FreeBSD Developer Summit and BSDCan

Summarized by Rik Farrow

On May 10, I headed off to Ottawa, Canada, for a several-day adventure with the three BSD communities. BSD, which started off as the Berkeley Software Distribution when Bill Joy arranged to ship out nine-track tapes containing assorted software (such as vi and csh, which he wrote, and sendmail), has forked twice into three groups. FreeBSD, the largest community, focuses on building a mainstream server/network operating system, with multiprocessor support. NetBSD, the next largest community, specializes in porting the BSD operating system to as many target CPUs as possible. Currently, 59 CPU architectures are supported. OpenBSD, a fork from NetBSD, is best known for its focus on improving security.

I caught the second day of the FreeBSD Developer Summit, an invitation-only meeting of about 50 developers. Eight long talks were packed into a long day, with a pub trip for lunch. Having a pub break somewhat disturbed my note-taking ability, but I will provide you with an overview of the talks, as well as some links if you want to search deeper.

The Developer Summit is a chance for FreeBSD developers to meet in person to catch up on the status of projects and plan for future work. Another key aspect of the summit is the chance for developers to meet each other in person—something that's especially important given the limitations of electronic communication.

The morning began with Dario Freni and Scott Ullrich dis-

cussing a LiveCD version of FreeBSD, called FreeSBIE (www.freesbie.org). You can use FreeSBIE like Knoppix, a popular Debian Linux–based LiveCD; that is, you can boot from the CD and use FreeBSD without installing anything on your hard drive. The developers described how the image can be made small enough to fit on a business-card-sized CD (8 MB), using a new toolkit called sysutils/freesbie, and to create other purpose-built CDs using FreeSbie.

Next off, Colin Percival described his Update 2.0 project. Percival recently became the FreeBSD security officer. Updating a FreeBSD system currently involves either collecting new sources and performing a make world in /usr/src or installing from scratch (which does ensure a clean upgrade). The Update 2.0 system supports installing binary security patches, making installed FreeBSD systems easier to maintain. For now, the system only works with security patches. Apple and the Mozilla Foundation use a version of Update v1. Part of making the binary update system officially supported involves moving it to a formal project infrastructure rather than using ad-hoc systems he's assembled previously, and part is making it a tool that can be reused by administrators to deploy their own updates, not just the security updates.

The KAME project involved the creation of a reference IPv6 and IPSec implementation for BSD operating systems in general. KAME began with Japanese researchers (http://www.kame.net/project-overview.html), and the IPv6/IPSec implementation has been merged into FreeBSD since 4.0. The KAME project has not supported the FreeBSD SMP architecture introduced in FreeBSD 5.0, so the FreeBSD stack has become the authoritative source of general IPv6 code.

Robert Watson spoke for the first of many times. Watson (http://www.watson.org/~robert/) has added auditing capabilities to TrustedBSD, a version of FreeBSD. The auditing support is based on Apple's audit implementation as found in Darwin, and it uses the same format as Sun's BSM, as there are already tools available for perusing those audit records. Audit records of this type refer to secure operating systems in the tradition of the Orange Book, and now the Common Criteria.

He has also decided to add NFSv4-style ACLs to the existing POSIX.1.e-style in TrustedBSD, and he hopes that Kirk McKusick will implement backup and restore support for ACLs. The decision to retrofit/update the MAC framework is based on four years of deployment experience, which ends up being mostly cleanup, since there are a number of companies shipping with the framework and they are, in fact, generally happy with it.

Watson also talked about needing to reduce the number of firewalls supported in FreeBSD from four to three (really!). Ip6fw will be eliminated, as ipfw now has full IPv6 support. The other firewalls supported in FreeBSD are pf and ipfilter.

Sam Leffler picked up after lunch. Leffler writes code for 802.11 infrastructure support for various wireless devices. Although many of the devices are Linux-based, Leffler prefers to begin working in the FreeBSD programming environment. Some of the work Leffler has done has not been folded into the system, because programmers need to modify existing drivers so that they will work with the extensions he has written for wireless roaming, repeaters, virtual APs, and WDS (Wireless Distribution System).

Requests for someone to take responsibility for some part of the kernel code were not uncommon during this conference. Anyone can become a part of the FreeBSD community by contributing patches, even for documentation. The more time and useful patches or code you contribute, the more important you become to the community. From an outsider's perspective, this concept looks very appealing and straightforward.

Randall Stewart of Cisco spoke next about SCTP. SCTP appeared over five years ago as an alternative to TCP, and Stewart wrote both a reference implementation and a book about SCTP. SCTP sets out to solve many of the weaknesses of TCP and includes the ability to multiplex streams within a single connection. Although there are five new system calls involved with SCTP (and kernel support also in Linux 2.6 and Solaris 10), there are currently no FreeBSD man pages. But SCTP has been used in telephone applications in China, Cisco BGP implementations (because it has protection against RST attacks), SIP proxies, and satellite communications.

