Linux Package Management

Nhiều hướng dẫn tài liệu tham khảo "package managers" và "package management tools." Nếu bạn là người mới đến thế giới [Linux](https://mangmaytinh.net/tags/linux/) và không hiểu mục đích của các công nghệ này, hoặc nếu bạn đã quen thuộc với các công cụ quản lý một gói nhưng muốn tìm hiểu làm thế nào để sử dụng những cái khác, hướng dẫn này sẽ cung cấp một giới thiệu về các công cụ quản lý gói chính.  
  
**Các khái niệm quản lý gói**  
Các bản phân phối hiện đại của hệ điều hành Linux đều cài đặt phần mềm trong các gói biên dịch trước, đó là nơi lưu trữ chứa các chương trình phần mềm, các file cấu hình, và thông tin về các phụ thuộc. Hơn nữa, công cụ quản lý gói theo dõi các bản cập nhật và nâng cấp vì vậy chúng ta không cần lo lắng về lỗi và bảo mật.  
  
Nếu không có quản lý gói, người sử dụng phải đảm bảo rằng tất cả các phụ thuộc cần thiết cho một phần của phần mềm được cài đặt và cập nhật, biên dịch các phần mềm từ mã nguồn (điều này sẽ rất mất thời gian), và quản lý cấu hình cho mỗi phần của phần mềm.  
  
Hệ thống quản lý gói cố gắng giải quyết những vấn đề này, dựa vào đó các nhà phát triển có thể nâng cao chất lượng tổng thể và thống nhất của một hệ điều hành dựa trên Linux. Các tính năng mà hầu hết các ứng dụng quản lý gói cung cấp là:  
  
**Package download**: Dự án hệ điều hành cung cấp các kho của các gói cho phép người dùng tải về các gói của họ từ một nhà cung cấp đáng tin cậy. Khi bạn tải về từ một người quản lý gói phần mềm, phần mềm có thể được xác thực và sẽ được giữ trong kho lưu trữ thậm chí nguồn ban đầu trở nên không đáng tin cậy  
  
**Dependency resolution**: Gói chứa siêu dữ liệu (meta-data)cung cấp thông tin về các gói khác được yêu cầu bởi gói trong câu hỏi. Điều này cho phép các ứng dụng và phụ thuộc của họ được cài đặt với một lệnh, và cho phép hệ điều hành để quản lý các bản cập nhật cho các gói.  
  
**A standard binary package format**: Các gói được chuẩn bị một cách thống nhất trên toàn hệ thống để thực hiện cài đặt dễ dàng hơn.  
  
**Common installation and configuration locations**: Các nhà phát triển bản phân phối Linux thường có quy ước làm thế nào ứng dụng được cấu hình và cách bố trí của các tập tin trong thư mục /etc / và thư mục /etc/init.d; bằng cách sử dụng các gói, sự phân phối có thể thực thi một tiêu chuẩn duy nhất.  
  
**Additional system-related configuration and functionality**: đôi khi, các nhà phát triển hệ điều hành sẽ phát triển bản vá lỗi và các script helper cho phần mềm của họ mà được phân phối trong các gói. Những thay đổi này có thể ảnh hưởng đáng kể đến kinh nghiệm người dùng.  
  
**Quality control**: Các nhà phát triển hệ điều hành sử dụng các quá trình đóng gói để kiểm tra và đảm bảo rằng phần mềm là ổn định và ít lỗi mà có thể ảnh hưởng đến chất lượng sản phẩm, và các phần mềm không làm cho hệ thống trở nên không ổn định.  
  
Nói chung, chúng tôi khuyên bạn nên cài đặt các phiên bản của phần mềm có sẵn trong kho của phân phối của bạn và đóng gói cho hệ thống điều hành của bạn. Nếu gói cho các ứng dụng hoặc phần mềm mà bạn cần phải cài đặt không có sẵn, chúng tôi khuyên bạn nên tìm các gói trong kho, trước khi cài đặt từ mã nguồn.  
Phần còn lại của hướng dẫn này sẽ giới thiệu cách sử dụng các hệ thống quản lý gói cụ thể và làm thế nào để biên dịch và đóng gói phần mềm.  
  
