Cygwin/XFree86 Contributor's Guide

Harold L Hunt, II

Cygwin/XFree86 Contributor's Guide

by Harold L Hunt, II

Revision History

Revision 0.0.6 2001-07-17 Revised by: huntharo@msu.edu

Revision 0.0.5 2001-07-12 Revised by: huntharo@msu.edu

Revision 0.0.4 2001-07-02 Revised by: huntharo@msu.edu
Contributer's was misspelled, changed to Contributor's.
Revision 0.0.3 2001-06-13 Revised by: huntharo@msu.edu

Revision 0.0.2 2001-06-12 Revised by: huntharo@msu.edu

Revision 0.0.1 2001-06-11 Revised by: huntharo@msu.edu
Initial content for the Programming chapter.
Revision 0.0.0 2001-05-22 Revised by: huntharo@msu.edu
Initial release.

Table of Contents

1. Overview **1**

2. Programming **1**

Overview 1

Native Compiling 1

Obtaining the Source Code 1

Compiling the Source Code 3

Compiling Overview 4

Standard Build 5

Debug Build 7

Installing a local build 9

Keeping your source code tree updated 11

Update the entire source code tree 11

Update a single file or directory 12

Cross Compiling 13

Obtaining binutils and gcc Source 13

Obtaining Cygwin Headers and Libs 14

Building binutils and gcc 15

Obtaining the Source Code 17

Building Cygwin/XFree86 18

3. Documentation **21**

4. Web Site Maintenance **22**

Bibliography **23**

Glossary **24**

A. GNU Free Documentation License **26**

0. PREAMBLE 26

1. APPLICABILITY AND DEFINITIONS 26

2. VERBATIM COPYING 27

3. COPYING IN QUANTITY 27

4. MODIFICATIONS 28

5. COMBINING DOCUMENTS 30

6. COLLECTIONS OF DOCUMENTS 30

7. AGGREGATION WITH INDEPENDENT WORKS 30

8. TRANSLATION 31

9. TERMINATION 31

10. FUTURE REVISIONS OF THIS LICENSE 31

How to use this License for your documents 32

Chapter 1. Overview

The Cygwin/XFree86 project can use your help! We will do everything we can to make experienced contributers productive as soon as possible. We also want to make it as easy as possible for new contributers to make Cygwin/XFree86 their first open source project.

Cygwin/XFree86 is part of the vast number of open source/free software programs that provide compatibility with closed source/commercial software products. Cygwin/XFree86 enables the coexistence of closed software and open software during the period of transition from an almost completely closed software market to an almost completely open software market.

Join in the excitement of opening your Windows machine to the X Window System.

We need programmers, documentation writers, and website maintainers.

Chapter 2. Programming

Overview

This chapter provides a consolidated overview of all of the information needed to begin making source code contributions to Cygwin/XFree86. Creating a source code contribution for Cygwin/XFree86 requires an amazingly small amount of information; however, prior to this document that tiny amount of information was difficult to obtain, as it was scattered across several documents and source code files. New programmers with no open source project experience, as well as programming gurus, will be able to make sourc code contributions to Cygwin/XFree86 after reading this chapter. Programming gurus are great; our intention is to create more of them.

You will want to start downloading the XFree86 source code tree immediately after reading the section called *Obtaining the Source Code*, if you have an active network connection at your disposal, as downloading the source code tree can take anywhere from 10 minutes to 10 hours, depending upon the speed of your connection. You will find it advantageous to have a source code tree as you read the later sections.

Native Compiling

Obtaining the Source Code

Cygwin/XFree86 source code is contained in, and distributed with, the XFree86 source code tree. Read-only CVS access to the XFree86 source tree (http://xfree86.org/cvs/) is also available from the XFree86 project.

The XFree86 CVS tree is not always buildable on Cygwin, as the XFree86 CVS tree is modified frequently, and sometimes those modifications cause building on Cygwin to fail; however, there are a few developers monitoring the build state of the XFree86 CVS tree on Cygwin, and they quickly fix build problems as they appear. It is unlikely that you will experience problems building XFree86 on Cygwin; in the event that you do experience build problems, you may wish to drop a note to cygwin-xfree@cygwin.com to let the developers there know that the tree is not building correctly.

