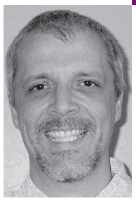
TOM HAYNES

OpenSolaris: the model



Tom Haynes is an NFS developer for Sun Microsystems, Inc., and is interested in the cost differences between open source and commercial offerings. He is exploring those costs by using OpenSolaris to design a NAS appliance.

tdh@sun.com

I FEEL LIKE THE NEW CHAIRMAN OF the hair loss club—I liked the product so much I went out and bought the company. Only I didn't buy the company, I simply joined it. The company I am talking about is Sun Microsystems, Inc., and the product is OpenSolaris. The premise is simple: Sun opens up its source vault and sucks more users into its web. Sun has a long development cycle between releases, and interested parties could always download the Solaris Express bits to play with new features. When I was at Network Appliance, Inc., we would do interoperability testing of NFSv4 based on the beta program. We saw the exact same binaries that any other customer of Sun could download. This binary availability was very crucial to the successful cross-deployment of a new protocol. After the release of Solaris 10, Sun decided to release the source code to the majority of the code base at the same time that the binaries, the release called Nevada, were made available. The parts not made available under the CDDL, or Common Development and Distribution License, were those sections that were already entangled under prior copyrights.

You can go to http://www.opensolaris.org to see what all the excitement is all about. There are already multiple distributors: for example, SchilliX (http://schillix.berlios.de) or Nexenta (http://www.gnusolaris.org)—think of these as early-day Debian or Slackware. There are development efforts underway to extend technology already found in Solaris 10, e.g., the BrandZ effort to extend zones (see "Solaris 10 Containers," by Peter Baer Galvin, in the October 2005 issue of *;login:*) to non-native operating systems—first up is Linux. And of course there is the recent release of ZFS in the Nevada Build 27 (or b27, as it is commonly called). ZFS is a radical new file system which has been under development at Sun for the last five years.

The two major draws of OpenSolaris are the commitment to quality and the early access to cutting-edge technology. It is easy to argue that GNU, Linux, and the *BSD efforts all provide the bleeding edge of technology. But the real cost can be in the quality of the built-in supportability of software installed at client sites. Note that I do not mean in the quality of open source products but, rather, in the quality of the support infrastructure in a data center.

A common scenario I have seen is a large data center with a heavy commitment to Linux-based compute servers running a very old kernel, say, a base RedHat 7.3 system with a 2.4.9 kernel. Either the company decided to roll that version out because it was the newest when they upgraded from a 2.2-based kernel, or they bought support from a third party. Regardless of how the decision was made, a further complication is that either a modification was made to the kernel source (the best case is that it was patched up), or the customer's application is dependent on that particular Linux kernel. And, finally, the company no longer has any support for the kernel—perhaps the contract ran out, or someone in management thought that free software was, well, free, and no budget was allocated for maintenance.

I know that if you contact Trond Myklebust, the Linux NFS client maintainer, for support, he will try to help you—no matter if you are a first-year student pounding away on an old hand-me-down laptop or the CIO of a company with a 5,000-node application farm. Depending on the problem, that student might get more help; the maintainer is a volunteer and prioritizes his time accordingly. If you stumble on a major bug he believes will impact the majority of Linux installations, he is going to give you attention. But if you have an interoperability problem with another vendor's product, one he may not have access to, then he is going to give you a fishing rod and teach you how to fish.

This approach is the QA model employed by most open source developers. They simply do not have the time, funds, equipment, or desire to test everything under the sun. So, instead, they provide new features in branches for the brave. These adventurers get bleeding-edge technology and the satisfaction of being able to contribute by finding bugs.

Sun plans to make money from OpenSolaris by being a service provider, and the biggest differentiations are quality and support. Sun already has infrastructure (personnel, equipment, tests) to do interoperability testing. The internal developers still have the same commitment to delivering bug-free software. They also have an organization dedicated to analyzing customer-found issues and providing fixes to customers.

Clearly, the interesting questions about Sun and OpenSolaris are concerns over how Sun and the open software model will interact. For example, if Sun is selling service and that same first-year student finds an issue in SchilliX, what level of support will he get? Or say he not only finds the bug, he fixes it and now wants the fix put back into the Solaris code. How can Sun juggle that need versus the need of the CIO paying an annual support contract for her enterprise data center?

