Portage56k 0.1 Manual

Andrew Roberts
(dev@andy-roberts.net)

23rd February 2004

Introduction

Portage56k is a program that makes it easier to install Gentoo portage packages on a system with (slow) dial-up Internet access, as long as you have access to another system that does have broadband Internet access. *Portage56k* is used on the system to which the software will be installed, and it produces a list of URLs that must be downloaded to install the software. These URLs can then be downloaded on the system with fast Internet, and copied to the target system via removable media (such as CDROMs).

Portage56k motivation

If you are reading this, then you are already most likely a Gentoo user, or at least considering becoming one. If you are the former, then in order to get Gentoo installed, you must already be familiar with the superb Portage package management system. If you are the latter, then you most likely have heard of Portage, which is what aroused your interest in Gentoo in the first place!

The Portage system works very well, and is a key factor in Gentoo's success as a Linux distribution. However, it largely favours those users who have direct access to a high speed Internet connection. This is because Gentoo prefers you to install the most up-to-date components from scratch. And rather than bundling all together on CDs that get released every year, the Gentoo Portage Tree makes all packages available online (and constantly updates). To pull even a modest system (kernel, shell, standard unix tools, XF86 and a popular window manager) will obviously require the downloading of many tens (or more likely hundreds) of megabytes. So surely that rules out any users who still only use a slow dial-up Internet connection? Well, not necessarily.

I myself am one of the unenlightened (otherwise known as tight-fisted!) and have yet to invest in a broadband connection. However, I do have a good reason: I spend my days in the School of Computing at the University of Leeds, UK, studying for my PhD. Our department has a ridiculously large amount of bandwidth of which I am free to use :) With the other Linux distributions I used to use, e.g., Red Hat, I would simply download any updates when I required them, burned to a CD and took it home to install the packages. I'd much prefer to still use a similar arrangement (after all, my grant isn't that large!)

You don't need to build Gentoo completely from scratch nowadays, with the Gentoo LiveCDs. A couple of ISO images that can be burned to disc, containing everything you need to get a Gentoo system up and running. Most of the packages are pre-compiled, so you can be up and running fairly quickly. So what do you do when you want to install the latest version of KDE, or update the kernel? How does the Portage system help you? Well, not a lot as it happens. You can do an '`emerge -p <package name>`' to discover what needs to be installed/updated to get your package running. But that doesn't tell you the actual file names of the tarballs that need to be downloaded. The

Gentoo FAQ recommends using '`emerge -pf`' to find out this information, but the output is that useful: it lists the filename, but it repeats it for every mirror it knows of. And what is particularly annoying, is that the output of the emerge cannot be redirected to a file (actually, some information does redirect, such as the 'Calculating dependencies' message, but the actual info we want to know doesn't!)

This wasn't adequate, in my opinion at least. Yet, I didn't want to give up using Gentoo so soon after I'd begun, because it still has so many good features. I therefore knocked up a quick Perl script to parse the output of a standard '`emerge -p`' (which can be redirected and piped.) The emerge tells you which packages it intends to install. The script will then search through the local Portage tree on the users' system to obtain all the necessary file names for every package. And because I know my local Gentoo mirror, I could output a complete list of URLs for each file to be downloaded from my local mirror. All I needed to do was to transfer this text files of URLs to my PC at university, run '`wget -i urls.txt`' and I'd have everything I needed downloaded in seconds. Burn to disc, take back to my home PC, copy all the distfiles to `/usr/portage/distfiles/` and then just run the emerge (without the -p flag!)

And too be fair, it's working fine at the moment :) But then my friend (a massive Gentoo fan) convinced me that there may be many other people out there in a similar situation: wanting to use Gentoo, but fear that without a broadband connection, it is not practical. If I were to expand upon my little humble script to make it a little more functional and user-friendly, then this could help the Gentoo movement by ensuring that dial-up users are not alienated. And hence *Portage56k* was born.

Installation

Portage56k is only in its early stages. I have not yet managed to create ebuilds for the tool. However, it's currently only a simple script, and so is easy to install! As it is scripted in Perl, you will obviously need to ensure that you have Perl installed on your system! (N.B., the hash (#) represents the command prompt.)