Periodically the Cygwin/XFree86 project releases a snapshot of the XFree86 source code that is known to compile. It is perfectly acceptable, though rarely necessary, to do your independent development work from a stable source code snapshot, rather than from the XFree86 CVS tree. Developing from a stable source code snapshots minimizes the scope of problems that you may encounter, which can greatly ease debugging of new features and other code modifications.

It is highly recommended to get the XFree86 *CVS* tree by using the CVS *pserver* method. CVS pserver is the easiest CVS method to use for anonymous access to public CVS trees, as you need only type in a well-known password to download the CVS tree. CVS pserver is rarely used to control checkins to CVS trees, as the passwords are stored and sent as plain text; thus, pserver is extremely insecure. However, security does not matter for a public read-only CVS tree.

Follow these steps to checkout the XFree86 CVS tree:

1. Open a Cygwin **bash** shell by double clicking the Cygwin icon on your desktop, or by selecting Start, then Programs, followed by Cygnus Solutions, and finally Cygwin Bash Shell. You will see output similar to the following:

   ```
   Harold@MyWindowsHost ~
   $
   ```

2. Create a directory in which to store your XFree86 CVS tree and your builds; `~/x-devel` is recommended:

   ```
   Harold@MyWindowsHost ~
   $ mkdir x-devel

   Harold@MyWindowsHost ~
   $
   ```

3. Change the current directory to your new XFree86 development directory:

   ```
   Harold@MyWindowsHost ~
   $ cd x-devel

   Harold@MyWindowsHost ~/x-devel
   $
   ```

4. Set the CVSROOT environment variable to point to the XFree86 CVS repository:

   ```
   Harold@MyWindowsHost ~/x-devel
   $ CVSROOT=:pserver:anoncvs@anoncvs.xfree86.org:/cvs

   Harold@MyWindowsHost ~/x-devel
   $ export CVSROOT

   Harold@MyWindowsHost ~/x-devel
   $
   ```

5. Login to the CVS server, using password “anoncvs”:

```
Harold@MyWindowsHost ~/x-devel
$ cvs login
(Logging in to anoncvs@anoncvs.xfree86.org)
CVS password:

Harold@MyWindowsHost ~/x-devel
$
```

6. Checkout the `xc/` directory from the XFree86 CVS tree:

> **Tip:** The *-zn* parameter specifies the compression level to use, from 1 to 9, with 9 being maximum compression.

> **Note:** As of 2001-06-12, a checked out XFree86 CVS source code tree contains 16,199 files in 3,021 folders, which is 242 MB of data, but requires 285 MB of storage space on a file system using 4 KB allocation units.

```
Harold@MyWindowsHost ~/x-devel
$ cvs -z4 checkout xc
cvs server: Updating xc

...

Harold@MyWindowsHost ~/x-devel
$
```

You may preserve a logfile for the checkout session by instead using the following command:

```
Harold@MyWindowsHost ~/x-devel
$ cvs -z4 checkout xc > xc-cvs-checkout.log 2>&1

Harold@MyWindowsHost ~/x-devel
$
```

Compiling the Source Code

Compiling Cygwin/XFree86 doesn't have to be hard, although the XFree86 source code tree contains over 250 MB of data. There are a few simple techniques that make building the source code, keeping the source code up to date, and keeping the source code organized much easier.

Compiling Overview

Compiling the XFree86 source code tree is a lot easier when you keep your builds in directories seperate from the source code directory. Keeping the source code and builds in seperate directories allows you to have many builds for different configurations, allows you to easily delete a build, and keeps the source code tree clean and manageable.

A small utility, `lndir.exe`, is needed to keep your builds directories sepereate from your source directory; `lndir.exe` works just like the standard **ln** on UNIX, but `lndir.exe` creates links recursively for all files and directories in the specified directory. The `lndir.exe` utility is included with the XFree86 source code tree; but the catch is that you need to build the tree before you get `lndir.exe`. `lndir.exe` has been compiled and is available at http://www.msu.edu/~huntharo/xwin/lndir.exe.bz2 (8 kB). Download the file, saving it to your Cygwin root directory (e.g. `c:\cygwin`), then follow the simple instructions below to install the utility:

