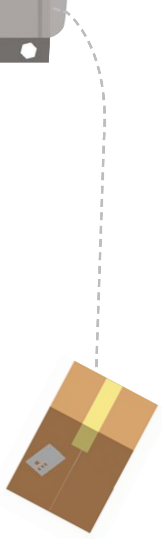
SEE TEXT ONLY

FreeBSD Toolchain

As with other BSD distributions, FreeBSD has the concept of a base system, an integrated kernel, and core userland, which are developed, tested, and released together by a common team. The userland includes the C language runtime, the build toolchain, basic system utilities, and some third-party libraries and applications. The toolchain includes the compiler and linker, which translate source code into executable objects, and related utilities for inspecting or modifying those objects.

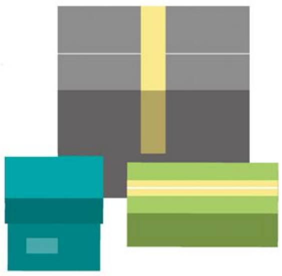


In addition to this base system, there's a third-party software ecosystem managed through the FreeBSD ports tree and binary packages. Over 26,000 third-party software packages exist in the ports tree.

For most of its history FreeBSD relied on the full GNU toolchain suite, including the GNU Compiler Collection (GCC) compiler and GNU binutils. They served the FreeBSD Project well from 1993 until 2007, when the GNU project migrated to releasing its software under the General Public License, version 3 (GPLv3). In contrast to version 2 of the license, GPLv3 places a number of new restrictions on software users, and FreeBSD developers and users found these objectionable. As a result, the GNU toolchain in FreeBSD was not updated to a new upstream version and quickly became outdated.

Beginning shortly after that transition, a number of developers in the FreeBSD Project started updating components of the FreeBSD toolchain to modern, copy-free alternatives. Some aspects of the toolchain are architecture-dependent, and in general the tools described below apply to FreeBSD's Tier-1 architectures.

BY ED MASTE



Compiler

The LLVM project is a collection of modular and reusable components used to build compilers and other toolchain utilities. It began as a research project at the University of Illinois and has since grown to become a production-quality tool set used in proprietary and open-source software and academic settings. Clang is the C and C++ compiler included in the LLVM suite.

We've experimented with Clang in FreeBSD since 2009, starting with a snapshot from the upstream project's Subversion repository. When first added, it was made available as an option to facilitate testing and development, but it was not enabled as the default compiler for some time. When FreeBSD 10 released in January 2014, it included Clang as the system compiler (that is, installed as `/usr/bin/cc`) for the 32- and 64-bit Intel x86 architectures. It was later made the default for the 32-bit FreeBSD ARM port. The 64-bit ARM port began with Clang as the system compiler.

Clang includes some support for the MIPS and PowerPC architectures, but is not yet capable of replacing the system compiler there; those architectures continue to use GCC.

FreeBSD 11 includes Clang 3.8.0, and work is underway in FreeBSD-CURRENT to update to Clang 3.9.0.

Linker

The linker takes as input individual object files produced by a compiler or assembler and links them into an executable binary or library. To date we've used the GNU linker in FreeBSD, most recently updated to version 2.17.50 in February 2011. This linker lacks link-time optimizations and support for certain debugging features.

GNU ld in the base system also lacks support for AArch64 (arm64) and RISC-V, two CPU architectures recently added to FreeBSD. Building FreeBSD for these currently requires the linker be installed from the ports tree or as a package, in which case it will be used automatically.

The LLVM family of projects includes a linker: LLD. It is a high-performance linker with support for ELF, COFF, and Mach-O. Where possible, it maintains command-line and functional compatibility with existing linkers such as GNU ld, but LLD's authors are not constrained by strict compatibility where it hampers performance or desired functionality.

Work began on LLD's ELF support in July 2015 and has progressed very quickly; it currently self-hosts on several of LLVM's supported architectures and includes almost all of the functionality required to build the FreeBSD base system.

In comparison with GNU ld, in FreeBSD LLD will provide AArch64 and eventually RISC-V support, full-program Link-Time Optimization (LTO), support for new Application Binary Interfaces (ABIs), other linker optimizations, debugging features, and much faster link times.

As with early experiments with Clang, LLD will first be added to FreeBSD as an option and will not be installed as the system linker (`/usr/bin/ld`). It will be available by adding **`-fuse-ld=lld`** to the compiler's command line arguments (for example, via the **`CFLAGS`** variable).

We've included LLD 3.9 alongside the Clang 3.9 update and expect it to be available in FreeBSD-CURRENT in October or November 2016. It is expected to be added to FreeBSD 11.1 as well.

Binary Utilities

The toolchain includes a number of small binary utilities for inspecting or modifying object files, binaries, and libraries. These are tools that report information about an object, binary, or library, or transform an object's contents or format. These tools were historically provided by GNU binutils.

The ELF Tool Chain Project maintains BSD-licensed implementations of essential compilation tools and libraries for working with ELF object files and binaries, and DWARF debugging information. It began as part of the FreeBSD Project, but became a standalone project in order to facilitate collaboration with the wider open-source community.

For FreeBSD 11 we have migrated to ELF Tool Chain's implementation of **`addr2line`**, **`c++filt`**, **`objcopy`**, **`nm`**, **`readelf`**, **`size`**, **`strip`**, and **`strings`**. The **`libelf`** and **`libdwarf`** libraries also come from ELF Tool Chain.