1. Download the tarball from `http://portage56k.sourceforge.net/downloads/`
2. Unpack the tarball:
   ```
   # tar xvfj portage56k-0.1.tar.bz2
   # cd portage56k-0.1
   ```
3. Ensure that the portage56k script has execute permissions. If not:
   ```
   # chmod 700 portage56k
   ```

Using Portage56k

At the moment, portage56k is only very basic. It can currently retrieve the required files for any package you wish to upgrade/install. It relies on emerge still being run with its output being redirected or piped to portage56k. The output from portage56k is just a list of URLs. (N.B., the hash (#) represents the command prompt.)

1.
   ```
   # emerge -p <package-name> | portage56k > urls.txt
   ```
   (Of course, replace '`urls.txt`' with any filename you wish)
2. Transport `urls.txt` to machine with fast Internet connection.
3. To download the files, you are free to use any tool that can connect to a FTP server and retrieve files. I recommend using the '`wget`' tool that can take a text file of URLs and download automatically.

```
# wget -i urls.txt
```
This is very useful if you need to download a lot of files. It would otherwise be very tedious to do it interactively with the standard '`ftp`' command!

4. Once all required files are downloaded, it is assumed that you have the means to copy them to some portable media. Burning to CD is the obvious choice. Instructions on how to use CD burning software will not be provided! The '`cdrecord`' tool will do everything you need to do, and k3b is a decent frontend.

5. Bring files to original PC. Copy all downloaded files to `/usr/portage/distfiles`.

6. Finally, run the '`emerge`' command (without the `-p`):
```
# emerge <package-name>
```
Emerge checks whether the files it needs are in the distfiles directory, if they are, then will proceed with compilation and merging. Otherwise, it will try to download.

Troubleshooting

I get the error "portage56k: Command not found."

There are two possible reasons for this error:

1. The most likely is due to the *portage56k* script not being in your path. There are a number of ways of solving this issue:

   - Copy script into a system `bin` directory (Needs root permission):
     ```
     (Assuming you are in the directory with portage56k script)
     # cp portage56k /usr/local/bin/
     ```
   - Link script into a system `bin` directory (Needs root permission):
     ```
     # cd /usr/local/bin
     # ln -s /path/to/script/portage56k
     ```
   - Create a local bin directory, copy script there, then add directory to your path.
     ```
     # mkdir ~/bin
     # export PATH=$PATH:$HOME/bin        (in bash or ksh)
     # set path=($path ~/bin)              (in csh or tcsh)
     ```
   - Alternatively, to run programs that are not in any directories in the *path* variable, the shell expects you to qualify as follows:
     ```
     # ./portage56k
     ```

2. The other possible explanation is that the Perl intepreter is not installed in the usual place (or at least not made accessible.) The current *portage56k* script expects to find Perl in `/usr/bin`. To check if Perl can be found via the PATH, type '`which perl`'. If it doesn't find anything, then I'd recommend installing Perl again. If it does find the '`perl`' command then edit the *portage56k* script using you text-editor of choice, and modify the first line to reflect the locate of the perl command on your system.

Todo

Short-term goals:

1. Create a simple install script.
2. Create a Gentoo ebuild package.
3. Port Perl code over to Python.
4. Make documentation more comprehensive and available in various formats.
5. Complete the Portage56k project website — important for access to news, code, support docs, etc.

Long-term goals:

1. Incorporate 'emerge' phase into portage56k itself, rather than having to manually redirect emerge output.
2. Add download functionality, so that if portage56k is also installed on the *download* PC, you can run portage56k with a specific argument, which will take care of the downloading, rather than having to manually grab files with 'wget' or other such tools.
3. Interface to the 'cdrecord' program to automatically burn downloaded files to disc.
4. A *GUI*!

Changelog

Version 0.1 — Released: 20th February 2004

- Initial Release.

Comments

If you have any comments or spotted bugs, etc., please contact `dev@andy-roberts.net`.