maketecheasier

How to Setup Local Repository in Ubuntu

Derrik Diener 15th Nov 2016

Have you ever wanted to save bandwidth when updating your Ubuntu computers? It's very possible by creating a local APT repository server for your local LAN using Apache and some standard tools. This method allows for the ability to host any Debian package file you desire locally to make it easier to distribute.

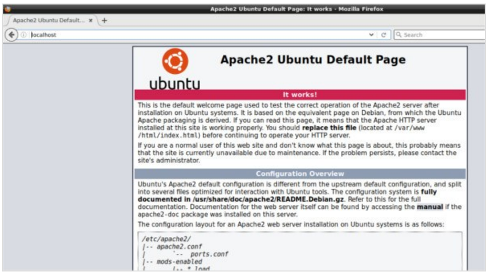
There are multiple reasons to have a local repository. The main reason would be to capture large updates downstream and re-distribute them via a local repository to save on Internet usage. Other reasons someone might want to do this would be to more easily distribute software that isn't found on the default Ubuntu servers.

So, how do you get it installed? Let's find out!

Installing Apache

For the local repository to work, an Apache server will need to be set up on the host system. Open up a terminal and enter the following command:

```
sudo apt-get install apache2
```


Apache2 Ubuntu Default Page: It works - Mozilla Firefox
Apache2 Ubuntu Default...
localhost
Search
ubuntu

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
```

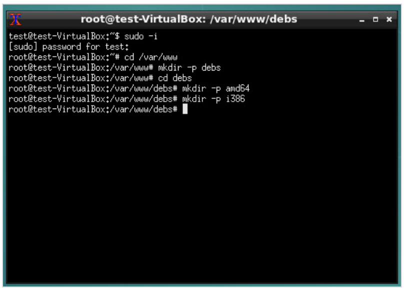
This will set up a website on Ubuntu and the folder structure that it needs to function. It can be tested out in any web browser by going to "http://localhost" or the LAN IP address of the machine using it (http://192.168.1.XX).

The Apache server folder structure lies entirely in "/var/www/." The default website page that the server sets up is in "/var/www/html/" and is labeled as "index." Setting up a local repository has little use for this default Apache page. However, it may be a good idea to make a start page to advertise ways LAN users can connect to the local repository.

Configuring Directories

The local Ubuntu package repository will not work without the correct folder structure. The Debian package repository directory structure is necessary here. In a terminal, do the following:

```
sudo -i
```



```
root@test-VirtualBox: /var/www/debs
test@test-VirtualBox:~$ sudo -i
[sudo] password for test:
root@test-VirtualBox:~# cd /var/www
root@test-VirtualBox:/var/www# mkdir -p debs
root@test-VirtualBox:/var/www# cd debs
root@test-VirtualBox:/var/www/debs# mkdir -p amd64
root@test-VirtualBox:/var/www/debs# mkdir -p i386
root@test-VirtualBox:/var/www/debs# 
```

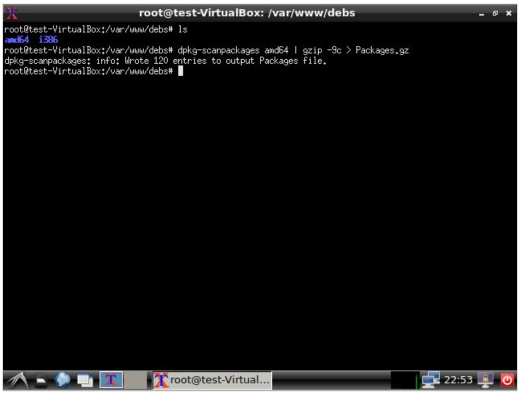
Now that the user has root permission, start off by entering the correct directory with the `cd` command.

```
cd /var/www/
```

The first part of the package repository structure is the "deb" folder. Create it with the `mkdi` r command.

```
mkdir -p debs
cd debs
mkdir -p amd64
mkdir -p i386
```

Creating the APT catalog



```
root@test-VirtualBox:/var/www/debs# ls
amd64  i386
root@test-VirtualBox:/var/www/debs# dpkg-scanpackages amd64 | gzip -9c > Packages.gz
dpkg-scanpackages: info: Wrote 120 entries to output Packages file.
root@test-VirtualBox:/var/www/debs#
```

The folders are set up. Now it is time to configure the catalog for APT to use. First, place any and all .Deb package files you'd like to keep in either the "/var/www/debs/amd64" folder or the "/var/debs/i386" folder (for 32 bit packages).