Robert Watson took over at this point, covering a myriad of topics very quickly. Besides his work in secure systems, Watson has been at the forefront in removing the Giant lock from the networking stack (see Michael Lucas's article in the October 2005 *;login:*). Watson explained where Giant had been removed, then mentioned that there were device drivers where Giant is still used (which hurts SMP performance). Watson called for people willing to rewrite some critical device drivers (which is not an easy task), and also for people willing to measure the performance of

applications such as Apache and MySQL (LAMP) on multiprocessor servers.

Marco Zek spoke briefly about IP stack virtualization. Stack virtualization provides multiple network stacks, for class exercises, honeypots, and to use with jails. The combination of jail technology with stack virtualization provides a significantly lighter-weight implementation than other OS virtualization approaches such as Xen. At the end of his talk, Zek set up a demonstration where people in the audience could hook up to a virtual network he had created using a GUI-based, drag-and-drop tool via an access point attached to his laptop. Network traffic in the virtual network could be seen within the GUI.

The final session of the day covered the use of FreeBSD in embedded systems. Poul-Henning Kamp mentioned that there is a need for a flash file system, one that levels writes to flash to extend its lifetime, as well as improvements in gdb for debugging over a serial port. There is an embedded system mailing list, small@freebsd.org, for people interested in FreeBSD on embedded systems. Warner Losh spoke about work he has done porting FreeBSD to embedded systems, and he showed examples of hardware he had worked with.

The general observation was made that FreeBSD is being widely used in the embedded and high-end embedded spaces, but that FreeBSD developers need to do a better job of supporting that community. Particular targets are providing a better Web site and online community, providing reference hardware information for platforms such as ARM and MIPS, and working on improving bundling and targeting tools for embedded environments.

The formal sessions had ended at this point, but after dinner many FreeBSD developers gathered in the eighth-floor lounge in the residence hall (where many people were staying) for hacking into the wee hours. This went on at least until Saturday night.

BSDCAN

Dan Langille began organizing the BSDCan conference several years ago. The USENIX Association was one of two large donors in 2006 and helped to pay the expenses of the speaker travel at this three-tracked conference (http://www.bsdcan.org/2006/). The conference is held at the University of Ottawa in classrooms in the SITE (computer science) building. The low cost of the conference, preconference tutorials, and on-campus accommodations help make this conference popular with those with small budgets. There were 193 attendees this year.

Langille began by explaining where to eat (pubs) and that there would be wireless access as soon as the University of Ottawa provided a route and a hole in their firewall, and then he announced other BSD gatherings. There will be a EuroBSDCon in Milan, Italy, sometime in the first half of November 2006, an AsiaBSDCon in Japan in March 2007, then another EuroBSDCon in Copenhagen in December 2007.

As there were three tracks and only one summarizer (me), all I can do is share some of the notes I took from the sessions that I attended. I first listened to Russell Sutherland of the University of Toronto talk about using FreeBSD in edge routers. He had tried using Linux because it includes policy-based routing, but he prefers FreeBSD's ipwf over iptables. Although the default route is to Internet2, he forwards traffic from dorms as well as commercial Internet traffic away from Internet2 (see Robert Haskins' article about packet shaping in this issue).

Poul-Henning Kamp (http://phk.freebsd.dk/) spoke next. Kamp is well known for his work with embedded systems and FreeBSD appliances. He showed pictures of multiple Soekris (soekris.com) boxes nailed up on the wall of his workshop, serving as firewalls, routers, and servers. Kamp explained that most disk drives were limited to room-temperature environments, making them unsuitable for use in very cold (or hot) environments. He discussed the use of flash memory instead of disks and explained that most flash memory expects to be used with FAT file systems. Kamp provided an interesting tip: that just tying a knot at the end of a cable run will act as a coil, similar to the ferrite coils you often see attached to device cables. Both the knot and the ferrite coils are supposed to damp down voltage surges by using inductance. Kamp also mentioned the use of nanoBSD, a stripped-down FreeBSD, for use in firewall appliances.

I next listened to Warner Losh talk about FreeBSD ARM running an ATMEL System on Chip (SoC). In the embedded-systems market, vendors will take a processor design like the ARM and pack as many devices onto the chip as possible. Losh mentioned that the ATMEL SoC he was working with reused the pins on the chips for multiple purposes: A set of pins could be used for serial, USB, or Ethernet, depending on what you wanted to do. His particular application was to build a small system that provided accurate timing signals (something Kamp is also inter-

ested in), and you can find his slides at http://people.freebsd.org/~imp/bsdcan2006/text0.html.

