Operating Freely

Chuck Cranor

If you are interested in working with the internals of an operating system now is a good time to get involved. In the past, operating system source code has been highly protected and thus hard for the average student to get a hold of. However, as a result of the work of Linus Torvalds, and the release of 4.4BSD-lite, there are now three major free Unix-like operating systems to choose from. In this article I will briefly describe these three operating systems and then detail some of the work I have done to contribute to the NetBSD effort.

The Linux Operating System

Linux is a free operating system which runs on standard IBM Intel 386/486/Pentium processors. The Linux kernel was written from scratch by Linus Torvalds and is now maintained by Linus and a host of people on the internet. While there is one Linux kernel, there are many different Linux distributions to choose from. The distributions differ in which applications are provided. One popular distribution, Slackware, comes with applications such as X windows, GNU Emacs, and LaTeX. This is very nice if you have no interest in obtaining, compiling, and installing these programs from source. Discussion on Linux takes place on the comp.os.linux.* USENET newsgroups. For more information, check out the Linux documentation on ftp://tsx-11.mit.edu/pub/linux/docs.

Berkeley Software Distributions Unix (BSD Unix)

The other major free operating systems have a common ancestor: BSD Unix from University of California, Berkeley (UCB). From 1979 to 1993 the UCB group coordinated and made significant contributions to the effort to add new features to Unix. Except for groups internal to Bell Labs and AT&T, the UCB group has had the greatest impact on the development of Unix. Features such as virtual memory management, the socket based TCP/IP protocol stack, reliable signals, job control, and the fast filesystem were added to Unix by researchers working with the Berkeley group. Today, most commercial versions of Unix have various bits of BSD Unix code spread throughout them.

One problem with early versions of BSD Unix was that parts of the source code were proprietary to Bell Labs and/or AT&T and could not be freely distributed. UCB responded to this problem by releasing the parts of the BSD source code that were contributed by Berkeley. After some legal wrangling, this has resulted in the final release of BSD Unix from UCB: 4.4BSD-lite. 4.4-lite is a freely available source code release of the non-proprietary parts of the BSD OS. It should be noted that 4.4-lite does not contain the full source code of the kernel -- there are still parts missing.

Before 4.4BSD was released, and while the legal status of BSD's source code was being resolved, a programmer named Bill Jolitz ported 4.3BSD-Net/2 to an IBM PC type machine with the Intel 80386 processor. Jolitz's BSD operating system, dubbed ``386BSD,'' consisted of freely available code from BSD-Net/2 with the missing proprietary bits filled in from scratch by Jolitz. Jolitz made 386BSD source code available to the network and many people started using it.

However, after 386BSD version 0.1 was released, Jolitz dropped off the network for personal reasons. This left the free BSD community in chaos -- there was no one leader to take Jolitz's place. Eventually the dust settled and the people interested in building a free BSD OS divided into two ``teams'' for the development of two OSs based on 386BSD: FreeBSD and NetBSD. While code is often shared between these two groups, each group is focused on providing OS support for different types of users and/or platforms.

FreeBSD

The FreeBSD team chose to focus their efforts on improving support for the i386/i486/Pentium architecture. Released in November of 1994, the latest version of FreeBSD is FreeBSD 2.0. While earlier versions of FreeBSD were based on Jolitz's port of 4.3-Net/2, the new version is fully based on a fresh port of 4.4BSD-lite. The FreeBSD team has enhanced 4.4-lite by adding features such as new device drivers for a wide range of PC devices, shared libraries, and improved virtual memory handling. The FreeBSD team has also implemented a ``package'' mechanism for installing and removing pre-compiled programs such as GNU Emacs. This makes life easier for users who have no interest in obtaining, configuring, and compiling applications. FreeBSD is often discussed on comp.os.386bsd.* and in the FreeBSD mailing lists (send an e-mail to majordomo@freebsd.org with the string "help" in the body for more information). It should be noted that there is currently a proposal out to create a set of comp.unix.bsd.freebsd.* newsgroups. FreeBSD can be obtained on CD-ROM or via the internet (ftp://ftp.freebsd.org/pub/FreeBSD).

NetBSD

The other freely available BSD-based OS is NetBSD. While the FreeBSD team chose to stick with the i386 processors, the NetBSD team has ported their OS to many different platforms. NetBSD version 1.0 was released in November 1994. NetBSD 1.0's kernel is 4.4BSD-lite based, and its applications are a

mix of 4.4-lite and 4.3-Net/2 applications. NetBSD 1.0 runs on nine different platforms: the IBM PC and clones with ix86 (x > 2) processors; the HP300, Macintosh, Amiga, Sun3, and da30 with Motorola 68k processors; the Sun4c with a Sparc processor; the DECstation with a Pmax processor; the PC532 with a NS 32523 processor; and the DEC VAX with a DEC VAX-11/750 processor. However, NetBSD 1.0 is not yet fully stable on some of these platforms. Like FreeBSD, NetBSD is often discussed on comp.os.386bsd.* (there is a currently a proposal in the works to create a set of comp.unix.bsd.netbsd.* newsgroups). NetBSD also has mailing lists which can be reached by sending a help message to majordomo@netbsd.org.

Working Freely: A Case Study

No article on the free Unix systems would be complete without a first hand description of what is involved with developing and contributing code to a free Unix system. I will describe my work with one free OS, NetBSD.