**Quản lý gói**[**Debian**](https://mangmaytinh.net/tags/debian/)**và**[**Ubuntu**](https://mangmaytinh.net/tags/ubuntu/)  
Hệ thống quản lý gói Debian, dựa trên một công cụ gọi là dpkg với hệ thống apt là một biện pháp hiệu quả, phổ biến và hữu ích của quản lý gói. Ngoài Debian, một số bản phân phối nổi bật khác của GNU/Linux có nguồn gốc từ hệ thống Debian, đáng chú ý là bản phân phối Ubuntu.  
Những hướng dẫn sau áp dụng cho các hệ thống Debian và Ubuntu.  
  
**Công cụ nâng cao gói (APT)**  
Bạn có thể đã quen với apt-get, một lệnh trong đó sử dụng các công cụ đóng gói tiên tiến để tương tác với hệ thống gói của hệ điều hành. Các lệnh có liên quan và hữu ích nhất là (được chạy với quyền root):

* **apt-get install package-name(s)** - Cài đặt gói phần mền quy định, cùng với bất kỳ phụ thuộc
* **apt-get remove package-name(s)**- Loại bỏ gói phần mềm quy định, nhưng không loại bỏ sự phụ thuộc
* **apt-get autoremove** - Loại bỏ bất kỳ phụ thuộc.
* **apt-get clean** - Loại bỏ các tập tin gói đã tải về (.deb) cho các phần mềm đã được cài đặt
* **apt-get purge package-name(s)** - Kết hợp các chức năng của loại bỏ và làm sạch cho một gói cụ thể, cũng như các file cấu hình
* **apt-get update** - Đọc tập tin /etc/apt/sources.list và cập nhật dữ liệu của hệ thống về gói sẵn để cài đặt. Chạy lệnh này sau khi thay đổi tập sources.list.
* **apt-get upgrade** - nâng cấp tất cả các gói có bản cập nhật có sẵn. Chạy lệnh này sau khi chạy apt-get update.

Trong khi **apt-get** cung cấp các chức năng thường dùng, lệnh **apt-cache** cung cấp thêm thông tin:

* **apt-cache search package-name(s)** - Nếu bạn biết tên của một phần mềm nhưng apt-get install không thành công hoặc điểm trỏ đến phần mềm sai, điều này có vẻ tên đã bị thay đổi.
* **apt-cache show package-name(s)** - Hiển thị thông tin phụ thuộc, số phiên bản và mô tả cơ bản của gói.
* **apt-cache depends package-name(s)** - Liệt kê những gói cụ thể mà phụ thuộc vào gói chính. Đây là những gói sẽ được cài đặt với apt-get install.
* **apt-cache rdepends package-name(s)** - kết quả đầu ra là một danh sách các gói mà phụ thuộc vào một gói cụ thể. Danh sách này thường xuyên có thể khá dài, vì vậy tốt nhất nên kết hợp thêm lệnh **less.**
* **apt-cache pkgnames**- Tạo ra một danh sách các gói cài trên hệ thống của bạn. Danh sách này thường là khá dài, vì vậy tốt nhất nên kết hợp thêm lệnh **less**, hoặc đưa output vào một tập tin văn bản.

Kết hợp hầu hết các lệnh này với chương trình apt-cache có thể cung cấp cho bạn rất nhiều thông tin hữu ích về hệ thống của bạn, phần mềm bạn có thể muốn cài đặt, và các phần mềm mà bạn đã cài đặt. Nếu bạn đang bị choáng ngợp bởi apt-cache hãy xem qua các tài liệu sau đây:

* [The Debian Package Directory](https://www.debian.org/distrib/packages)
* [The Ubuntu Package Directory](http://packages.ubuntu.com/)

**Aptitude**  
Aptitude là một giao diện front-end cho APT. Ngoài một giao diện đồ họa, Aptitude cung cấp một giao diện dòng lệnh kết hợp với chức năng APT nhất định. Một số lệnh đáng chú ý là:

* **aptitude update, aptitude install, aptitude remove, aptitude clean, or aptitude purge** - Tương tự như apt-get.
* **aptitude search or aptitude show**- Tương tự như apt-cache.
* **aptitude download** - Tải về một tập tin .deb cho một gói vào thư mục hiện hành.

Aptitude cũng bao gồm nâng cấp an toàn, có nghĩa là nó không loại bỏ các gói hiện có, mà ngăn cản hệ thống nâng cấp các gói cụ thể.  
  