I've done some work with other embedded-systems programmers, and I asked Losh about using the JTAG interface, a serial interface used for debugging integrated circuits. Losh said he did not use it.

FREEBSD

In the next talk I attended, Robert Watson explained how FreeBSD works. I had often wondered about the various versions (currently 4, 5, and 6, with 7 to come sometime in the foreseeable future). The three active versions are all maintained, but new developments will only appear in the newest track, 7. The *stable* version means just that (it is considered stable and safe to use), whereas *current* means the latest build, ready for testing (terminology that I think takes some getting used to). In practice, you want a stable version, unless you want to find bugs and interesting corner cases. As a new release gets close to release, it goes through *code slush* (no new features), *code freeze*, *beta*, *release candidate,* and then finally the release itself.

The BSD License is one thing that distinguishes BSD from Linux, as it encourages commercial use. Another is how FreeBSD advances. There is a core group of nine developers, elected by committers, who can commit code into the CVS trees. The core group manages but does not control direction. What gets developed leads the way, although the core group does make the final decision about what gets committed. The core group also handles disputes and lends authority when things need to get done.

There are 346 committers, 185 of whom are ports committers (people who commit changes to the applications supported on FreeBSD). Currently, there are over 13,500 ports supported, an average of 73 ports per committer. There are also over 1500 ports maintainers, people who help with ports but cannot commit changes. Throughout the conference, there were mentions of parts of the kernel, drivers, and ports in need of a volunteer to maintain them. Consistent work on a project leads to becoming a committer, and this also includes work on documentation.

FreeBSD is also backed by the FreeBSD Foundation, which provides real support, including help with legal and licensing issues, hardware donations, and funds for the FreeBSD developers conference and some travel fees.

WIPS

Robert Watson introduced the work-in-progress talks, starting with Poul-Henning Kamp's Varnish project. Varnish is a Web-caching server designed for the needs of newspapers and other sources that change Web content quickly. Varnish can be loaded at any time, can have multiple configs (VCL language) loaded at once, can include conditionals and forwarding, can do this with clusters, has a command-line interface, and can pull up statistics on objects, logging to shared memory segment, and a logdaemon processes this shared memory in the format of your choice (Apache, custom, and real-time views). Varnish appears under a BSD license and is sponsored by Norway's biggest paper, Verdens Gang (www.vg.no).

Murray Stokely (murray@freebsd.org) spoke about the Summer of Code (SoC), which provides $4500 grants to students to work on coding projects. Google spent $2 million last summer (2005). Of the 400 applications for BSD, 20 were funded, representing half of last summer's SoC grants. At the time of the conference it was already too late to apply for 2006 SoC, but there is always a need for mentors. Stokely mentioned several related URLs: code.google.com, netbsd.org/contrib/projects.html, and freebsd.org/projects/summerofcode.html.

Jan Schaumann (jschauma@netbsd.org) discussed a medley of SoC projects, including bpg (licensed PGP for BSD), Apple's HFS+ support, NDIS driver support ported from FreeBSD, memory-based file systems (being ported *to* FreeBSD), userspace filesystem support, journaling for FFS, automated regression testing framework, zeroconf, and improving mbufs. Google will select the top 20 of 80 BSD-related SoC applications this year.

Christian S.J. Peron discussed his work with TrustedBSD, including auditing work targeted at the Common Criteria CC/CAPP. Kernel and OS parts are relatively mature, but lots of key userspace programs are not there yet. Lots of programs do not understand audit, so you can insert NO_AUDIT into make.conf to prevent it from being included in packages you build in the ports system. Login, ssh, logout, and other things were changed to support audit context. Peron worked from OpenBSM library to get the bits into place. He created a general-purpose audit submission mechanism to avoid code replication and made use of tokens for event type, subject, optional text token, and return value. Su needs to submit an audit trail; he wants to do this with sudo too (this might appear in the Apple branch). CAPP also requires user/group manipulation

recording, audit records for system halts or shutdowns, and audit records of daemons that execute code in the context of other users, such as cron, at, and sendmail.

Csaba Henk (csaba.henk@creo.hu) talked about a userspace file system interface named FUSE. The current FUSE patches into the VFS, then requires a context switch to go to the userspace file system and another context switch to return, which is expensive. He used FUSE Linux (by Miklos Szeredi: fuse.sf.net), an easy-to-use API that helped in the port to FreeBSD (as part of SoC). The kernel component is being written from scratch to keep it in the BSD license, and because the Linux VFS structure is too different for a straight port. Henk mentioned that DragonflyBSD has a message-passing design and needs to use a userspace file system.

Colin Percival wrote FreeBSD Update, which led to bsdiff, which is used by Apple and Firefox for distributing updates. He is now rewriting Update, because the build code is very complicated currently. In the new version you select which version of code is to be updated, and Update will use source code and a set of patches rather than a CVS code tree. FreeBSD Update will become officially supported. People who build special versions on other platforms will be able to use either the older or the newer (2.0) version.