1. Launch a Cygwin **bash** prompt. You should see a screen similar to the following:

```
Harold@MyWindowsHost ~
$
```

2. Change to your Cygwin root directory:

```
Harold@MyWindowsHost ~
$ cd /

Harold@MyWindowsHost /
$
```

3. Uncompress lndir.exe.bz2:

```
Harold@MyWindowsHost /
$ bunzip2 lndir.exe.bz2

Harold@MyWindowsHost /
$
```

4. Copy lndir.exe to `/bin`:

```
Harold@MyWindowsHost /
$ cp lndir.exe /bin

Harold@MyWindowsHost /
$
```

5. Verify that **lndir** is working:

```
Harold@MyWindowsHost /
$ lndir
usage: lndir.exe [-silent] [-ignorelinks] fromdir [todir]

Harold@MyWindowsHost /
$
```

6. The **lndir** utility is now installed.

Standard Build

Follow these steps to create a standard, non-debug, build:

1. Change the current directory to your XFree86 development directory:

```
Harold@MyWindowsHost ~
$ cd x-devel

Harold@MyWindowsHost ~/x-devel
$
```

2. Create a directory to house your builds, `~/x-devel/build` is recommended:

```
Harold@MyWindowsHost ~/x-devel
$ mkdir build

Harold@MyWindowsHost ~/x-devel/build
$
```

3. Change the current directory to your build directory:

```
Harold@MyWindowsHost ~/x-devel
```

```
$ cd build

Harold@MyWindowsHost ~/x-devel/build
$
```

4. Create a directory for your standard build, `std` is recommended:

```
Harold@MyWindowsHost ~/x-devel/build
$ mkdir std

Harold@MyWindowsHost ~/x-devel/build
$
```

5. Change the current directory to your standard build directory:

```
Harold@MyWindowsHost ~/x-devel/build
$ cd std

Harold@MyWindowsHost ~/x-devel/build/std
$
```

6. Create symlinks to your source tree, using **lndir**, in your standard build directory:

> **Note:** As of 2001-06-12, creating symlinks to the source tree creates 11,664 files in 1,510 folders, which is only 2.24 MB of data, but requires 45.5 MB of storage space on a file system using 4 KB allocation units.

```
Harold@MyWindowsHost ~/x-devel/build/std
$ lndir ../../xc/
../../xc/config:
../../../xc/config/cf:

...

Harold@MyWindowsHost ~/x-devel/build/std
$
```

7. Run a standard build of the entire tree, which takes between 30 minutes and 5 hours, saving the output of the build commands to `World.log`:

> **Note:** As of 2001-06-12, a standard build of the entire tree requires 255.5 MB of storage space on a file system using 4 KB allocation units; that is in addition to the 45.5 MB of previously generated symlinks.
>
> As a benchmark, a standard build runs for 71 minutes on a machine with a 1.2 GHz Atlhon, 256 MB DDR RAM, and a 7200 RPM ATA/100 HD.

```
Harold@MyWindowsHost ~/x-devel/build/std
$ make World > World.log 2>&1

Harold@MyWindowsHost ~/x-devel/build/std
$
```

Debug Build

Follow these steps to create a build with debugging information:

1. Change the current directory to your XFree86 development directory:

   ```
   Harold@MyWindowsHost ~
   $ cd x-devel

   Harold@MyWindowsHost ~/x-devel
   $
   ```

2. If you have not already done so, create a directory to house your builds, `~/x-devel/build` is recommended:

   ```
   Harold@MyWindowsHost ~/x-devel
   $ mkdir build

   Harold@MyWindowsHost ~/x-devel/build
   $
   ```

3. Change the current directory to your build directory:

   ```
   Harold@MyWindowsHost ~/x-devel
   $ cd build

   Harold@MyWindowsHost ~/x-devel/build
   ```

```
$
```

4. Create a directory for your debug build, `debug` is recommended:

```
Harold@MyWindowsHost ~/x-devel/build
$ mkdir debug

Harold@MyWindowsHost ~/x-devel/build
$
```

5. Change the current directory to your debug build directory:

```
Harold@MyWindowsHost ~/x-devel/build
$ cd debug

Harold@MyWindowsHost ~/x-devel/build/debug
$
```

6. Create links to your source tree, using **lndir**, in your standard build directory:

> **Note:** As of 2001-06-12, creating symlinks to the source tree creates 11,664 files in 1,510 folders, which is only 2.24 MB of data, but requires 45.5 MB of storage space on a file system using 4 KB allocation units.

```
Harold@MyWindowsHost ~/x-devel/build/debug
$ lndir ../../xc/
../../xc/config:
../../../xc/config/cf:

...

Harold@MyWindowsHost ~/x-devel/build/debug
$
```

7. Run a debug build of the entire tree, which takes between 30 minutes and 5 hours, saving the output of the build commands to `World.log`:

> **Note:** As of 2001-06-12, a debug build of the entire tree requires 566.5 MB of storage space on a file system using 4 KB allocation units; that is in addition to the 45.5 MB of previously generated symlinks.

