Gentoo Workshop

Mark Platek
Clarkson University
Fall 2010

Resources

- Gentoo Handbook:

  http://www.gentoo.org/doc/en/handbook/handbook-x86.xml

  http://www.gentoo.org/doc/en/handbook/handbook-amd64.xml

- Gentoo FAQs and HOWTOs:

  http://www.gentoo.org/doc/en/index.xml?catid=desktop

  http://en.gentoo-wiki.com/wiki/Main_Page

- Gentoo Forums

  http://forums.gentoo.org

- Myself – ask if you have questions!

  Email: platekme@clarkson.edu (or platekme89@gmail.com)

  IM: backslash54 v2

Now, where were we?

- At this point, everyone should have a bootable system that is fully up to date.
- We'll be covering the following topics today:
  - More about portage
  - ALSA Configuration
  - X Server installation
  - Portage overlays
  - Anything else you want to know about

Portage: dealing with deprecated packages

- When packages are deprecated and are to be dropped from the portage tree, a message will be printed after you sync.
- Sync your portage trees – you should get a warning about package slocate, which will be dropped from the portage tree next month. It advises replacing slocate with mlocate.
- We'll remove our installed version of slocate with the following command:

  `emerge --depclean slocate`
- We could also use `emerge --unmerge slocate`, but unmerge isn't dependency-aware.
- Finish by emerging mlocate.

equery

- I mentioned that you can use equery to find out which USE flags are available for which ebuilds, let's explore that further.
- Equery is available in the 'gentoolkit' package, emerge that first.
- Try running `equery uses mplayer`. Without checking the USE flags first, we might have built mplayer without support for anything!
- As always, you can add USE flags to package.use in /etc/portage!
- Equery has other uses, check the man page for more information.

revdep-rebuild

- Sometimes the dependency tree will break when certain packages (especially libraries) are updated.
- Portage provides tools to help us deal with this problem. revdep-rebuild searches through all installed packages, looking for ones that were built upon old versions of libraries. It subsequently re-emerges them.
- You'll need to use revdep-rebuild at some point, as broken packages won't work again until they are re-emerged (since old library files are removed).
- You can invoke revdep-rebuild with `revdep-rebuild`. You can also give it a certain library name to rebuild all packages for.

lafilefixer

- The lafilefixer tool performs a similar job. It doesn't re-emerge, though, it just updates links to libtool archives (.la files).
- This is important because installed packages share .la files, so if a package is changed it could affect others.
- This tool is found in the 'lafilefixer' package. You should emerge it now, and run it periodically: `lafilefixer --justfixit`.

Aside: package names

- So far, we've been referring to packages by name. “firefox” installs firefox, and so on.
- But, the name of the firefox package is actually:

  www-client/firefox
- The red part is a directory in /usr/portage. The blue part is a directory in /usr/portage/www-client. As you can see, the portage tree is broken down into categories. Also in www-client, you find packages like chromium, konqueror, etc.
- You can also refer to specific versions of packages. For instance,

  =www-client/firefox-3.6.12
- refers to version 3.6.12 of firefox. A package referred to in this way is called an “atom”.
- The <package>-<version> syntax is generally adhered to, but not always. Use `emerge -s <package>` to see what the version syntax is.
- You can also use >/>= and </<= to denote higher or lower versions than the given atom.

Cleaning up after your machine, part 1

- As packages are updated, dependencies are sometimes dropped. This leaves packages installed on your system that are not claimed as dependencies.
- You can clean these packages with `emerge --depclean`. First, make sure your system is completely up to date:

  `emerge -uDN world`
- Run depclean with -p first to make sure the results are what you expect:

  `emerge -p --depclean`
- This tells us that some perl packages are no longer necessary. Run it again without the -p to actually unmerge them.
- Afterwards, run revdep-rebuild and lafilefixer to make sure nothing was broken. You should always run revdep-rebuild after unmerging a package.

Cleaning up after your machine, part 2

- Very little source code actually lives in the portage tree. Sources are generally downloaded as-needed to /usr/portage/distfiles.
- Portage doesn't delete source files, even for outdated packages. This is so that mirrors aren't strained unnecessarily. But, it leaves old source code laying around that is unlikely to be used again.
- We can use the `eclean` utility to remove old sources from /usr/portage/distfiles. On an active system with lots of things installed, this can be gigabytes of data!
- Let's try it now:

  `eclean -p distfiles`
- Run it again without the -p to actually perform the cleaning. Run it with --destructive to remove all but the bare minimum source code to reinstall all packages on the system.

Aside: starting init scripts

- In Ubuntu and Fedora, you control init scripts with the 'service' command. In Gentoo, you do it directly. For example,

  `/etc/init.d/mysqld start`

- calls the mysqld init script with the argument 'start'. This starts the init script.
- Some other common arguments are 'restart' and 'stop'. You can probably guess what they do.
- Call an init script without any arguments and it will print its syntax.