**/etc/apt/sources.list**  
Tập tin /etc/apt/sources.list điều khiển các kho gói từ APT. Tập tin này chứa các dòng trong định dạng sau:

Mã:

deb location-of-resources distribution component(s)

Một số ví dụ:

Mã:

deb http://mirrors.aluhost.com/debian/ jessie main contrib

deb http://www.deb-multimedia.org jessie main non-free

Nói chung, người ta không muốn thêm các mục mới đến sources.list mà không có nhiều xem xét kỹ lưỡng, cập nhật các bộ nhớ cache gói với các kho bổ sung và chạy nâng cấp đôi khi có thể dẫn đến việc cài đặt các gói bị hỏng, phụ thuộc không được đáp ứng, và hệ thống không ổn định . Trong các hệ thống của Debian, hạ cấp thường rất khó.  
  
Phần component của dòng chia kho dựa trên số lượng hỗ trợ mà các nhà phát triển của hệ điều hành có thể cung cấp cho các gói (ví dụ như main vs contrib).  
  
Việc bố trí các nguồn sources.list là một chút khác nhau trong hệ thống Ubuntu. Các dòng có định dạng giống nhau nhưng tên của các nhà phân phối và các thành phần khác nhau:

* Phiên bản Ubuntu có một tên scheme khác nhau. Phiên bản 14.04 có tên là "trusty" trong sources.list, 15.10 là "wily" và 16.04 là "xenial." Những tên theo một mẫu tự chữ cái.
* Thành phần Ubuntu là: "main" và "restricted" cho hỗ trợ gói miễn phí và không miễn phí; “universe” và “multiverse” không hỗ trợ phần mềm miễn phí và không miễn phí.

**Sử dụng dpkg**  
apt-get và apt-cache cung cấp một giao diện dễ dùng hơn, và kết nối đến các kho cho các công cụ quản lý gói cơ bản có tên dpkg và debconf. Những công cụ này khá mạnh mẽ, và giải thích chức năng đầy đủ của chúng là vượt ra ngoài phạm vi của tài liệu này. Tuy nhiên, một sự hiểu biết cơ bản về cách sử dụng những công cụ này rất hữu ích. Một số lệnh quan trọng là:

* **dpkg -i package-file-name.deb**- Cài đặt một file .deb
* **dpkg --list search-pattern** – Liệt kê danh sách gói hiện được cài đặt trên hệ thống
* **dpkg --configure package-name(s)** - Chạy một giao diện cấu hình để thiết lập một gói.
* **dpkg-reconfigure package-name(s)**- Chạy một giao diện cấu hình trên một gói đã được cài đặt

Để biết thông tin về việc xây dựng gói riêng của bạn, hãy tham khảo [Debian New Maintainers Guide](https://www.debian.org/doc/manuals/maint-guide/).  
  
**Quản lý Package**[**Fedora**](https://mangmaytinh.net/tags/fedora/)**và**[**CentOS**](https://mangmaytinh.net/tags/centos-8/)  
Fedora và CentOS là các bản phân phối liên quan chặt chẽ, là upstream và downstream (tương ứng) của Red Hat Enterprise Linux. Sự khác biệt của chúng xuất phát từ cách gói được chọn để đưa vào kho.  
  
Cả hai hệ thống sử dụng yum để tương tác với kho hệ thống và cài đặt phụ thuộc, và cũng bao gồm một công cụ cấp thấp hơn được gọi là rpm, cho phép bạn tương tác với từng gói.  
  
**Yellow Dog Updater, Modified (YUM)**  
Công cụ YUM được phát triển cho hệ thống Yellow Dog Linux như là một thay thế cho Yellow Dog Updater (yup). RedHat tìm thấy công cụ YUM là một bổ sung có giá trị cho hệ thống của họ. Ngày nay, YUM là gói mặc định và kho lưu trữ công cụ quản lý cho một số hệ điều hành.  
  