> As a benchmark, a debug build runs for 71 minutes on a machine with a 1.2 GHz Atlhon, 256 MB DDR RAM, and a 7200 RPM ATA/100 HD. You may have noticed that the standard build time and the debug build time are identical.

```
Harold@MyWindowsHost ~/x-devel/build/debug
$ ./config/util/makeg.sh World > World.log 2>&1

Harold@MyWindowsHost ~/x-devel/build/debug
$
```

Installing a local build

Installing a local build enables you to verify that a build of the entire source tree is operational. It is wise to verify the operation of full builds of the source tree from time to time, as full builds will occasionally be broken by changes that other developers are making to the XFree86 source code tree.

Installing a local build on top of an existing build is not a good idea, as this can mask problems that occured during the build process, or it can cause problems that are unrelated to the build process; either situation is undesireable. It is generally a good idea to move your old installation out of the way before installing a local build, and these instructions will assume that you desire to do so. Follow the instructions below to install a local build:

1. Move the `/etc/X11` directory to `/etc/X11_`*`build-prefix_date_time`*:

   ```
   Harold@MyWindowsHost ~
   $ mv /etc/X11 /etc/X11_build-prefix_date_time

   Harold@MyWindowsHost ~
   $
   ```

2. Move the `/usr/X11R6` diretory to `/usr/X11R6_`*`build-prefix_date_time`*:

   ```
   Harold@MyWindowsHost ~
   $ mv /usr/X11R6 /usr/X11R6_build-prefix_date_time

   Harold@MyWindowsHost ~
   $
   ```

3. Change the current directory to your desired XFree86 build directory:

```
Harold@MyWindowsHost ~
$ cd ~/x-devel/build/build-prefix

Harold@MyWindowsHost ~/x-devel/build/build-prefix
$
```

4. Make the **`install`** target, which installs binaries, fonts, libraries, and configuration files; in short, **`install`** installs everything except the **man** pages:

> **Note:** As of 2001-06-12, the **`install`** target copies 5,074 files in 83 folders into `/usr/X11R6`, requiring 89.2 MB of storage space for a standard build or 177 MB of storage space for a debug build, and 276 files in 39 folders into `/etc/X11`, requiring 2.57 MB of storage space. All stated storage requirements are for a file system using 4 KB allocation units.
>
> As a benchmark, **`install`** runs for 20 minutes on a machine with a 1.2 GHz Atlhon, 256 MB DDR RAM, and a 7200 RPM ATA/100 HD. Standard and debug installs both complete in the stated time.

```
Harold@MyWindowsHost ~/x-devel/build/build-prefix
$ make install > install.log 2>&1

Harold@MyWindowsHost ~/x-devel/build/build-prefix
$
```

5. Make the **`install.man`** target, which only installs the **man** pages:

> **Note:** As of 2001-06-12, the **`install.man`** target copies 541 files in 3 folders into `/usr/X11R6/man`, requiring 4.22 MB of storage space, and 544 files in 1 folder into `/usr/X11R6/lib/X11/doc`, requiring 4.76 MB of storage space. All stated storage requirements are for a file system using 4 KB allocation units.
>
> As a benchmark, **`install.man`** runs for 2 minutes on a machine with a 1.2 GHz Atlhon, 256 MB DDR RAM, and a 7200 RPM ATA/100 HD.

```
Harold@MyWindowsHost ~/x-devel/build/build-prefix
$ make install.man > install.man.log 2>&1

Harold@MyWindowsHost ~/x-devel/build/build-prefix
$
```

Keeping your source code tree updated

CVS makes keeping your source code tree up to date easy. You may update your entire source code tree at once, or you can update individual directories or files, if you so choose.