The trick for Sun is that the CIO is going through professional services and the student is going through volunteer services. At the end of the day, they might get help from the same individual, but that depends on the commitment of the developer to the open source movement. Sun has asked its employees to help out with OpenSolaris, but it has not mandated that they do so—it is freedom of choice. And there are not only Sun employees helping out on the project.

Any individual asking for help on the OpenSolaris discussion forums, including that CIO, can expect the same level of support. It might just be more like that fishing rod analogy than some people are wanting and the response time might be in days instead of minutes. And that CIO might even find the student responding.

The other question I posed was how an individual outside of Sun gets fixes put back into the tree. In one approach, the individual or distributor maintains their own source base and does not even try to give back to the community. This model is akin to the way many startups in Silicon Valley operate—they take a FreeBSD release and tack on their IP. Perhaps they feed back general bugs (or even contribute scaled-down versions of their product), but they normally integrate from changes made at the source.

The second approach is for the outside individual to find a sponsor within Sun to champion their change. The sponsor arranges for a bug to be filed, code to be reviewed, and the fix put back into Solaris. Interestingly, the "outside individual" might be a Sun employee. For example, although I work in the NFS team, at night I might want to work on porting OpenSolaris to the UltraSPARC 1 platform. I might get it working and then look for a sponsor—perhaps in the kernel team.

The example also shows that by opening up its source, Sun has to make commitments which seem to run counter to its planning process. The UltraSPARC 1 was supported well into the late releases of Solaris Express for Solaris 10. But as a business, Sun decided to retire support for the system—the EOL was actually for the 32-bit SPARC kernel, but as there were outstanding issues with the UltraSPARC I chips in 64-bit mode, it was retired as well. As an individual, I could decide to backport OpenSolaris to this platform.

Sun has also pledged that it will provide ethical support to their employees who want to contribute to OpenSolaris. While Sun does employ full-time workers to develop OpenSolaris, for the most part such development is completely voluntary. At other companies, I've had to sign an NDA which precluded me from contributing to open source projects that could provide an advantage to competitors. At times, I was asked by the Linux NFS client maintainer to please not even look at that source code. He didn't want to risk contaminating it, under a new licensing model being considered by Linus Torvalds.

But I feel free to contribute to OpenSolaris, not only in NFS but in other modules. I know that if I want a break from my day job, I can still contribute, even if indirectly, to my company. I even know that if I do resurrect the UltraSPARC 1, I am likely to make someone smile in appreciation of the effort.

If I pull off those rose-colored glasses you might think I am wearing, I can see that Sun has taken a large risk. This plan could easily backfire on the company. Consider, for example, ZFS, the new file system. Sun has made it available in source form before it shipped in a commercial product. Instead of joining Sun, I could have gathered some venture capital and started shipping low-end NAS boxes in direct competition. My contribution would have been the business plan, not the cool technology. Also, if Sun is filing any patents on ZFS, it has to do so much earlier than normal (i.e., the technology has been publicly introduced).

Sun is betting the farm on differentiating its product offerings, not through the technology but, rather, through the support and service it can provide once that product is installed at a customer site. One nightmare they will have to contend with is that a customer may no longer be running a Sun kit and may not have bought an AMD-based Ultra 20. Instead, they may have taken their Linux farm with a hodgepodge of x86-based systems and loaded up either a stock Nevada b30 or SchilliX 0.4.

When Sun controlled the allowable hardware, it effectively was managing the service it needed to provide, though admittedly it has always been possible to add third-party hardware even to their proprietary systems. They have also been shipping the x86 version of Solaris for quite some time. But for a long time, the x86 product looked unsupported. I saw a couple of trade articles announcing the end of the product.

By opening up the vault and committing heavily to the x86 line, they have exposed themselves to the same nightmare of device driver management that other vendors and open source distributions have had to handle.

Sun already has a support model for someone running a Nevada b30 system—they accept bug reports and you can find employees interacting on the OpenSolaris discussions. With an OpenSolaris distribution, support will probably be the same except for a support contract that entails the migration of the boxes to the latest and greatest Solaris. If the concern is the availability of a certain new feature, Sun does backport some technology from Nevada into Solaris 10.

The neat thing about OpenSolaris is that anyone can contribute. Besides testing new technologies, you can see how a commercial product is built. The scripts used to build OpenSolaris are the same ones used to build Nevada, the next commercial release of Solaris. You can also dig into different releases of the source code to try to get an idea of how the underlying technology is changing. A key point to remember is that you are in essence viewing a beta release candidate—from build to build you might find a unique bug. And if you do, make sure to file it!