How to Install OpenERP Odoo 8 on Ubuntu Server 14.04 LTS

Introduction

Welcome to the latest of our very popular ~~OpenERP~~ Odoo installation "How Tos".

The new release of Odoo 8.0 is a major upgrade introducing a great many new features and a new name.

Odoo 8.0 is not only better looking and easier to use, it also brings many improvements to the existing feature-set and adds a number of brand new features which extend the scope of the business needs covered by Odoo. Ecommerce, CMS, Integrated BI...

Rather than me blathering on about what's new, you can simply just go and read the release notes here.

The How To

Following that introduction, I bet you can't wait to get your hands dirty...

Just one thing before we start: You can simply download a .deb (for Debian/Ubuntu type systems) or a. rpm (Redhat/CentOS) package of OpenERP and install that. Unfortunately that approach doesn't provide us (Libertus Solutions) with enough fine-grained control over where things get installed, and it restricts our flexibility to modify & customise, hence I prefer to do it a slightly more manual way (this install process below should only take about 10-15 minutes once the host machine has been built).

This time, rather than using a source tarball as the basis for installation we are going to take the code straight from the Odoo 8.0 branch on Github. This should help when it comes to installing updates and bug fixes in the future by being able to issue a `git`

`pull` command to update the code. *Bear in mind before doing a pull request you should always have backups and you may need to update your Odoo database(s) as well.*

So without further ado here we go:

Step 1. Build your server

I install just the bare minimum from the install routine (you may want to install the `openssh-server` during the install procedure or install subsequently depending on your needs).

After the server has restarted for the first time I install the `openssh-server` package (so we can connect to it remotely) and ~~`denyhosts`~~ to add a degree of brute-force attack protection. There are other protection applications available: I'm not saying this one is the best, but it's one that works and is easy to configure and manage. If you don't already, it's also worth looking at setting up key-based ssh access, rather than relying on passwords. This can also help to limit the potential of brute-force attacks. [NB: This isn't a How To on securing your server...]

```
sudo apt-get install openssh-server denyhosts
```

UPDATE: Note that it seems `denyhosts` is no longer being maintained and is not in the main Ubuntu repository any more. I'm aware of a possibly suitable alternative called `fail2ban` but have not used it yet; do you own research. Thanks to Rami for the cluebat!

UPDATE2: Thanks to Paul for the pointer. I have added `python-unicodecsv` to the list of dependencies. Apparently this is required to correctly restore backups.

Now make sure your server has all the latest versions & patches by doing an update:

```
sudo apt-get update
sudo apt-get dist-upgrade
```

Although not always essential it's probably a good idea to reboot your server now and make sure it all comes back up and you can login via ssh.

Now we're ready to start the Odoo install.

Step 2. Create the Odoo user that will own and run the application

```
sudo adduser --system --home=/opt/odoo --group odoo
```

This is a "system" user. It is there to own and run the application, it isn't supposed to be a person type user with a login etc. In Ubuntu, a system user gets a UID below 1000, has no shell (it's actually `/bin/false`) and has logins disabled. Note that I've specified a "home" of `/opt/odoo`, this is where the OpenERP server code will reside and is created automatically by the command above. The location of the server code is your choice of course, but be aware that some of the instructions and configuration files below may need to be altered if you decide to install to a different location.

[Note: If you want to run multiple versions of Odoo/OpenERP on the same server, the way I do it is to create multiple users with the correct version number as part of the name, e.g. openerp70, openerp61 etc. If you also use this when creating the Postgres users too, you can have full separation of systems on the same server. I also use similarly named home directories, e.g. /opt/odoo80, /opt/openerp70, /opt/openerp61 and config and start-up/shutdown files. You will also need to configure different ports for each instance or else only the first will start.]

A question I have been asked a few times is how to run the Odoo server as the odoo system user from the command line if it has no shell. This can be done quite easily:

```
sudo su - odoo -s /bin/bash
```

This will `su` your current terminal login to the odoo user (the "`-`" between `su` and `odoo` is correct) and use the shell `/bin/bash`. When this command is run you will be in odoo's home directory: `/opt/odoo`.