Update the entire source code tree

1. Change the current directory to your XFree86 development directory:

```
Harold@MyWindowsHost ~
$ cd x-devel

Harold@MyWindowsHost ~/x-devel
$
```

2. Change the current directory to the root of the XFree86 source code tree, `xc/`:

```
Harold@MyWindowsHost ~/x-devel
$ cd xc

Harold@MyWindowsHost ~/x-devel/xc
$
```

3. To update your entire XFree86 source code tree, run the following command:

   Tip: The *-zn* parameter specifies the compression level to use, from 1 to 9, with 9 being maximum compression.

   The *-d* parameter instructs **cvs** to rebuild the directory list, which causes new directories in the source code tree to be downloaded (new directories are skipped if you do not specify *-d*).

```
Harold@MyWindowsHost ~/x-devel/xc
$ cvs -z4 update -d

Harold@MyWindowsHost ~/x-devel/xc
$
```

Update a single file or directory

1. Change the current directory to your XFree86 development directory:

```
Harold@MyWindowsHost ~
$ cd x-devel

Harold@MyWindowsHost ~/x-devel
$
```

2. Change the current directory to the directory that contains the file you wish to update, or change the current directory to the directory that you wish to update:

```
Harold@MyWindowsHost ~/x-devel
$ cd xc/directory_to_update

Harold@MyWindowsHost ~/x-devel/xc/directory_to_update
$
```

3. To update a single file, or a set of specified files, run the following command:

> **Tip:** The *-zn* parameter specifies the compression level to use, from 1 to 9, with 9 being maximum compression.

```
Harold@MyWindowsHost ~/x-devel/xc/directory_to_update
$ cvs -z4 update filename_1 [filename_2 ...]

Harold@MyWindowsHost ~/x-devel/xc/directory_to_update
$
```

4. To update a single directory, and its subdirectories, run the following command:

> **Note:** The *-zn* parameter specifies the compression level to use, from 1 to 9, with 9 being maximum compression.
>
> The *-d* parameter instructs **cvs** to rebuild the directory list, which causes new directories in the source code tree to be downloaded (new directories are skipped if you do not specify *-d*).

```
Harold@MyWindowsHost ~/x-devel/xc/directory_to_update
```

```
$ cvs -z4 update -d

Harold@MyWindowsHost ~/x-devel/xc/directory_to_update
$
```

Cross Compiling

Obtaining binutils and gcc Source

binutils and gcc source code releases that are known to compile on Cygwin and distributed by the Cygwin project. Therefore, it is highly recommended that you obtain the binutils and gcc sources from the Cygwin mirror network (http://cygwin.com/mirrors.html).

Follow these steps to download Cygwin/XFree86 binaries:

1. Create a directory to store the binutils and gcc sources in, such as `/cygwin/src/`
2. Visit the Cygwin mirrors page (http://cygwin.com/mirrors.html) to find your closest mirror
3. The ftp url for your mirror site should take you to the `cygwin/` directory on the mirror
4. Download the following files from `cygwin/`, saving them to `/cygwin/src/` The compressed file size appears after each file in the list below.

> **Downloading with a Web Browser:** Some web browsers automatically decompress saved files when you use the left mouse button to follow the link to a file; bunzip2 will report, “Data integrity error when decompressing.”, when attempting to decompress a file that has been decompressed by your web browser. Prevent your files from being automatically decompressed by clicking the right mouse button on a file link and choosing a command such as Save Target As... or Save Link As... from the context sensitive menu. Better yet, download your files with a stand alone ftp client.

- `binutils/binutils-20010425-2-src.tar.gz` (9.1 MB; required, necessary to build gcc and Cygwin/XFree86)
- `gcc/gcc-2.95.3-5-src.tar.bz2` (8.1 MB; required, necessary to build Cygwin/XFree86)

Obtaining Cygwin Headers and Libs

The simplest method of building a cross compiler for Cygwin requires that you have the Cygwin headers and libraries available at the time of building the cross compiler. Cygwin headers and libraries are installed when Cygwin is installed, so the headers and libraries can be obtained from an existing Cygwin installation.

> **Don't simply copy the headers and libraries:** Some of the headers and libraries are symbolic links to other headers or libraries. Copying these files using a program that is not aware of Cygwin's symlink emulation will result in some of the header and library files being broken. The method described below will preserve the symbolic links used by the header and library files.