Bạn có thể sử dụng các lệnh sau để tương tác với YUM:

* **yum install package-name(s)** - Cài đặt các gói cùng với bất kỳ phụ thuộc yêu cầu
* **yum remove package-name(s)**- Loại bỏ các gói cụ thể từ hệ thống của bạn
* **yum search search-pattern**- Tìm kiếm danh sách các tên gói và mô tả cho các gói phù hợp với mô hình tìm kiếm và cung cấp một danh sách các tên gói, với kiến trúc và mô tả ngắn gọn về các nội dung gói.
* **yum deplist package-name(s)** - Danh sách tất cả các thư viện và các module mà gói có tên phụ thuộc vào, cùng với tên của các gói (bao gồm cả các phiên bản) cung cấp phụ thuộc.
* **yum check-update** - làm mới bộ nhớ cache cục bộ của cơ sở dữ liệu yum vì vậy thông tin phụ thuộc và các gói mới nhất luôn được cập nhật.
* **yum info package-name(s)** - Kết quả của lệnh info cung cấp tên, mô tả của gói, cũng như liên kết tới trang chủ cho phần mềm, phiên bản phát hành và kích thước cài đặt của phần mềm.
* **yum reinstall package-name(s)** - Xóa và sau đó tải về một bản sao mới của tập tin gói và cài lại đặt các phần mềm trên hệ thống của bạn
* **yum localinstall local-rpm-file**- Kiểm tra sự phụ thuộc của một file .rpm địa phương và sau đó cài đặt nó
* **yum update optional-package-name(s)** - Tải xuống và cài đặt tất cả các bản cập nhật bao gồm các bản vá lỗi, phiên bản bảo mật và nâng cấp, được cung cấp bởi các nhà phân phối của hệ thống điều hành của bạn.
* **yum upgrade** - Nâng cấp tất cả các gói được cài đặt trong hệ thống của bạn lên phiên bản mới nhất.

**RPM Package Manager (RPM)**  
Các lệnh sau đây nên được chạy dưới quyền user root. Một số lệnh phổ biến:

* **rpm --install --verbose --hash local-rpm-file-name.rpm or rpm -ivh filename.rpm**- Cài đặt gói rpm từ tập tin. rpm cũng có khả năng cài đặt các file RPM từ nguồn http và ftp.
* **rpm --erase package-name(s) hoặc rpm -e** - Loại bỏ các gói. Thông thường sẽ không hoàn thành nếu package-name phù hợp với nhiều hơn một gói.
* **rpm --query --all hoặc rpm -qa** - liệt kê tên của tất cả các gói cài đặt hiện tại.
* **rpm --query package-name(s) hoặc rpm -q** - cho phép bạn để xác nhận liệu một gói đã được cài đặt trong hệ thống của bạn chứa.  
  **rpm --query --info package-name(s) hoặc rpm -qi** - hiển thị các thông tin về một gói cài đặt
* **rpm --query --list package-name(s) hoặc rpm -ql** - tạo ra một danh sách các tập tin cài đặt của một gói phần mềm chỉ định. Điều này được bổ sung bởi:
* **rpm --query --file hoặc rpm -q qf filename** - kiểm tra để xem gói được cài đặt sở hữu các tập tin nào.

Lưu ý RPM không tự động kiểm tra phụ thuộc, vì vậy bạn phải cài đặt chúng bằng tay. Để biết thêm thông tin xin vui lòng tham khảo những nguồn bên ngoài:

* [iDevelopment Info - RPM Commands](http://www.idevelopment.info/data/Unix/Linux/LINUX_RPMCommands.shtml)
* [Quick Guide to RPM](http://www.tfug.org/helpdesk/linux/rpm.html)

**Tập tin /etc/yum.conf**

Tập tin nằm ở thư mục /etc/yum.conf cung cấp tùy chọn cấu hình hệ thống cho YUM, cũng như thông tin về kho. Thông tin kho lưu trữ cũng có thể nằm trong các tập tin kết thúc bằng .repo trong tập tin /etc/yum.repos.d  
Các tùy chọn trong đoạn [main] không cần phải sửa đổi, mặc dù bạn có thể thiết lập các log và bộ nhớ cache thay thế cho cơ sở dữ liệu bằng cách thêm vào các dòng sau:

Mã:

vi /etc/yum.conf

Mã:

logfile=/var/log/yum.log

cachedir=/var/cache/yum

Bạn có thể sử dụng mẫu sau đây để xác định một đoạn mới cho một kho lưu trữ mới, thay thế với giá trị riêng của bạn:

Mã:

vi /etc/yum.repos.d/example.repo

Mã:

[REPO-NAME]

name=REPOSITORY-NAME

mirrorlist=HTTP-ACCESSIBLE-MIRROR-LIST

#baseurl=BASE-URL-FOR-REPOSITORY

gpgcheck=BOOLEAN-VALUE-TO-VERIFY-REPOSITORY

gpgkey=FILE-PATH-TO-GPG-KEY

Ví dụ sau đây là cấu hình mặc định cho kho lưu trữ "Base” trong CentOS 7:

Mã:

vi /etc/yum.repos.d/CentOS-Base.repo

Mã:

[base]

name=CentOS-$releasever - Base

mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os

#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/

gpgcheck=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7  
  
[**Slackware**](https://mangmaytinh.net/tags/slackware/)**Package Management**  
Gói trong Slackware Linux được phân phối như tarball, thường sử dụng gzip hoặc LZMA để nén. Những tarball có thể được nhận diện bởi các hậu tố, .tgz hoặc .txz. Định dạng này bao gồm một bố trí hệ thống tập tin hoàn chỉnh, cũng như các script khác sẽ được chạy khi cài đặt hoặc loại bỏ các phần mềm. Gói Slackware không cung cấp thông tin các phụ thuộc; điều này cho phép linh hoạt và kiểm soát hơn .  
  
Gói này cũng có thể được xây dựng bằng cách sử dụng SlackBuilds, script shell mà biên dịch mã nguồn hoặc đóng gói lại gói phân phối nhị phân cho dễ dàng cài đặt và gỡ bỏ trên Slackware. Những script được thông qua như là một tiêu chuẩn cộng đồng của các trang web như SlackBuilds.org, cung cấp nhiều gói thông thường của bên thứ ba không có sẵn trong Slackware.  
  
Slackware bao gồm **pkgtool** để quản lý gói địa phương và **slackpkg** cho cài đặt từ xa của các gói từ mirror chính thức. Đối với nhiệm vụ ít tương tác, có **installpkg**, **upgradepkg**, và **removepkg**.  
 **Làm việc với gói tại địa phương**  
Công cụ pkgtool cung cấp khả năng quản lý các gói trên hệ thống địa phương thông qua một giao diện đơn dựa trên văn bản. Mỗi lựa chọn là rất dễ sử dụng, như việc cài đặt các gói từ thư mục hiện tại hoặc một thư mục khác, loại bỏ các gói đã cài đặt, xem một danh sách các tập tin trong một gói, và chạy script Slackware Setup.  
  
Hoạt động cài đặt gói cung cấp một danh sách các gói trong một trình đơn, với khả năng lựa chọn những gói để cài đặt. Tùy chọn loại bỏ gói cung cấp một sự lựa chọn tương tự, với một danh sách các gói đã cài đặt. Xem một gói có thể hữu ích để xác định những gì có trong nó. Thông tin này bao gồm một mô tả bằng văn bản của user tạo ra gói cùng với danh sách của tập tin.  
  
Các tùy chọn cài đặt các script thường không áp dụng cho Linode, mặc dù có một tùy chọn netconfig có thể hữu ích cho một số người dùng.  
  
Đối với những người tìm kiếm một cách tiếp cận dòng lệnh, các lệnh \*pkg là khá đơn giản trong sử dụng.

* **installpkg package-name(s)** - Cài đặt một gói từ thư mục hiện tại, hoặc một đường dẫn bạn chỉ định.
* **upgradepkg package-name(s)** - nâng cấp một gói từ thư mục hiện tại, hoặc một đường dẫn bạn chỉ định.
* **removepkg package-name(s)** - Loại bỏ một gói từ hệ thống.

**Làm việc với gói điều khiển từ xa**  
Chương trình slackpkg là một bổ sung gần đây để Slackware cho phép các gói chính thức Slackware được cài đặt nâng cấp và sử dụng remote FTP hoặc HTTP mirror. Trước khi sử dụng Slackpkg, mirror nên được đặt trong /etc/slackpkg/mirror, và có thể được thêm vào hoặc chọn từ danh sách có sẵn. Chỉ có một mirror có thể được hoạt động, và được chọn bằng cách uncommenting nó (xóa kí tự #).  
  
Trong khi slackpkg cung cấp một giao diện dựa trên menu, nó cũng có thể được chạy trong một chế độ điều khiển chỉ bằng cách thiết lập DIALOG = off trong /etc/slackpkg/slackpkg.conf.

