org Foundation). Since the establishment of the foundation in 2004, the two most visible changes to X were the conversion of the build system to a modularized design and the embracement of git, the distributed version control system.

The interaction between various components of the windowing system were described, both the direct and indirect rendering modes, as well as plans for the future regarding DRI2 and Gallium 3D, which aim at making the Direct Rendering Manager simpler and more in line with the architectural characteristics of modern 3D hardware.

Matthieu also summarized major changes in X.Org 7.3, which was released in September 2007, and future plans for the upcoming 7.4 release. For example, in late 2007 and early 2008, AMD has released free programming documentation that made the new radeonhd driver possible, and 7.4 will be the first release to include this new driver, supporting ATI Radeon HD chipsets.

Last but not least, the situation with X around the BSD systems was covered. OpenBSD 4.3 includes X.Org 7.3, and work on the DR is underway based on the code from NetBSD. However, NetBSD is the only significant system that continues to ship XFree86 releases in its base system, although X.Org is also available through pkgsrc, a managing facility for third-party packages. One interesting problem that OpenBSD and NetBSD face is on the legacy architectures front, where some architectures have limited processor and memory resources and sometimes lack shared library support, making it increasingly difficult to support newer X releases on such machines.

BSD LICENSED C++ COMPILER

Summarized by Constantine A. Murenin (cnst@openbsd.org)

Chris Lattner, Apple, Inc.

Chris Lattner is the chief architect of the Low Level Virtual Machine Compiler Infrastructure, currently managing the LLVM group at Apple Inc. In this Invited Talk, Chris provided an outline of the technological advances of the BSD-like licensed LLVM compiler suite.

In a nutshell, the LLVM project consists of the language-independent optimizer and code generator, llvm-gcc 4.2 front-end, and the clang front-end. In the introduction to the talk, Chris described why there is a need for a new compiler technology, which, for someone somewhat familiar with the GNU Compiler Collection, wasn't all that surprising: GCC keeps getting slower with every release, cannot be easily reused in other applications, and is bloated to the point where it is quite difficult to read and modify the code. LLVM, however, takes a modular approach, where more components can be shared across different compilers and processor architectures.

The GCC 4.x design was highlighted, and this was followed by an explanation of how the llvm-gcc 4.2 is designed to work: llvm-gcc is a drop-in replacement for the gcc and uses the GCC front-end with the LLVM optimizer and code generator. Chris has reported that not only are the LLVM optimizer and code generator faster in such a GCC compound (30% improvement at -O3) but they also produce better code (5% to 10% improvement on an x86/x86-64); moreover, they allow some interesting applications, such as just-in-time compiling, optimization of the C/C++ code, and generation of the executable code at the install time.

Special attention was devoted to one other topic, namely, LLVM on the OpenGL front. Mac OS X 10.5 provides mechanisms for colorspace conversion, code for which has hundreds of conversion combinations among the color formats, and patterns can be applied to the input and output in the "case" statements inside a couple of "switch" statements inside a "for" loop for every pixel. For such an example, run-time optimization can greatly improve the performance, with 5.4x being the average improvement and 19.3x being the maximum speed-up, depending on the source and destination color formats. Some insights were also given regarding the Mac OS OpenGL state before LLVM came about. Chris was happy to note that no polygon can get onto the screen in Mac OS X without LLVM.

The next big part of the talk was a presentation on clang, a front-end for C, C++, and Objective-C. In comparison, GCC, apart from being slow and memory-hungry, doesn't serve the needs of various IDE applications, such as indexing of the code for "jump to the definition" features or "smart-editing." One of the most significant problems with GCC front-end is, however, the limited information that is usually provided when the compiler encounters errors in the source code. Since clang always keeps the information about the columns where the errors occur, error messages explicitly contain not only the line but also the column, as well as providing ASCII graphics of the exact point on the line where the errors occur, accompanied by a more meaningful error message than the GCC usually offers. This feature was very well received by the audience.

INTRODUCTION TO DEBUGGING THE FREEBSD KERNEL

Summarized by Bjoern A. Zeeb (bz@FreeBSD.org)

John H. Baldwin, Yahoo! Inc.

John Baldwin started with an overview of the various places to find documentation on the subject. He continued by showing how to use the interactive kernel debugger for investigating deadlocks. For the developers in the audience he talked about how to enhance ddb(4) by adding new commands. The next section was on the kgdb(1) debugging kernel modules and scripting by adding user-defined commands. He closed with a summary on the different strategies for debugging kernel crashes versus system hangs. Slides and a paper with more information are available online at <http://people.freebsd.org/~jhb/papers/bsdcn/2008/>.

WIPS

Summarized by Mathieu Arnold (mat@FreeBSD.org)

- * Ivan Voras on mdcached: It is a caching daemon, much better than memcached, optimized for multicore servers, and very fast; you can put tags on data and search with tags, which seems to be a handy idea.
- * Frank Pikelner on Versiera: This multi-OS server management software, developed by Netcraft, seems nice.
- * Philip Paeps on syscons and other scary things: Philip says he came into device drivers because of a touchpad that was working rather strangely. He's also going to break syscons in the upcoming months by taking it apart and separating it from the framebuffer and ttys.
- * Peter Loshier on IPv6 and the root name servers: We'll be out of IPv4 in two years, so get used to it, but 6 of the 13 root servers have had IPv6 for quite some time now, and those IPs have been added to the root zone file.
- * Bjoern Zeeb on multi-IPv4/IPv6/no-ip jails: There have been multiple patches for multiple IPv4, vimage, IPv4+IPv6, and jailv6, and some things are moving along nicely. Ultimately, we'll have DDB and SCTP support. The current system is pretty light, and it works in production. Things to do include source-routing; cpuset selection; adding a name to the jail so that it can be put in ps, for instance; support for bsnmpd; and adding resource limits.
- * Zachary Loafman on FreeBSD at Isilon: They have a distributed filesystem called OneFS, sponsored the work to have NFSv3 locks working, and have tons of other things (as shown in too many interesting slides with no time to take notes), but they lack time to contribute them and are hiring and willing to sponsor projects.
- * Mark Linimon on Bugathons + BugBusting BoF: Bugathons bring volunteers, and it really helps to categorize PRs; volunteers bring in lots of fresh blood and also many fresh interesting ideas, so if you want, you can help too.

* Julian Elisher on Vimage, MRT: The kernel modules will be virtualized one after another.

* George Neville-Neil on network testing: TCP is king, in general, and peasants like multicast and UDP get much less testing. The network test utility mctest, with sources and sinks, is in src/tools/tools/mctest. PCS is another network tester and has been improved; this year it's a Google SoC project.

* George Neville-Neil on XEN: HEAD works with Xen 3.1 and 3.2 in perforce, and Xen 3.0.3 is in the pipe for Amazon EC2. It'll happen right after FreeBSD switches to svn, and it will support 64-bit architectures.

* Julian Elisher on multiple routing tables: He showed two big schemas with lots of structures, pointers, and other stuff; one is bad because the API changes too much; the other is good because it does not change that much.

THE VERY END

Summarized by Bjoern A. Zeeb (bz@FreeBSD.org)

If you want to find out how Dan felt about the conference, go to his blog at <http://dan.langille.org/> and check for "Three Weeks in the Life of a Conference Organizer." Thank you, Dan, for another fantastic BSDCan! Is this the end? No. Do you want to know what else happened those days? You had better come and find out yourself next year. See you at BSDCan 2009, when it will be the biggest OS(S) conference in town.