

---

6.11. Binutils-2.18 Linux From Scratch - Version 6.4 Chapter 6. Installing Basic System Software Prev Re-adjusting the Toolchain Next GMP-4.2.4 Up Home

6.11. Binutils-2.18 The Binutils package contains a linker, an assembler, and other tools for handling object files. Approximate build time: 1.7 SBU testsuite included Required disk space: 186 MB testsuite included 6.11.1. Installation of Binutils Verify that the PTYs are working properly inside the chroot environment. Check that everything is set up correctly by performing a simple test: `expect -c "spawn ls"` If the following message shows up, the chroot environment is not set up for proper PTY operation: The system has no more ptys. Ask your system administrator to create more. This issue needs to be resolved before running the test suites for Binutils and GCC. Binutils does not recognize versions of Texinfo newer than 4.9. Fix this issue by applying the following patch: `patch -Np1 -i ../binutils-2.18-configure-1.patch` Apply the following patch to prevent some failures when running the the test suite: `patch -Np1 -i ../binutils-2.18-GCC43-1.patch` Suppress the installation of an outdated standards.info file as a newer one is installed later on in the Autoconf instructions: `rm -fv etc/standards.info sed -i.bak '/^INFO/s/standards.info //' etc/Makefile.in` The Binutils documentation recommends building Binutils outside of the source directory in a dedicated build directory: `mkdir -v ../binutils-build cd ../binutils-build` Prepare Binutils for compilation: `../binutils-2.18/configure --prefix=/usr \ --enable-shared` Compile the package: `make tooldir=/usr` The meaning of the make parameter: `tooldir=/usr` Normally, the `tooldir` (the directory where the executables will ultimately be located) is set to `$(exec_prefix)/$(target_alias)`. For example, i686 machines would expand that to `/usr/i686-pc-linux-gnu`. Because this is a custom system, this target-specific directory in `/usr` is not required. `$(exec_prefix)/$(target_alias)` would be used if the system was used to cross-compile (for example, compiling a package on an Intel machine that generates code that can be executed on PowerPC machines). Important The test suite for Binutils in this section is considered critical. Do not skip it under any circumstances. Test the results: `make check` Install the package: `make tooldir=/usr install` Install the libiberty header file that is needed by some packages: `cp -v ../binutils-2.18/include/libiberty.h /usr/include`

6.11.2. Contents of Binutils Installed programs: `addr2line`, `ar`, `as`, `c++filt`, `gprof`, `ld`, `nm`, `objcopy`, `objdump`, `ranlib`, `readelf`, `size`, `strings`, and `strip` Installed libraries: `libiberty.a`, `libbfd.{a,so}`, and `libopcodes.{a,so}`

Short Descriptions `addr2line` Translates program addresses to file names and line numbers; given an address and the name of an executable, it uses the debugging information in the executable to determine which source file and line number are associated with the address `ar` Creates, modifies, and extracts from archives `as` An assembler that assembles the output of `gcc` into object files `c++filt` Used by the linker to de-mangle C++ and Java symbols and to keep overloaded functions from clashing `gprof` Displays call graph profile data `ld` A linker that combines a number of object and archive files into a single file, relocating their data and tying up symbol references `nm` Lists the symbols occurring in a given object file `objcopy` Translates one type of object file into another `objdump` Displays information about the given object file, with options controlling the particular information to display; the information shown is useful to programmers who are working on the compilation tools `ranlib` Generates an index of the contents of an archive and stores it in the archive; the index lists all of the symbols defined by archive members that are relocatable object files `readelf` Displays information about ELF type binaries `size` Lists the section sizes and the total size for the given object files `strings` Outputs, for each given file, the sequences of printable characters that are of at least the specified length (defaulting to four); for object files, it prints, by default, only the strings from the initializing and loading sections while for other types of files, it scans the entire file `strip` Discards symbols from object files `libiberty` Contains routines used by various GNU programs, including `getopt`, `obstack`, `strerror`, `strtol`, and `strtoul` `libbfd` The Binary File Descriptor library `libopcodes` A library for dealing with opcodes—the “readable text” versions of instructions for the processor; it is used for building utilities like `objdump`. Prev Re-adjusting the Toolchain Next GMP-4.2.4 Up Home

---