* **slackpkg check-updates**- Kiểm tra các mục mới tới changelog trên remote mirror. Điều này có thể hữu ích trong một script cron để thông báo cho người quản trị hệ thống các bản vá lỗi mới.
* **slackpkg update** - tải bản cập nhật cho các thay đổi và tập tin danh sách Slackware. Việc kiểm tra này rất hữu ích cho việc tìm kiếm cập nhật bảo mật, và phải được chạy trước khi cố gắng để tải về phần mềm cập nhật.
* **slackpkg install-new** - lệnh này sẽ được chạy khi nâng cấp lên một phiên bản mới.
* **slackpkg install package-name(s)** – Tìm kiếm cho bất kỳ gói phù hợp với tên chỉ định, và trình bày cho người dùng một menu cho phép lựa chọn cài đặt. Lưu ý rằng tất cả các lệnh chấp nhận một tên gói cũng sẽ làm việc với hàng loạt cài đặt Slackware, như ap, d, hoặc xap.
* **slackpkg upgrade-all** - Trình bày cho người dùng một menu liệt kê tất cả các gói trên remote mirror mà không phù hợp với phiên bản hiện tại đang được cài đặt trên hệ thống của bạn.
* **slackpkg upgrade package-name(s)**- Tìm kiếm bất kỳ gói phù hợp với tên cho trước, và trình bày một menu cho phép nâng cấp.
* **slackpkg clean-system** - Trình bày một menu liệt kê tất cả các gói trên hệ thống địa phương mà không có mặt trên upstream mirrors.
* **slackpkg remove package-name(s)** – Loại bỏ gói cụ thể trên hệ thống**.**
* **slackpkg reinstall package-name(s)** - Cài đặt lại các gói đã có trên hệ thống. Điều này rất hữu ích nếu các tập tin nhất định trong gói đã bị lỗi.
* **slackpkg search package-name(s)** - Tìm kiếm các tên gói nhất định, và hiển thị phù hợp với gói cũng như trạng thái cài đặt.
* **slackpkg file-search filename**- Tìm kiếm và cài đặt mô tả gói từ xa cho các tên tập tin nhất định, và hiển thị phù hợp với gói cũng như tình trạng cài đặt.
* **slackpkg blacklist package-name(s)** - Thêm các gói chỉ định vào danh sách đen nằm trong /etc/slackpkg/blacklist. Gói danh sách đen sẽ không được cài đặt, nâng cấp, hoặc xóa bỏ bởi slackpkg.
* **slackpkg info package-name(s)** - Hiển thị các thông tin về gói.

**Quản lý gói trong**[**Arch**](https://mangmaytinh.net/tags/arch/)**Linux với Pacman**  
Arch Linux sử dụng các gói nhị phân trong một định dạng .tar.xz, và cũng cung cấp một "cổng" xây dựng hệ thống tạo điều kiện cho xây dựng các gói.  
Arch Linux chạy trên một lịch trình phát hành, có nghĩa là các gói được bổ sung vào kho lưu trữ chính khi các phụ thuộc của chúng đã sẵn sàng để cung cấp.  
Do đó, quản trị viên của Arch Linux phải xem xét đầu ra của pacman cẩn thận trước khi quyết định nâng cấp hoặc cập nhật bất kỳ gói nào  
  
**Pacman**  
Công cụ pacman là rất mạnh mẽ, nhưng nó cũng rất đơn giản. Có ba lệnh lõi để quản lý gói cơ bản:

* **pacman --query package-name(s) hoặc pacman -Q** - Tìm kiếm cơ sở dữ liệu gói cho một package name và số phiên bản.
* **pacman --sync package-name(s) hoặc pacman -S** - Cài đặt gói mới, download nội dung mới cho các cơ sở dữ liệu và/hoặc nâng cấp hệ thống, tùy thuộc vào các tùy chọn và tên gói hoặc các gói.
* **pacman --remove package-name(s) hoặc pacman -R** - Loại bỏ các gói hoặc các gói.

