Building Ice for C++ on Solaris

This page describes the Ice source distribution, including information about compiler requirements, third-party dependencies, and instructions for building and testing the distribution. If you prefer, you can download a binary package that contains pre-compiled libraries, executables, and everything else necessary to build Ice applications on Solaris.

On this page:

C++ Build Requirements for Solaris

Ice for C++ was extensively tested using the operating systems and compiler versions listed on our platforms page.

Third-party Libraries

Ice has dependencies on a number of third-party libraries:

- Berkeley DB 5.3
- expat 2.0
- OpenSSL 0.9.8 or later
- bzip2 1.0
- mcpp 2.7.2 (with patches)

All these libraries, with the exception of Berkeley DB and mcpp, are available from pkg.oracle.com.

Build Tools

GNU Make 3.81 is required to build Ice on Solaris. It can be installed using pkg.oracle.com (package name: gnu-make).

Compiling and Testing Ice for C++ on Solaris

Extract the Ice archive in any directory you like (for example, in your home directory):

```
$ gtar xvfz Ice-3.5.1.tar.gz
```

Change the working directory to `Ice-3.5.1/cpp`:

```
$ cd Ice-3.5.1/cpp
```

Edit `config/Make.rules` to establish your build configuration. The comments in the file provide more information. Pay particular attention to the variables that define the locations of the third-party libraries.

Now you're ready to build Ice:

```
$ gmake
```

This will build the Ice core libraries, services, tests and examples.

Python is required to run the test suite. After a successful build, you can run the tests as follows:

```
$ gmake test
```

This command is equivalent to:

```
$ python allTests.py
```

If everything worked out, you should see lots of "ok" messages. In case of a failure, the tests abort with "failed".

If you want to try out any of the demos, make sure to update your `PATH` environment variable to add the `bin` directory, and your `LD_LIBRARY_PATH` environment variable to add the `lib` directory. For 64-bit builds, add `lib` to `LD_LIBRARY_PATH_64`:

```
$ export PATH=`pwd`/bin:$PATH
$ export LD_LIBRARY_PATH=`pwd`/lib:$LD_LIBRARY_PATH
$ export LD_LIBRARY_PATH_64=`pwd`/lib:$LD_LIBRARY_PATH_64
```

64-bit Source Builds on Solaris

To build Ice in 64-bit mode, you need to do the following:

- Obtain or build all the third-party dependencies, and put the 64-bit libraries in the appropriate `lib` subdirectory (`lib/64` on SPARC, `lib/amd64` on x86).
- Set the environment variable `LP64` to yes, as shown below:

  ```
  $ export LP64=yes
  ```
- Build and test as described above.

Installing a C++ Source Build on Solaris

Simply run `gmake install`. This will install Ice in the directory specified by the `prefix` variable in `config/Make.rules`.

After installation, make sure that the *`prefix`*`/bin` directory is in your `PATH`.

If you choose to not embed a runpath into executables at build time (see your build settings in `cpp/config/Make.rules`) or did not create a symbolic link from the runpath directory to the installation directory, you also need to add the *`prefix`*`/lib` directory to your `LD_LIBRARY_PATH`.

When compiling Ice programs, you must pass the location of the *`prefix`*`/include` directory to the compiler with the `-I` option, and the location of the *`prefix`*`/lib` directory with the `-L` option.

If you built in 64-bit mode, the libraries are installed in *`prefix`*`/lib/64` or *`prefix`*`/lib/amd64`, depending on your system's architecture. Executables are installed in *`prefix`*`/bin/sparcv9` or *`prefix`*`/bin/amd64`. No other changes are necessary.