```
dpkg-scanpackages amd64 | gzip -9c > Packages.gz
```

or for 32bit

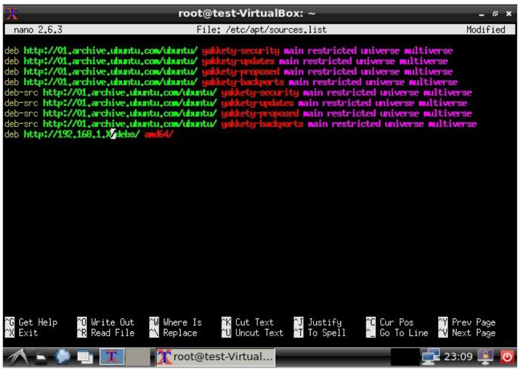
```
dpkg-scanpackages i386 | gzip -9c > Packages.gz
```

Setting up clients

With everything set up, the APT repository is no different than any other repository that a user would find online. This makes it very simple and easy to add to Ubuntu.

Inside the sources file, add your LAN APT repository using your APT servers LAN IP address. This is found with `ip addr` show in the terminal. Look for "inet 192.168.." under the network adapter on the system communicating with the Internet. Alternatively, log into your router and find the IP address it has given the machine hosting the APT repo server.

Note: to make things easier on laptops, consider port forwarding the IP address of the APT repo host machine outside of LAN to the Internet on port 80 and using that in place of a local IP address like the tutorial says. This will ensure that updates will not break when a computer leaves the LAN for a brief time.



```
 nano 2.6.3                  File: /etc/apt/sources.list                  Modified

deb http://01.archive.ubuntu.com/ubuntu/ yakkety-security main restricted universe multiverse
deb http://01.archive.ubuntu.com/ubuntu/ yakkety-updates main restricted universe multiverse
deb http://01.archive.ubuntu.com/ubuntu/ yakkety-proposed main restricted universe multiverse
deb http://01.archive.ubuntu.com/ubuntu/ yakkety-backports main restricted universe multiverse
deb-src http://01.archive.ubuntu.com/ubuntu/ yakkety-security main restricted universe multiverse
deb-src http://01.archive.ubuntu.com/ubuntu/ yakkety-updates main restricted universe multiverse
deb-src http://01.archive.ubuntu.com/ubuntu/ yakkety-proposed main restricted universe multiverse
deb-src http://01.archive.ubuntu.com/ubuntu/ yakkety-backports main restricted universe multiverse
deb http://192.168.1.X/debs/ amd64/
```

With the internal IP address known, enter the following in the terminal:

```
sudo nano /etc/apt/sources.list
```

This will allow the user to edit the list of servers that Ubuntu pings for updates. Add the repository to the list and save it.

```
deb http://192.168.1.X/debs/ amd64/
```

or for 32bit

```
deb http://192.168.1.X/debs/ i386/
```

Press "Ctrl + o" to save the changes once done. Finally, update Ubuntu by doing `sudo apt-update`.

Note: replace X with whatever internal IP address your router has given out. Alternatively, change 192.168.1.X with the external IP address so that your APT server is ping-able when not at home.

Conclusion

Now that everything is set up, any user with the repository in the "sources.list" file will be able to easily get packages from the local APT cache as if they were regular updates. No other special configurations required, and bandwidth savings can begin.

This method is the superior way to go as it makes it possible to host packages locally for bandwidth sake.The other positive is that there is very little configuration to do, and end-users will not know any better.

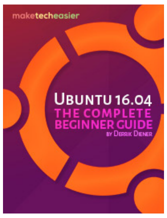
Would you host your own APT cache server on Ubuntu to save bandwidth? Tell us below!

Is this article useful? Yes No

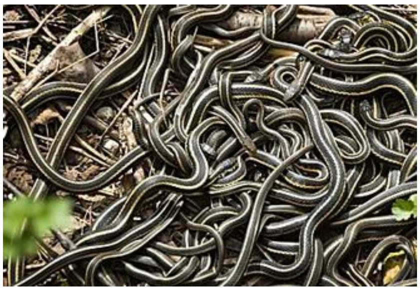
Related Ebooks

The Beginner's Guide to KDE

The Beginner's Guide to Linux Command Line



The Complete Beginner's Guide to Ubuntu 16.04



2 comments

Rick

"Note: to make things easier on laptops, consider port forwarding the IP address of the APT repo host machine outside of LAN to the Internet on port 80 and using that in place of a local IP address like the tutorial says. This will ensure that updates will not break when a computer leaves the LAN for a brief time."

This is very bad advice. Firstly, it's unnecessary. If the laptop is only going to be away from the LAN for a "brief time", it's unnecessary to do any updates to the software on it…just wait until you're back at the LAN to do the updates. Plus, since you're not at home to add newly updated software to your local repository, it's pointless to have the repository accessible via the Internet, since it won't get used anyway.

Secondly, opening standard ports like 80 to the Internet is simply asking for trouble. 'Bots are constantly scanning IP addresses on the Internet for standard ports that are open. Finding such an open port means that they will start hammering the IP address, looking for a way in and a way to exploit any vulnerabilities in the connected software, which can slow down your own use of your Internet connection.

Thirdly, the Terms Of Service of most residential ISP's prohibit the use of servers, and a lot of times the ISP will block all the standard ports to prevent their customers from doing so. Thus, opening the port is useless, since the ISP's block will prevent you from accessing it. And, since you're violating the TOS, the ISP is likely to terminate your service.

Setting up a local repository is a good idea if you have multiple computers to maintain; I've thought of doing it myself. But exposing your computer to the Internet is an unnecessary risk that could end up causing you more trouble than help.

Nov 16, 2016 at 1:55 pm

Luis Reinier

I have a problem with that method. When I finished all instruction I run command "sudo apt update" with console (Ubuntu 16.04 LTS), and the output is a messages that say something like: The file:/route_my_repository repository does not have a publication file.

Jun 21, 2017 at 6:54 am

Comments are closed.