1. Launch your Cygwin environment, using either the icon on your Desktop, the icon in your Start Menu, or by running cygwin.bat from your Cygwin directory (e.g. `c:\cygwin`); you should see a window like the following:

   ```
   Harold@MyWindowsHost ~
   $
   ```

2. Change the current directory to Cygwin root directory:

   ```
   Harold@MyWindowsHost ~
   $ cd /

   Harold@MyWindowsHost /
   $
   ```

3. Create an archive of the contents of the `/lib` directory:

   ```
   Harold@MyWindowsHost /
   $ tar -czf cygwin-lib.tgz lib/

   Harold@MyWindowsHost /
   $
   ```

4. Change the current directory to the `usr` directory in your Cygwin root directory:

```
Harold@MyWindowsHost /
$ cd /usr

Harold@MyWindowsHost /usr
$
```

5. Create an archive of the contents of the `/usr/include` directory:

```
Harold@MyWindowsHost /usr
$ tar -czf cygwin-include.tgz include/

Harold@MyWindowsHost /usr
$
```

6. Transfer cygwin-lib.tgz and cygwin-include.tgz to your build host using any method that you have available (e.g. ftp, samba, diskette, etc.). Save the files in the `/cygwin/i686-pc-cygwin/` directory.
7. Open a shell on your cross compiling build host; you should see a window like the following:

```
[harold@MyCrossHost harold]$
```

8. Change the current directory to the `/cygwin/i686-pc-cygwin/` directory in your build host root directory:

```
[harold@MyCrossHost harold]$ cd /cygwin/i686-pc-cygwin/
[harold@MyCrossHost /cygwin/i686-pc-cygwin/]$
```

9. Extract the cygwin-lib.tgz and cygwin-include.tgz archives:

```
[harold@MyCrossHost /cygwin/i686-pc-cygwin/]$ tar -xzf
cygwin-lib.tgz
[harold@MyCrossHost /cygwin/i686-pc-cygwin/]$ tar -xzf
cygwin-include.tgz
```

Building binutils and gcc

1. Open a shell on your cross compiling build host; you should see a window like the following:

```
[harold@MyCrossHost harold]$
```

2. Change the current directory to the `/cygwin/src/` directory in your build host root directory:

```
[harold@MyCrossHost harold]$ cd /cygwin/src/
[harold@MyCrossHost src]$
```

3. Extract the binutils-20010425-2-src.tar.gz and gcc-2.95.3-5-src.tar.bz2 archives:

```
[harold@MyCrossHost src]$ tar -xzf
binutils-20010425-2-src.tar.gz
[harold@MyCrossHost src]$ bunzip2
gcc-2.95.3-5-src.tar.bz2
[harold@MyCrossHost src]$ tar -xf
gcc-2.95.3-5-src.tar
```

4. Change the current directory to the `/cygwin/src/binutils-20010425-2` directory:

```
[harold@MyCrossHost src]$ cd /cygwin/src/binutils-20010425-2
[harold@MyCrossHost binutils-20010425-2]$
```

5. Create a `build` directory and change the current directory to that directory:

```
[harold@MyCrossHost binutils-20010425-2]$ mkdir build
[harold@MyCrossHost binutils-20010425-2]$ cd build
[harold@MyCrossHost build]$
```

6. Configure binutils:

```
[harold@MyCrossHost build]$ ../configure
-prefix=/cygwin -exec-prefix=/cygwin
-target=i686-pc-cygwin -host=i686-pc-linux > configure.log 2>&1
[harold@MyCrossHost build]$
```

7. Build binutils:

```
[harold@MyCrossHost build]$ make all > all.log 2>&1
[harold@MyCrossHost build]$
```

8. Install binutils:

```
[harold@MyCrossHost build]$ make install > install.log 2>&1
[harold@MyCrossHost build]$
```

9. Modify the PATH environment variable to include the directories that the binutils executables were installed in:

```
[harold@MyCrossHost build]$
PATH=$PATH:/cygwin/bin:/cygwin/i686-pc-cygwin/bin
[harold@MyCrossHost build]$
```

10. Change the current directory to the `/cygwin/src/binutils-20010425-2` directory:

```
[harold@MyCrossHost src]$ cd /cygwin/src/gcc-2.95.3-5
[harold@MyCrossHost gcc-2.95.3-5]$
```