When you have done what you need you can leave the odoo user's shell by typing `exit`.

Step 3. Install and configure the database server, PostgreSQL

```
sudo apt-get install postgresql
```

Then configure the Odoo user on postgres:

First change to the postgres user so we have the necessary privileges to configure the database.

```
sudo su - postgres
```

Now create a new database user. This is so Odoo has access rights to connect to PostgreSQL and to create and drop databases. Remember what your choice of password is here; you will need it later on:

```
createuser --createdb --username postgres --no-createrole --no-superuser --pwprompt odoo
Enter password for new role: ********
Enter it again: ********
```

Finally exit from the postgres user account:

```
exit
```

Step 4. Install the necessary Python libraries for the server

```
sudo apt-get install python-dateutil python-decorator python-docutils python-feedparser \
python-gdata python-gevent python-imaging python-jinja2 python-ldap python-libxslt1 python-lxml \
python-mako python-mock python-openid python-passlib python-psutil python-psycopg2 python-pybabel \
python-pychart python-pydot python-pyparsing python-pypdf python-reportlab python-requests \
python-simplejson python-tz python-unicodecsv python-unittest2 python-vatnumber python-vobject \
python-werkzeug python-xlwt python-yaml wkhtmltopdf
```

With that done, all the dependencies for installing Odoo 8.0 are now satisfied (note that there are some changes between this and the packages required for OpenERP 7.0).

UPDATE & NOTE: It has been pointed out to me that the Qweb templating engine in Odoo 8 is warning that the version of wkhtmltopdf is too old. It turns out that Ubuntu 14.04 packages version 0.9.9 of this library even though this is rather old. To update your Ubuntu server please follow the instructions on this page. Many thanks to Ruben Kannan for pointing this out :-).

UPDATE & NOTE II: Zak suggests an alternative method to get and install the most recent version of wkhtmltopdf in this comment below. Thanks Zak.

Step 5. Install the Odoo server

Install Git.

```
sudo apt-get install git
```

Switch to the Odoo user:

```
sudo su - odoo -s /bin/bash
```

Grab a copy of the most current Odoo 8 branch *(Note the "." at the end of this command!)*:

```
git clone https://www.github.com/odoo/odoo --depth 1 --branch 8.0 --single-branch .
```

(This might take a little while depending on the speed of your Internet connection.)

Note: Thanks to Ian Beardslee for the cluebat. Have now added `--depth 1` to the command so it only retrieves the latest version without all the history. The download is now quite a bit quicker.

Once it's finished exit from the odoo user: `exit`.

Step 6. Configuring the OpenERP application

The default configuration file for the server (`/opt/odoo/debian/openerp-server.conf`) is actually very minimal and will, with only a small change work fine so we'll copy that file to where we need it and change it's ownership and permissions:

```
sudo cp /opt/odoo/debian/openerp-server.conf /etc/odoo-server.conf
sudo chown odoo: /etc/odoo-server.conf
```

```
sudo chmod 640 /etc/odoo-server.conf
```

The above commands make the file owned and writeable only by the odoo user and group and only readable by odoo and root.

To allow the odoo server to run initially, you should only need to change two lines in this file. Toward to the top of the file change the line `db_password = False` to the same password you used back in step 3. Then modify the line `addons_path = /usr/lib/python2.7/dist-packages/openerp/addons` so that it reads `addons_path = /opt/odoo/addons` instead.

One other line we might as well add to the configuration file now, is to tell Odoo where to write its log file. To complement my suggested location below add the following line to the `odoo-server.conf` file:

```
logfile = /var/log/odoo/odoo-server.log
```

Use your favourite text editor here. I tend to use nano, e.g.

```
sudo nano /etc/odoo-server.conf
```

Once the configuration file is edited and saved, you can start the server just to check if it actually runs.

```
sudo su - odoo -s /bin/bash
/opt/odoo/openerp-server
```

If you end up with a few lines eventually saying OpenERP (Yes. The log still says OpenERP and not Odoo) is running and waiting for connections then you are all set.

If there are errors, you'll need to go back and find out where the problem is.