ELF Tool Chain also includes a version of FreeBSD's **`ar`**, **`brandelf`**, and **`elfdump`** utilities. We will migrate to those in the future if they gain compelling new features or improvements.

Runtime Libraries

A number of libraries are required to provide runtime support for the toolchain. The compiler-rt library comes from the LLVM project and includes several components. Low-level target-specific hooks required by GCC or Clang's code generation are known as "builtins." These are routines that perform operations that the compiler will not emit directly into the output object code.

Compiler-rt includes sanitizer runtimes that provide Clang's runtime support for identifying erroneous software behaviour. AddressSanitizer (asan) detects out-of-bounds access to heap, stack, and globals, use-after-free and related errors, and double and invalid free. It is available by adding `-fsanitize=address` to the compiler's command line.

The UndefinedBehaviourSanitizer (ubsan) catches undefined behaviour; that is, an operation for which the language does not have specified behaviour. Examples include overflowing integer operations (such as addition and bit shifting), division by zero, and using uninitialized variables. The `-fsanitize=undefined` command line argument enables ubsan.

Clang includes other sanitizers in the upstream repository, including ThreadSanitizer for detecting data races in multithreaded applications and MemorySanitizer for detecting uninitialized reads. Work is underway to make these available with the system compiler in a later FreeBSD version.

For C++ runtime support we use a combination of PathScale's libcxxrt for low-level Application Binary Interface (ABI) support. This includes Run-Time Type Information (RTTI), exception handling, and thread-safe initializers. The C++ standard library is LLVM's libc++.

Debugger

A replacement debugger also comes from the LLVM project: LLDB. As with the rest of LLVM, LLDB is built as a set of reusable components, and builds on other parts of LLVM and Clang. LLDB has access to a full Clang compiler, which it uses as the expression parser. This provides high fidelity in interpreting the user's expressions: LLDB is capable of parsing any expression valid in the original source code being debugged.

LLDB is included in FreeBSD 11 for the 32- and 64-bit ARM architectures, and 64-bit Intel (amd64).

In addition to LLDB in the base system, the GNU GDB debugger is available by building from the ports tree or installing the package. It has been updated and includes improved threading support and kernel debugging. GDB 7.11.1 is available with a simple `pkg install gdb` command.

DTrace

FreeBSD 11 includes support for using Compressed Type Format (CTF) data in userland. This means that userland Function Boundary Tracing (FBT) probes can have typed arguments when the corresponding executable or libraries are compiled with CTF. The FreeBSD base system build infrastructure includes support for compiling with CTF with a setting in the `/etc/src.conf` file.

FreeBSD 11 also adds support for Statically Defined Tracing (STD) probes in userland. A `.d` file containing probe definitions can be included in the list of source files, and the build infrastructure will automatically generate a header containing probe macros that can be referenced in source files.

Future Work

Work is underway on a number of additional toolchain components which will become available in FreeBSD in the future. LLVM provides a code coverage tool llvm-cov, which can operate in a mode similar to GNU gcov, and can operate with Clang's instrumentation-based profiling. LLVM also provides an OpenMP runtime (libomp), an Application Programming Interface (API) for shared memory multiprocessing.

With LLD as the linker, two GNU binutils tools remain: the assembler (as) and objdump. There is currently no candidate to replace as. However, it is actually not required to build the FreeBSD base system on Tier-1 architectures: Clang's integrated assembler is used to build assembly source files. GNU as could simply be removed, or a GNU as-compatible driver could be built from the integrated assembler. Objdump is also not required by the base system build and could be removed. LLVM provides an llvm-objdump that is currently limited in functionality, but will likely be a viable replacement in due course.

Ongoing efforts in the FreeBSD community and in the upstream Clang/LLVM project are attempting to bring the Clang, LLD, and LLDB suite to the lower-tier FreeBSD architectures that are still using GCC today. ●

ED MASTE manages project development for the FreeBSD Foundation and works in an engineering support role with the University of Cambridge Computer Laboratory. He is also a member of the elected FreeBSD Core Team. Aside from FreeBSD and ELF Tool Chain, he is a contributor to a number of other open-source projects, including LLVM, QEMU, and Open vSwitch. He lives in Kitchener, Canada, with his wife, Anna, and sons, Pieter and Daniel.

THROUGH DECEMBER 2016 BY DRU LAVIGNE

Events Calendar

The following BSD-related conferences will take place in November and December 2016. More information about these events, as well as local user group meetings, can be found at **www.bsdevents.org**.

Meet BSD California • Nov 10–12 • Berkeley, CA

https://meetbsd.com • The 5th biennial MeetBSD conference will be held at UC Berkeley. This conference provides a mix of formal presentations, unconference activities, and opportunities to network with other BSD users and developers. The BSDA certification exam will be available at this event. Registration is required to attend this conference.

LISA 16 • Dec 4–9 • Boston, MA

https://www.usenix.org/conference/lisa16/ • The 30th Large Installation System Administration conference will be held at the Sheraton Boston. There will be a FreeBSD booth in the expo area. Registration is required to attend the conference and a nominal fee is required to attend the expo.

Thank you!

The FreesBSD Foundation would like to acknowledge the following companies for their continued support of the Project. Because of generous donations such as these we are able to continue moving the Project forward.

FreeBSD FOUNDATION

Are you a fan of FreeBSD? Help us give back to the Project and donate today!
freebsdfoundation.org/donate/

Iridium

NetApp

Gold

NETFLIX

Silver

accelerationsystems

vmware Tarsnap

Please check out the full list of generous community investors at freebsdfoundation.org/donors