Looking to the Future

BY JOHN BALDWIN

FreeBSD's 13.0 release delivers new features to users and refines the workflow for new contributions. FreeBSD contributors have been busy fixing bugs and adding new features since 12.0's release in December of 2018. In addition, FreeBSD developers have refined their vision to focus on FreeBSD's future users. An abbreviated list of some of the changes in 13.0 is given below. A more detailed list can be found in the release notes.

Shifting Tools

Not all of the changes in the FreeBSD Project over the last two years have taken the form of patches. Some of the largest changes have been made in the tools used to contribute to FreeBSD. The first major change is that FreeBSD has switched from Subversion to Git for storing source code, documentation, and ports. Git is widely used in the software industry and is more familiar to new contributors than Subversion. Git's distributed nature also more easily facilitates contributions from individuals who are not committers. FreeBSD had been providing Git mirrors of the Subversion repositories for several years, and many developers had used Git to manage in-progress patches. The Git mirrors have now become the official repositories and changes are now pushed directly to Git instead of Subversion. FreeBSD 13.0 is the first release whose sources are only available via Git rather than Subversion. The first phase of this process focused on adapting the Project's existing workflows and tools (such as Phabricator and Bugzilla) to work with the new Git repositories. The next phase will allow us to explore additional tools such as pre-commit testing and continuous integration.

Not all of the changes in the FreeBSD Project over the last two years have taken the form of patches.

A second major change is the adoption of AsciiDoc for the source format of FreeBSD's documentation and website. FreeBSD's documentation consists of three broad groups: manual pages, books and articles (such as the FreeBSD Handbook), and the project website. Books and articles were previously written in an SGML markup language called DocBook, and the website was written directly in HTML. While DocBook is very expressive and provides support for many features such as callouts, footnotes, and indices, it is a verbose format. Since the original design of DocBook, lighter-weight markup languages such as MarkDown have become prevalent. As-

ciiDoc is a lighter-weight markup language similar to MarkDown that retains the expressiveness of DocBook. The FreeBSD documentation team recently converted all of the books, articles, and website to AsciiDoc. This provides a simpler and easier to read format that will make it easier for new folks to contribute documentation.

Manual pages continue to be written in a dialect of troff known as mdoc.

Planning for Future Systems

One of the changes in FreeBSD's focus over the past few years has been to emphasize support for systems that users will be using in the future over support for older systems used by a decreasing number of users. This does not mean abandoning support for all systems which are not brand new. However, as some older systems recede further into history, the benefit of maintaining support for those systems in the tree no longer justifies the cost. FreeBSD 13.0 removes support for older 32-bit ARM systems as well as the UltraSparc platform. Device drivers for some older devices that are no longer commonly used have also been removed. In addition, in recognition of the dominance of 64-bit x86 systems, the 32-bit x86 architecture has been demoted to a Tier 2 architecture.

Streamlining our focus has allowed the Project to devote more resources to other architectures and drivers whose use will grow in the future. ARM, PowerPC, and RISC-V have all received substantial changes including support for new drivers and improved performance. The 64-bit x86 architecture now supports Hygon Dhyana processors as well as support for 57-bit user virtual addresses on newer Intel processors. Finally, all of the architectures in 13.0 are supported by the in-tree LLVM toolchain including the clang compiler and lld linker. By no longer maintaining compatibility with legacy GPLv2 toolchains, FreeBSD can now adopt modern language and toolchain features. (For more on this, see Ed Maste's article, FBSD 13 Tool Chain also in this issue.) Along with changes to replace or retire other GPL-licensed components in the base system, this also means that FreeBSD 13.0 ships with only two GPL utilities and one LGPL library in the base system.

> A second major change is the adoption of AsciiDoc for the source format of FreeBSD's documentation and website.

OpenZFS

FreeBSD has included ZFS in the base system for over a decade. FreeBSD's ZFS support was originally ported from OpenSolaris and for a long time tracked the ZFS support in the public OpenSolaris (later illumos) repository. Over the past few years, active development of ZFS has moved out of the illumos repository into the cross-platform OpenZFS project. FreeBSD 13.0 replaces the illumos-derived ZFS support with code from OpenZFS. This brings in several new features including encrypted datasets and ZSTD compression. (See also Allan Jude's article, Zstandard Compression in OpenZFS also in this issue)

Networking

13.0 includes several networking changes. Kernel TLS offload enables a single web server to transmit hundreds of gigabits of HTTPS traffic (see John Baldwin's article, TLS Offload in the Kernel). The NFS client and server now support NFSv4.2. This includes a new system call to per-

mit optimized server-side file copies. The NFS client and server also support NFS over TLS via kernel TLS offload.

Security

FreeBSD 13.0's kernel contains several improvements to the kernel cryptography framework used for geli(8), ZFS, IPsec, and kernel TLS. 64-bit ARM systems will now make use of accelerated software cryptography for the AES-GCM and AES-XTS ciphers out of the box via the armv8crypto(4) driver. Both 32-bit and 64-bit x86 systems also include support for accelerated software cryptography in the default kernel via the aesni(4) driver.

Boot Loader

The per-kernel boot loader includes several changes. First, when booting from UEFI, the default install now installs the full boot loader to the EFI system partition. Previously, a small boot loader in the EFI system partition was used to locate and boot the full boot loader. This two-stage process proved unwieldy and the firmware now loads the full boot loader directly. Secondly, on x86 systems the boot loader now uses a graphical display on the video console both when booting via UEFI and when booting via BIOS. This graphical console is then handed off to the kernel for use as a framebuffer by the vt(4) driver.

Virtualization

FreeBSD 13.0 includes several virtualization improvements both as a guest and a host. The VirtIO suite of device drivers in the kernel now support version 1 of the VirtIO specification. This improves compatibility with hypervisors, emulators, and simulation models which provide VirtIO devices. The bhyve(8) hypervisor includes several changes including improved VNC support (including compatibility with the built-in "Screen Sharing" VNC client in macOS), VirtIO 9p filesystem sharing, and initial support for virtual machine snapshots.

We hope you enjoy FreeBSD 13.0. Join us for the next adventure developing FreeBSD 14!

Conclusion

FreeBSD 13.0 is the product of contributions from the Project's community over the past two years. Thank you to everyone who has contributed to this release by testing snapshots, reporting bugs, submitting patches, working with users on social media, and countless other tasks. We hope you enjoy FreeBSD 13.0. Join us for the next adventure developing FreeBSD 14!

JOHN BALDWIN is a systems software developer. He has directly committed changes to the FreeBSD operating system for 20 years across various parts of the kernel (including x86 platform support, SMP, various device drivers, and the virtual memory subsystem) and userspace programs. In addition to writing code, John has served on the FreeBSD core and release engineering teams. He has also contributed to the GDB debugger and LLVM. John lives in Concord, California, with his wife, Kimberly, and three children: Janelle, Evan, and Bella.