Otherwise simply enter `CTL+C` to stop the server and then `exit` to leave the openerp user account and go back to your own shell.

Step 7. Installing the boot script

For the final step we need to install a script which will be used to start-up and shut down the server automatically and also run the application as the correct user. There

is a script you can use in `/opt/odoo/debian/init` but this will need a few small modifications to work with the system installed the way I have described above. Here's a link to the one I've already modified for Odoo version 8.

Similar to the configuration file, you need to either copy it or paste the contents of this script to a file in `/etc/init.d/` and call it `odoo-server`. Once it is in the right place you will need to make it executable and owned by root:

```
sudo chmod 755 /etc/init.d/odoo-server
sudo chown root: /etc/init.d/odoo-server
```

In the configuration file there's an entry for the server's log file. We need to create that directory first so that the server has somewhere to log to and also we must make it writeable by the openerp user:

```
sudo mkdir /var/log/odoo
sudo chown odoo:root /var/log/odoo
```

Step 8. Testing the server

To start the Odoo server type:

```
sudo /etc/init.d/odoo-server start
```

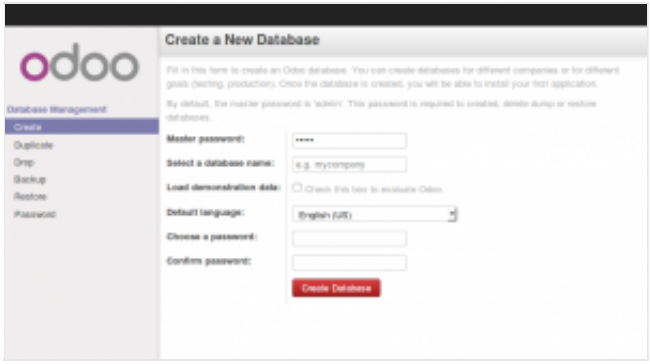
You should now be able to view the logfile and see that the server has started.

```
less /var/log/odoo/odoo-server.log
```

If there are any problems starting the server you need to go back and check. There's really no point ploughing on if the server doesn't start...

If the log file looks OK, now point your web browser at the domain or IP address of your Odoo server (or localhost if you are on the same machine) and use port 8069. The url will look something like this:

```
http://IP_or_domain.com:8069
```



Odoo 8 New Database

What you should see is a screen like this one (it is the Database Management Screen because you have no Odoo databases yet):

What I do recommend you do at this point is to change the super admin password to something nice and strong (Click the "Password" menu). By default this password is just "admin" and knowing that, a user can create, backup, restore and **drop** databases! This password is stored in *plain text* in the `/etc/odoo-server.conf` file; *hence why we restricted access to just odoo and root*. When you change and save the new password the `/etc/odoo-server.conf` file will be re-written and will have a lot more options in it.

Now it's time to make sure the server stops properly too:

```
sudo /etc/init.d/odoo-server stop
```

Check the log file again to make sure it has stopped and/or look at your server's process list.

Step 9. Automating Odoo startup and shutdown

If everything above seems to be working OK, the final step is make the script start and stop automatically with the Ubuntu Server. To do this type:

```
sudo update-rc.d odoo-server defaults
```

You can now try rebooting you server if you like. Odoo should be running by the time you log back in.

If you type `ps aux | grep odoo` you should see a line similar to this:

```
odoo 1491 0.1 10.6 207132 53596 ? Sl 22:23 0:02 python
/opt/odoo/openerp-server -c /etc/odoo-server.conf
```

Which shows that the server is running. And of course you can check the logfile or visit the server from your web browser too.

That's it! Next I would suggest you create a new database filling in the fields as desired. Once the database is initialised, you will be directed straight to the new main configuration screen which gives you a feel for the new User Interface in Odoo 8 and

shows you how easy it is to set up a basic system.

Tags: Odoo, OpenERP, PostgreSQL, Ubuntu

Odoo 8 Opening Screen

This entry was posted on Friday, September 19th, 2014 at 9:43 by Alan Lord and is filed under FLOSS in the news, Libertus, OpenERP, Ubuntu. You can follow any responses to this entry through the RSS 2.0 feed. You can leave a response, or trackback from your own site.