First, let me describe the organization of the NetBSD team. The ``leaders'' of the NetBSD team are called the ``core'' group. There are four members of the NetBSD core, who set the general direction of the OS. For each platform to which the OS is being ported there is a ``port master'' who coordinates the port. The rest of the team consists of the users who run NetBSD and contribute bug reports, fixes, and new code.

Communication between various members of the NetBSD team is important for maintaining a working system and minimizing frustration. While there are USENET newsgroups related to NetBSD, most of the real technical discussion on NetBSD takes place on the NetBSD mailing lists. There are lists that relate to the OS as a whole, and lists for most ports. There is also a list to which all formally submitted bug reports go, and a list that gets log messages when ever a source file is changed. Anyone can subscribe to these lists in order to get the most up-to-date information on the status of NetBSD.

Porting NetBSD/sparc

Since I do not have a PC, I was unable to get involved with the free OSs until the NetBSD/sparc port was released. The NetBSD/sparc port was done by Theo de Raadt based on code contributed by Chris Torek (who had ported 4.4BSD to the Sparc). There are four main types of Sparc machines: sun4, sun4c, sun4m, and sun4d. While each of these types of machines have the same instruction set, they differ in machine-specific implementation. Each type of Sparc machine has its own version of the kernel. A sun4 machine cannot run a sun4c kernel. The initial port of NetBSD/sparc ran only on the sun4c machines such as the Sparc1, Sparc1+, Sparc2, and Sparc IPX.

Last year my research group received a donation of several sun4 computers (SUN-4/300s). We wanted to run NetBSD/sparc on some of them, but NetBSD did not run on sun4s. At this point, most people would have given up and run SunOS, but I had the NetBSD source code and an attitude, so I decided to port NetBSD/sparc to the 4/300.

I spent about three months working part-time on the 4/300 port; most of this time was spent working on the boot sequence and device drivers. I started by debugging the sun4c boot sequence under the 4/300. The 4/300's boot PROM is totally different than the sun4c one. The sun4c boot PROM, ``openprom,'' is complex -- it has more features than you can shake a stick at! For example, the sun4c has routines to probe and detect devices built into the PROM, while the 4/300 requires these routines to be implemented in the OS (by me!).

The 4/300 has a somewhat different memory layout and virtual memory system than the sun4c. The 4/300 has its physical memory laid out contiguously, while there are gaps in the sun4c's physical memory space. The 4/300 has a write-back cache memory where as the sun4c has a write-through cache. In addition, the 4/300 has a memory page size of 8192 bytes, while the sun4c has a memory page size of 4096 bytes -- so the virtual memory management system had to be adjusted. The sun4 also has a different way of handling virtual memory faults than the sun4c. I had to write some assembly language code to address some of these differences.

Although the two machines have the same hardware interface to the serial line, the sun4 serial line can't handle data as fast as the sun4c. In fact, if you try and push the 4/300's serial line too fast it crashes the machine with stray interrupts. This usually happened to me in the middle of a debugging print statement, making my life difficult for a couple of weeks. Finally I solved the problem by taking a cue from the NetBSD sun3 port and putting some delays in the device driver to prevent it from sending data before the chip is ready.

Another problem which really had me pulling out my hair for a while was a memory fault I would get after the system went multiuser. I tried everything to debug it! I put in print statements. I used the debugger to disassemble parts of the kernel. I even used the PROM's monitor to check the machine code which was faulting. No matter what I did, everything checked out okay: the software looked fine. Finally, I booted SunOS on my test machine and started running stuff on it. It was then I discovered that SunOS would crash too! It turned out to be a hardware problem: one of the memory modules on my system would fail when placed under a heavy load.

In order to get NetBSD/sparc working on the 4/300 I had to address all these problems (plus a few more). After finishing the port, I generated a set of patches which incorporated 4/300 support into NetBSD-current and mailed them off to the Theo de Raadt (the NetBSD/sparc port master). He took my changes, fixed them up a bit and incorporated them into the main NetBSD source tree where they are now available for general use by anyone with a Sun-4/300.

The NetBSD source tree resides on a special machine which is the ``home'' of the project. This machine is called sun-lamp.cs.berkeley.edu (now also known as ftp.netbsd.org). The master copy of the NetBSD source code is stored on this machine. Only the NetBSD core and port masters have have full accounts on this machine (so they can updated the code). There are two ways for general users to access this machine: anonymous FTP and SUP (Software Update Protocol). Archive files of NetBSD sources and

binaries can be ftp'd from sun-lamp's anonymous ftp area.

The NetBSD source code is controlled with a Concurrent Versions System (CVS). CVS keeps track of all changes made to the source code and allows several versions of the code to exist at once. For example, the CVS system contains both the 1.0 release of NetBSD and the current working sources of NetBSD. Unlike commercial source code releases, the very latest version of the source code for free Unix systems is available to anyone. This version is tagged ``-current,'' as in NetBSD-current. NetBSD-current is distributed using a program called SUP. When you run SUP, it contacts sun-lamp or another NetBSD SUP server and determines what source files have changed since you last ran SUP. It then downloads those files from the archive, possibly over-writing the old version.

Conclusion

In this article I've described the three major freely available Unix-like operating systems: Linux, FreeBSD, and NetBSD. I then provided some details on how the NetBSD team operates and how I contributed to the NetBSD/sparc port. If you are interested in operating systems and have a computer to spare, I encourage you to give one or more of these systems a try. While doing kernel work is often difficult and frustrating, it is a good feeling to be contributing to something that other people can freely use to work or have fun with.