11. Create a `build` directory and change the current directory to that directory:

```
[harold@MyCrossHost gcc-2.95.3-5]$ mkdir build
[harold@MyCrossHost gcc-2.95.3-5]$ cd build
[harold@MyCrossHost build]$
```

12. Configure gcc:

```
[harold@MyCrossHost build]$ ../configure
-prefix=/cygwin -exec-prefix=/cygwin
-target=i686-pc-cygwin -host=i686-pc-linux
-enable-haifa > configure.log 2>&1
[harold@MyCrossHost build]$
```

13. Build gcc:

```
[harold@MyCrossHost build]$ make all > all.log 2>&1
[harold@MyCrossHost build]$
```

14. Install gcc:

```
[harold@MyCrossHost build]$ make install > install.log 2>&1
[harold@MyCrossHost build]$
```

Obtaining the Source Code

Obtaining the source code when cross compiling XFree86 is nearly identical to the process described in the section called *Obtaining the Source Code* of the Native Compiling section. The only divergence from the aformentioned instructions is that you will be using a **bash** shell on your cross compiling host, rather than on your native Cygwin host.

Building Cygwin/XFree86

Building the source code when cross compiling XFree86 is nearly identical to the process described in the section called *Compiling the Source Code* of the Native Compiling section. One divergence from the aformentioned instructions is that you will be using a **bash** shell on your cross compiling host, rather than on your native Cygwin host; other divergences follow.

1. Download host.def (http://www.msu.edu/~huntharo/xwin/host.def), and place the file in `xc/config/cf`.
2. Modify host.def/PostIncDir to reflect the actual version of gcc compiled, such as #define PostIncDir /cygwin/lib/gcc-lib/i686-pc-cygwin/2.95.3-5/include. The version number in the path will change depending on which version of gcc you built and installed.
3. **lndir** should generally already be installed if your build host already has the X Window System installed. In any case, you certainly won't want to install the build of **lndir** mentioned in the section called *Native Compiling* as that executable is for Cygwin only.
4. When building the entire tree, you must pass *`IMAKE_DEFINES`* and *`BOOTSTRAPCFLAGS`* to **make** to cause the build system to build for the target, Cygwin, platform:

   ```
   [harold@MyCrossHost std]$ make World
   BOOTSTRAPCFLAGS="-D__CYGWIN__ -Ulinux -DCrossCompiling=1"
   IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > World.log 2>&1
   [harold@MyCrossHost std]$
   ```

5. When rebuilding individual elements of the tree after doing a build of the entire tree, you must pass *`IMAKE_DEFINES`* to **make** to cause the build system to build for the target, Cygwin, platform:

   ```
   [harold@MyCrossHost Xserver]$ make World
   IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > World.log 2>&1
   [harold@MyCrossHost Xserver]$
   ```

6. When building a debug version of the entire tree, use **makeg**, which should generally already be installed, and pass *IMAKE_DEFINES* and *BOOTSTRAPCFLAGS* to **makeg** to cause the build system to build for the target, Cygwin, platform:

```
[harold@MyCrossHost debug]$ makeg World
BOOTSTRAPCFLAGS="-D__CYGWIN__ -Ulinux -DCrossCompiling=1"
IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > World.log 2>&1
[harold@MyCrossHost debug]$
```

7. When rebuilding debug versions of individual elements of the tree after doing a debug build of the entire tree, you must pass *IMAKE_DEFINES* to **makeg** to cause the build system to build for the target, Cygwin, platform:

```
[harold@MyCrossHost Xserver]$ makeg World
IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > World.log 2>&1
[harold@MyCrossHost Xserver]$
```

8. When installing a standard build of the entire tree, you must pass *DESTDIR* and *IMAKE_DEFINES* to **make** to install the target platform build into `/stagingdir` and to cause the build system to build a few remaining programs for the target, Cygwin, platform:

> **Tip:** Never run **make install install.man** on your host platform without the *DESTDIR* parameter, as that will cause the Cygwin build of XFree86 to be installed overtop of your local X Window System installation, which would completely destroy your host systems' X Window System installation.

```
[harold@MyCrossHost std]$ make install install.man
DESTDIR=/stagingdir
IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > install.log 2>&1
[harold@MyCrossHost std]$
```