ALSA

- Let's continue with our installations. We'll start by setting up ALSA.
- ALSA support is enabled in our kernels, but we don't have sound yet! Start by emerging the 'alsa-utils' package.
- Then, add the 'alsasound' init script to the boot runlevel:

  `rc-update add alsasound boot`
- This is one of the few init scripts that should ever be added to the boot runlevel.
- Start alsasound:

  `/etc/init.d/alsasound start`
- Now, check your levels with `alsamixer`.
- Finally verify that it works by emerging the 'madplay' package and downloading this test audio file (of Linus Torvalds giving the correct pronunciation of “Linux”):

  http://www.paul.sladen.org/pronunciation/torvalds-says-linux.mp3

X Server: Preface

- Crushing your expectations:

  You probably won't get your X server working during this workshop. It takes time and fiddling to get it right.

X Server I

- Unless you are Mark Shuttleworth, you will want to use the X server for displaying your GUI. However, installing it can be a bit trying, depending on how well-supported your graphics card is.
- We'll start by adding a few X-specific variables to make.conf. First,

  INPUT_DEVICES=”evdev”
- This indicates to use HAL and udev to autoconfigure input peripherals. Add 'synaptics' in there if you have a touchpad.

  VIDEO_CARDS=””
- The contents of this variable will depend on what video card you have. Please refer to the Gentoo X Server FAQ to find out what you have to do:

  http://www.gentoo.org/doc/en/xorg-config.xml
- At this point, everyone will have to set their variables differently. Please ask if you don't understand what to do.

X Server II

- The next step is to emerge the X Server itself. Once you are certain your make.conf is set up correctly, run the following:

  `emerge -qv xorg-server hal twm xterm`
- After that, update your environment again: `env-update && source /etc/profile`.
- Add the HAL daemon to the default runlevel, and start it up:

  `rc-update add hald default && /etc/init.d/hald start`
- HAL configures your input based on rules set in certain configuration files. Place them now:

  `cp /usr/share/hal/fdi/policy/10osvendor/10-input-policy.fdi /etc/hal/fdi/policy`
  `cp /usr/share/hal/fdi/policy/10osvendor/10-x11-input.fdi /etc/hal/fdi/policy`
- You can edit the files just placed in /etc/hal/fdi/policy to your liking. Touchpad users, note that you'll need to set up tapping this way. There is a good touchpad FAQ:

  http://en.gentoo-wiki.com/wiki/Synaptics_Touchpad
- When you are finished, restart hald.

X Server III

- Now, set your XSESSION variable by creating /etc/env.d/90xsession:

  `echo XSESSION="twm" > /etc/env.d/90xsession`
- We're using twm for now, you'll want to set this to “gnome” eventually.
- The next step is to try starting X without an xorg.conf. If you can do this, you are *very* fortunate.

  `startx`
- Everyone else will have to make an xorg.conf. X can automatically generate one, so let's try that:

  `Xorg -configure`
- Now try to start X with the new xorg.conf generated:

  `X -retro -config /root/xorg.conf.new`
- If all it well, you'll get a grey screen and mouse.

X Server IV

- Once you can reliably start the X server, copy /root/xorg.conf.new to /etc/X11/xorg.conf.
- From here begins the fiddly process of tweaking your xorg.conf.
- If you are using evdev, you should remove all references to mice and keyboard from xorg.conf, or there will be conflicts and you won't be able to use your input devices.
- The rest of xorg.conf is beyond the scope of this presentation. I strongly encourage you to look at the X Server howto:

  http://www.gentoo.org/doc/en/xorg-config.xml

Portage Overlays

- The portage tree has a great deal of software available, but it is of course not comprehensive. It is extensible, however - overlays (in the form of additional software trees) can be added to the portage tree.
- The tool used to manage overlays is `layman`. Emerge it now.

  `emerge layman git subversion`
- It's necessary to install VC tools because overlays are generally maintained under VC, unlike the portage tree. You must sync overlays manually.
- Add the following line to your make.conf:

  ```
  source /var/lib/layman/make.conf
  ```
- Since that file doesn't exist yet, we need to create it. Add this line:

  ```
  PORTDIR_OVERLAY=""
  ```
- Now, we can start using layman!

Portage Overlays II

- To list available overlays, invoke `layman -L`.
- To add an overlay:

  `layman -a <overlay name>`
- Now we can install packages from that overlay as if they were part of the portage tree!
- Don't forget to periodically sync your overlays with `layman -S`.
- If you don't want an overlay, remove it like so:

  `layman -d <overlay name>`
- Overlays are often used to development purposes, or for beta-quality software. You generally won't need to use them.

Questions?

- This is the end of the material I have prepared, but do you want to try anything else?
- Do you have any questions about Portage, or Gentoo in general?

Thank you for participating, I hope you guys enjoyed it. If you have any questions/problems/etc. In the future, I'm more than happy to help.

Copyright