Warner Losh (imp@freebsd.org) talked about the state of processor ports for embedded-system support. With FreeBSD/ARM, the Core functionality is good, and work-in-progress includes the p4 tree, Cirrus Logic EP-9302, XScale improvements, AT91RM9200 Boot loader, and IIC (I2C) and MMC infrastructures. The Intel world is difficult to support because of nondisclosure issues. Losh said that we need to improve cross-building support and GDB protocol support.

Kip Macy also talked about work on supporting the new T1 Ultrasparc architecture, where there can be up to eight cores on a single chip, each supporting four threads. An advantage of the Sparc design is that it provides about the same performance as dual-Xeon processor systems at less than half the TDP (Thermal Design Power). But there is still work needed in getting the port stable, as the current sun4v version will panic with pmap races under 90% load. Volunteers are welcome, said Macy.

BOFS

After the last WiP, Langille announced that there would be a Postgres conference in Toronto in July (conference.postgressql.org). Then he listed six BoFs for the evening, including a Google BoF, an open cryptographic BoF, and a BSD certification BoF. I attended the latter and learned that the FreeBSD organization has spent a lot of time on, and gone a long way toward, producing a certification test for BSD sysadmins.

SATURDAY MORNING

As befits a conference where many attendees stay up late hackin or drinking, the first sessions began at 10 a.m. Saturday morning. I listened to Reyk Floeter (reyk@vantronix.net) discuss some features of OpenBSD support for wireless. He first explained that any connection could be deauthenticated, even when WEP was in use, as that portion of a wireless frame is not encrypted or authenticated. He went on to explain that OpenBSD hostapd had been used at What the Hack in Europe last summer, and he showed maps of the coverage. The OpenBSD hostapd apparently supports wireless roaming, which is interesting.

Floeter also talked about using the trunk interface in OpenBSD in failover mode. The trunk interface allows several network interfaces to be combined so that they perform as one, but with both higher throughput and failover. He then discussed improvements in OpenBSD IPSec implementation and support for the IEEE WLAN access protocol, 80211i/WPA2. He ended the talk with a plea to "Stop the Blob," a reference to using binary code blocks provided by vendors unwilling to provide documentation or support for open source projects. Stop-the-Blob T-shirts were on sale in the lobby of the conference building.

FREE BEER

I stayed in the same classroom so that I could hear Greg Lehey (grog@freebsd.org) talk about free beer. I thought this was a reference to BSD licensing (as in Poul-Henning Kamp's "free as in free beer" license) but this talk was really about brewing beer, using FreeBSD running on an old 386DX box, a set of relays, two temperature sensors, a light bulb, and a refrigerator to control the temperature of the future beer's wort while it is fermenting. So I not only learned a lot about beer history and brewing, I also learned about using FreeBSD as a control system. I could have also listened to Poul-Henning Kamp talk about his own FreeBSD control projects, which include many remote applications in environments with extreme weather, a talk that occurred in another track.

After lunch (at a pub), John-Mark Garner (jmg@freebsd.org) gave a presentation about writing device drivers in FreeBSD. Of course, you can't learn how to write device drivers in an hour, but Garner did a good job of providing an overview of the framework available. I finally learned what has happened to minor devices (made unnecessary because of devfs). Garner also talked about softc, a newer, more efficient framework for writing device drivers, about how resources (memory, IRQs, and ports) should be handled, and about bus probing and DMA.

Chris Buescher and Scott Ullrich discussed the various firewalls available in the BSD environment. BSD suffers from an embarrassment of riches here, and the presenters created a large chart, which you can find in their slides at pfsense.org/bsdcan/, to compare the features of the three firewall families, ipfw, ipfilter, and pf. They went on to explain the m0n0wall project, a version of FreeBSD stripped down for use in firewall appliances and controlled completely through the use of PHP over a Web interface (m0n0wall.ch). They then described their own project, pfSense (pfsense.org), where they forked their own version from m0n0wall because they wanted to create a firewall install that was much more featureful. Whereas m0n0wall is based on FreeBSD 4.1 for its faster network performance, pfSense uses FreeBSD 6.1, which has wireless networking support that FreeBSD 4.1 lacks. PfSense includes networking tools, such as tcpdump and HSFC traffic-shaping, borrowed from OpenBSD, and uses pf for firewall support, giving it the ability to do OS fingerprint–based blocking.

Dan Langille ended the conference by giving away books and T-shirts. Some books were given to people chosen randomly [by using random() to assign numbers to all attendees, then sorting] and for various feats. Someone won a book by spending over six hours trying to get through Canadian customs. (There was actually someone who had spent longer, but he had already won a book.)