Dưới đây là một số ví dụ với mô tả ngắn gọn:

* **pacman -Qi package-name(s)** - Hiển thị thông tin về một gói nhất định, bao gồm cả thông tin phụ thuộc, ngày cài đặt của gói, một liên kết đến các nguồn upsteam và các thông tin hữu ích khác.
* **pacman -Qo filename** – Xuất ra số phiên bản và tên của gói đang sở hữu một tập tin chỉ định.
* **pacman -Qs package-name(s)**- Tìm kiếm trong số các gói cài đặt phù hợp với package-name.
* **pacman -Qu** – Danh sách các gói phiên bản cũ được cài đặt mà đang cần phải cập nhật
* **pacman -Sy** - Triggers một database refresh, và đồng bộ hóa cơ sở dữ liệu địa phương với các cơ sở dữ liệu từ xa.
* **pacman -Su** - Kích hoạt một hệ thống cập nhật đầy đủ và download gói mới để nâng cấp hệ thống.
* **pacman -Sc** - Xóa bỏ cài đặt các gói từ các bộ nhớ cache và cố gắng để làm sạch bản cũ của cơ sở dữ liệu lưu trữ.
* **pacman -S --ignore package-name(s)** - Bỏ qua nâng cấp một gói chỉ định hoặc danh sách các gói
* **pacman -Rs** Loại bỏ một gói và phụ thuộc của nó, miễn là các phụ thuộc không cần thiết và không được cài đặt một cách rõ ràng bởi người sử dụng. Lệnh này là nghịch đảo của pacman -S.
* **pacman -Ru** Loại bỏ các gói không cần thiết.

**Tùy chọn cấu hình**  
Cấu hình cho pacman được định nghĩa trong tập tin /etc/pacman.conf, trong khi đường dẫn của các kho mirror được chứa trong /etc/pacman.d/mirrorlist. Danh sách mirror được tạo ra và được ưu tiên trong quá trình cài đặt và có thể bạn sẽ không cần phải thay đổi điều này  
  
Các tùy chọn trong /etc/pacman.conf vượt ra ngoài phạm vi của tài liệu này. Bạn có thể truy cập vào các trang hướng dẫn cho tập tin cấu hình này bằng lệnh **man pacman.conf**.  
Nếu bạn cần thêm một kho lưu trữ bên thứ ba bổ sung, làm như sau:

Mã:

vi /etc/pacman.conf

Mã:

[REPOSITORY-NAME]

Server = SERVER-LOCATION

Include = REPOSITORY-LIST

**The Arch Build System (**[**ABS**](https://mangmaytinh.net/tags/abs/)**)**  
Hệ thống xây dựng Arch cho phép người sử dụng để biên dịch và cài đặt phần mềm không có trong kho lưu trữ Arch. Hướng dẫn ngắn gọn này nêu các bước để xây dựng một gói sử dụng ABS.  
Tất cả các lệnh giải thích ở đây nên được dùng root trừ trường hợp quy định.  
Cài đặt framework abs và các gói cơ bản-devel:

Mã:

pacman -Sy abs base-devel

Chỉnh sửa /etc/abs.conf để cho dòng Repos chỉ ra các kho thích hợp. Lưu ý, các kho bắt đầu bằng kí tự (!) là bị vô hiệu hóa. Các dòng có thể trông giống như:

Mã:

REPOS=(core extra community !testing)

Để tạo ra một cây ABS địa phương trong /var/abs, chạy lệnh abs root. Bây giờ bạn có thể duyệt các thư mục /var/abs, mà chứa một đại diện của các bộ sưu tập gói với các thư mục đại diện cho mỗi kho, thể loại, và phần mềm.  
  
Arch khuyến cáo rằng bạn tạo ra một thư mục build ở một vị trí khác, chẳng hạn như ~ /abs/.  
Bắt đầu quá trình xây dựng bằng cách sao chép các file từ các cây ABS vào thư mục build của bạn:

Mã:

cp -r /var/abs/REPO/PACKAGE ~/abs

Thay đổi thư mục của gói:

Mã:

cd ~/abs/PACKAGE

Bạn có tùy chọn thay đổi các tập tin PKGBUILD. Có một chức năng xây dựng shell bạn có thể sử dụng để thêm các bản vá lỗi bổ sung cho các tập tin nếu bạn phải sửa đổi phần mềm hoặc xây dựng tiến trình. Đó là chức năng shell thường trông giống như:

Mã:

vi ~/abs/PACKAGE/PKGBUILD

Mã:

build() {

cd $startdir/src/$pkgname-$pkgver.orig

patch -Np1 -i

$startdir/src/${pkgname}\_${pkgver}-$\_patchlevel.diff || return 1

./configure --prefix=/usr

make || return 1

make install

}

Xây dựng gói phần mềm, sử dụng lệnh sau:

Mã:

makepkg -s

Lệnh makepkg tạo ra một gói có chứa thông tin phụ thuộc. Thực hiện lệnh sau:

Mã:

pacman -U PACKAGE.pkg.tar.xz

Hãy chắc chắn thay thế PACKAGE với tên gói đầy đủ chính xác như nó xuất hiện trong hệ thống tập tin. Arch bây giờ sẽ cài đặt các gói phần mềm và bất kỳ phụ thuộc yêu cầu.  
  
