Main /

Building PTLib on Unix

This page describes how to build PTLib from revision 2.4.2 (Wolf) or later. For information on building earlier releases of PTLib, see this page

Contents

1. Introduction

PTLib is built from source using `make` and `configure` in the same way as most other Unix programs.

If you want to install PTLib into a shared location accessible by all users of the system (normally /usr/local), and you have root access, then you can use the default configure and install procedure Introduction described here.

If you want do not want to install PTLib into a shared location, because you do not have root access or because you want to have more than one version of PTLib on the same host, then follow this procedure.

2. Prerequisites

You will need the following:

- gcc 4.7 or later. Earlier versions of gcc may work, but are not maintained.

- A copy of the PTLib source archive from the SourceForge download page or from Subversion. This is extracted into a directory referrred to as *srcdir*

For Ubuntu etc, there are the following optional packages:

```
apt install g++ git make autoconf libpcap-dev libexpat-dev libssl1.0-dev libsasl2-dev libldap-dev \
            unixodbc-dev liblua5.3-dev libv8-dev libncurses-dev libsdl2-dev libavformat-dev libswscale-dev \
            libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev
```

For CentOS, Fedora etc, use:

```
yum install g++ git make autoconf libpcap-devel expat-devel openssl-devel cyrus-sasl-devel openldap-devel \
            unixODBC-devel lua-devel v8-devel ncurses-devel SDL2-devel libavformat-devel libswscale-devel \
            gstreamer1.0-devel gstreamer-plugins-base1.0-devel
```

3. Default build and install

By default, PTLib will install into the /usr/local tree. In order for other packages (including Opal) to locate PTLib, the directory /usr/local/lib/pkgconfig must be added to the search path for pkg-config. This is usually done by setting the PKG_CONFIG_PATH environment variable.

It is also necessary to set the LD_LIBRARY_PATH environment variable so that programs can find the PTLib libraries at link or run time.

If you are using plugins, the PTLIBPLUGINDIR environment variable must be set to the location of the installed plugins. By default, this is /usr/local/lib/ptlib-*minor*.*major*.*patch*, where *minor*.*major*.*patch* is the version of PTlib being used.

For bash/sh

```
  export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig
  export LD_LIBRARY_PATH=/usr/local/lib
  export PTLIBPLUGINDIR=/usr/local/lib/ptlib-minor.major.patch
```

For tcsh/csh

```
  setenv PKG_CONFIG_PATH /usr/local/lib/pkgconfig
  setenv LD_LIBRARY_PATH /usr/local/lib
  setenv PTLIBPLUGINDIR /usr/local/lib/ptlib-minor.major.patch
```

Use the following command to build and install the release version (optimised) as a shared library

```
  cd srcdir
  ./configure
  make
  sudo make install
```

Use the following command to build and install the release version (optimised) as a static library

```
  cd srcdir
  ./configure --disable-shared
  make
  sudo make install
```

Use the following command to build and install the debug version (includes debugging symbols) as a shared library

```
  cd srcdir
  ./configure --enable-debug
  make debug
  sudo make install
```

Use the following command to build and install the debug version (includes debugging symbols) as a static library

```
  cd srcdir
  ./configure --enable-debug --disable-shared
  make debug
  sudo make install
```

Multiple versions can be built in the same directory tree and installed separately

4. Build and use without install

Set the PTLIBDIR environment variable to point to *srcdir* as follows. This allows other libraries that use PTLib to find the code without using pkg-config.

IMPORTANT: you must set the environment variable ***before*** executing the configure script or the system will get very confused.

It is also necessary to set the LD_LIBRARY_PATH environment variable so that programs can find the PTLib libraries at link or run time. The correct directory name depends on the host architecture:

```
   For Linux x86 32 bit     ${PTLIBDIR}/lib_linux_x86
   For Linux x86 64 bit     ${PTLIBDIR}/lib_linux_x86_64
```

If you are using plugins, the PTLIBPLUGINDIR environment variable must be set to the location of the compiled plugins. This will be the same directory as used above for the LD_LIBRARY_PATH

For bash/sh

```
   export PTLIBDIR=srcdir
   export LD_LIBRARY_PATH=${PTLIBDIR}/lib_linux_x86
   export PTLIBPLUGINDIR=${PTLIBDIR}/lib_linux_x86
```

For tcsh/csh

```
   setenv PTLIBDIR srcdir
   setenv LD_LIBRARY_PATH ${PTLIBDIR}/lib_linux_x86
```

```
    setenv PTLIBPLUGINDIR ${PTLIBDIR}/lib_linux_x86
```

Once this is done, configure and build the appropriate version of the code using the same commands as described in section 3 above, but omit the *sudo make install*

Note that it may be necessary to add /usr/local/lib to LD_LIBRARY_PATH and set PKG_CONFIG_PATH to /usr/local/lib/pkgconfig in order for other packages to be detected correctly.

5. Editing configure.ac, and running aclocal and autoconf

If you are developer, it may be necessary to modify the configure.ac file and regenerate the configure script using autoconf.

The following message may appear

```
  configure.ac:5: error: possibly undefined macro: dnl
        If this token and others are legitimate, please use m4_pattern_allow.
        See the Autoconf documentation.
```

If this occurs, then you need to regnerate the aclocal.m4 file by running the following command:

```
  aclocal
```

PmWiki can't process your request

Cannot acquire lockfile

We are sorry for any inconvenience.

More information

Return to http://wiki.opalvoip.org/index.php