9. When installing a debug build of the entire tree, you must pass *DESTDIR* and *IMAKE_DEFINES* to **makeg** to install the target platform build into `/stagingdir` and to cause the build system to build debug versions of a few remaining programs for the target, Cygwin, platform:

> **Tip:** Never run **makeg install install.man** on your host platform without the *DESTDIR* parameter, as that will cause the Cygwin debug build of XFree86 to be installed overtop of your local X Window System installation, which would completely destroy your host systems' X Window System installation.

```
[harold@MyCrossHost debug]$ makeg install install.man
DESTDIR=/stagingdir
IMAKE_DEFINES="-D__CYGWIN__ -Ulinux" > install.log 2>&1
[harold@MyCrossHost debug]$
```

Chapter 3. Documentation

Foo!

Chapter 4. Web Site Maintenance

Foo!

Bibliography

Books

[ScheiflerGettys92] Robert W. Scheifler, James Gettys, Jim Flowers, and David Rosenthal, 1992, 1-55558-088-2, Butterworth-Heinemann, *X Window System: The Complete Reference to Xlib, X Protocol, ICCCM, and XLFD*.

[Richter99] Jeffrey Richter, 1999, 1-57231-996-8, Microsoft® Press, *Programming Applications for Microsoft® Windows®: Mastering the critical building blocks of 32-bit and 64-bit Windows®-based applications*.

[Petzold99] Charles Petzold, 1999, 1-57231-995-X, Microsoft® Press, *Programming Windows: The definitive guide to the Win32® API*.

[McKay99] Everett N. McKay, 1999, 0-7356-0586-6, Microsoft® Press, *Developing User Interfaces for Microsoft® Windows®: Practical and effective methods for improving the user experience*.

[JonesOhlund99] Anthony Jones and Jim Ohlund, 1999, 0-7356-0560-2, Microsoft® Press, *Network Programming for Microsoft® Windows®: Clear, practical guide to Microsoft's networking APIs*.

[Yuan01] Feng Yuan, 2001, 0-13-086985-6, Prentice Hall PTR, *Windows® Graphics Programming: Win32® GDI and DirectDraw®*.

[CohenWoodring98] Aaron Cohen and Mike Woodring, 1998, 1-56592-296-4, O'Reilly & Associates, Inc., *Win32® Multithreaded Programming: Building Thread-Safe Applications*.

[CameronRosenblattRaymond96] Debra Cameron, Bill Rosenblatt, and Eric Raymond, 1996, 1991, 1-56592-152-6, O'Reilly & Associates, Inc., *Learning GNU Emacs: UNIX Text Processing*.

[Lewine91] Edited by Dale Dougherty, Donald A. Lewine, 1991, 0-937175-73-0, O'Reilly & Associates, Inc., *POSIX Programmer's Guide: Writing Portable UNIX Programs*.

[KernighanRitchie88] Brian W. Kernighan and Dennis M. Ritchie, 1998, 1978, 0-13-110370-9, Prentice Hall PTR, *The C Programming Language: ANSI C*.

Glossary

C

Concurrent Versions System

CVS is an open source version control system used by the majority of open source projects. More information can be found at the CVS project homepage (http://www.cvshome.org).

F

firewall

Firewall software attempts to protect an internal network from intrusions originating from an external network.

P

pserver

CVS pserver, short for “password server”, is one of the user authentication methods supported by CVS. CVS pserver is not secure, as passwords are transmitted and stored as plain text. However, CVS pserver is desireable for read-only anonymous access to open source CVS trees, as CVS pserver is by far the easiest method to use.

V

Virtual Private Network

Virtual Private Networks are encrypted tunnels through which private data can be safely transmitted

over a private network (e.g. the Internet).

X

X Display Manager

An X Display Manager presents a graphical login screen to X users. Often an XDM will allow the user to select a desktop environment or window manager to be for their login session. Some X Display Managers are xdm, gdm (Gnome Display Manager), and kdm (KDE Display Manager).

X Display Manager Control Protocol

XDMCP allows XDM to process logins for users remote to the machine that XDM is running on; login sessions will be run on the machine running XDM. For example, at a university you may use XDMCP to login to an X session running on an engineering department computer from your dorm room.

See Also: X Display Manager.

Appendix A. GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not

explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the

Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section’s title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may

differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.