**Thêm thông tin về Pacman và ABS**  
Nếu bạn quan tâm đến việc học thêm về Arch và các công cụ quản lý gói của nó, tham khảo những nguồn bên ngoài cho các tài liệu được cung cấp bởi cộng đồng Arch:

* [The Arch Build System](https://wiki.archlinux.org/index.php/Arch_Build_System)
* [Pacman](https://wiki.archlinux.org/index.php/Pacman)

[**Gentoo**](https://mangmaytinh.net/tags/gentoo/)**Linux Package Management**  
Gentoo cung cấp toàn bộ hệ điều hành của nó trong định dạng nguồn.  
Giống như Arch, các dự án Gentoo sản xuất phiên bản mới của Gentoo Linux trên một chu kỳ phát hành  
Chúng tôi khuyến khích bạn cài đặt "gentoolkit" để cung cấp các công cụ quản lý gói bổ sung, chẳng hạn như equery. Cài đặt gói này bằng lệnh sau đây:

Mã:

emerge app-portage/gentoolkit

Emerge/Portage Commands

* **emerge --sync** - Cập nhật các bản sao địa phương của cây portage, để hệ thống địa phương của bạn có thể tải về và cài đặt phiên bản mới nhất của phần mềm
* **emerge --update --deep world**- Kiểm tra và cập nhật tất cả các gói trên hệ thống lên phiên bản mới nhất.
* **emerge --search keyword or emerge -s keyword** - tìm kiếm tên của tất cả các gói cho các từ khóa nhất định. Lệnh này chấp nhận một biểu thức thường xuyên như là đối số từ khóa.
* **emerge --searchdoc keyword or emerge -S keyword** - Tìm kiếm các mô tả đầy đủ cho một từ khóa nhất định. Lệnh này chấp nhận một biểu thức thường xuyên như là đối số từ khóa.
* **emerge package-name(s)** - Cài đặt các gói hoặc gói cụ thể.  
  **emerge -u package-name(s)** - Cập nhật các gói cụ thể lên phiên bản mới nhất. Sử dụng cờ -uD cập nhật phụ thuộc.
* **emerge --depclean package-name(s) or emerge -c package-name(s)** - Loại bỏ các gói hoặc gói cụ thể.  
  **emerge --depclean**- Loại bỏ các gói. Điều này có nghĩa là loại bỏ tất cả các gói không được cài đặt một cách rõ ràng và không phải là phụ thuộc vào bất kỳ phần mềm cụ thể. Chúng tôi khuyên bạn nên chạy nó với tùy chọn --pretend trước khi chạy lệnh này trên một hệ thống.
* **emerge -evp --deep world** - Liệt kê tất cả các gói được cài đặt hiện tại trên hệ thống.
* **equery depends package-name(s)** - Liệt kê tất cả các gói phụ thuộc vàaijmột gói cụ thể.
* **equery files package-name(s)** - Liệt kê tất cả các tập tin sở hữu bởi một gói
* **equery belongs filename** – Liệt kê danh sách các gói sở hữu một tập tin cụ thể

**USE Flags**  
Portage cũng có thể cài đặt các biến thể khác của một gói với cờ USE, cho phép người sử dụng có thể kích hoạt hỗ trợ một tùy chọn đặc biệt liên quan đến gói đó. Để khám phá cờ USE có sẵn cho một gói nhất định, ban hành các lệnh sau đây:

Mã:

equery uses package-name

Các lệnh equery phụ thuộc vào gói gentoolkit. Điều này sẽ cung cấp thông tin về những USE flag có sẵn và đã được cài đặt. Để xác định USE thêm:

Mã:

echo "package-name USE-flags" >> /etc/portage/package.use

emerge package-name

Điều này sẽ cài đặt gói cụ thể với các tùy chọn thích hợp được kích hoạt.