Gateway Server Configuration

The purpose of this document is to list the steps necessary to configure an Ubuntu server to be ready for Gateway application deployment.

Notes

1. An Ubuntu server has already been provisioned.
2. Single server hosts both application and database.
3. You can access the server (most commonly via SSH), and have `sudo` permission to perform system updates.

Server Configuration

Server access

SSH into the server

Perform system updates:

```
sudo apt-get update && sudo apt-get -y upgrade
```

Create `deploy` user (optional)

The steps and files associated with this document are based on the **`deploy`** user. This user is referenced in some configuration files (Nginx etc.). If you'd like to avoid updating those files with your user name, you can follow the steps below to create a `deploy` user.

1. Create user:

`sudo useradd -d /home/deploy -m deploy`

2. Set the password for `deploy` user:

`sudo passwd deploy`

Enter password and confirm it. This password will be required by RVM for the Ruby installation.

3. Open `/etc/passwd` and make sure the line for `deploy` has `/bin/bash` in the end.

4. Add the `deploy` user to `sudoers` as well.

Run `sudo visudo` and paste the following into the file: `deploy ALL=(ALL) NOPASSWD: ALL`

Save the file and exit.

Install required programs: Git, nodejs, Nginx, PostgreSQL, PostGIS, MDBTools

```
sudo apt-get install -y git nodejs nginx postgresql postgresql-contrib libpq-dev postgis postgresql-9.5-postgis-2.2 mdbtools
```

RVM and Ruby

1. Install RVM

- Login `deploy` user as superuser: `su - deploy`
- `gpg --keyserver hkp://keys.gnupg.net --recv-keys 409B6B1796C275462A1703113804BB82D39DC0E3`
- `\curl -sSL https://get.rvm.io | bash -s stable`

NOTE: This will install RVM into the **`deploy`** user's home directory. Logout and login again to load RVM into the deploy user's shell. Logout with **Ctrl+D** and login again with `su - deploy`.

2. Install Ruby:

`rvm install 2.2.0`

PostgreSQL and PostGIS configuration

1. If data need to be stored in a mounted volume (Optional)

Follow this article to configure PostgreSQL data in a mounted volume.

2. Create a new user (‘gateway’), it’ll prompt you for the password:

```
sudo -u postgres createuser -P gateway
```

3. Create database ‘gateway_prod`

```
sudo -u postgres createdb -O gateway gateway_prod
```

4. Enable Gateway needed extensions on the database

```
sudo -u postgres psql -c "CREATE EXTENSION postgis; CREATE EXTENSION postgis_topology; CREATE EXTENSION
intarray" gateway_prod
```

Install RGeo

1. Install `aptitude`

```
sudo apt-get install aptitude
```

2. Install Geos and Proj

```
sudo aptitude install libgeos-dev libproj-dev
```

3. Create shared object in `/usr/lib`

```
sudo ln -s /usr/lib/x86_64-linux-gnu/libgeos-3.5.0.so /usr/lib/libgeos.so
```

Install GDAL/OGR (shapefile export)

```
sudo add-apt-repository ppa:ubuntugis/ppa && sudo apt-get update
```

`sudo apt-get install gdal-bin`

Gateway application first-time configuration

Application Configuration

1. Set server specific environment variables

   Add the following into (`~/.bashrc`), assuming it's a **production** server

   `export RAILS_ENV=production`

   `export RACK_ENV=production`

2. Clone Gateway from Github to server

   `git clone git@github.com:camsys/nymtc-gateway.git /home/deploy/gateway`

   This would download gateway code into `/home/deploy/gateway` (**NOTE:** `deploy` is the user being used throughout this document.)

3. Bundler

- Enter `gateway` folder
- Install bundler: `gem install bundler`
- `bundle install --without test development`

4. application.yml

- `touch config/application.yml`
- Generate a new secret key via `rake secret`
- Copy **`./deployment_docs/application.yml`**
- Open `application.yml` and update its values as needed

5. database.yml

- `touch config/database.yml`
- copy **`deployment_docs/database.yml`** into it

6. Add two folders in `/tmp` for puma files: `tmp/pids` and `tmp/sockets`

Import data into database

Easiest way is to restore data from a backup.

- To backup data, use `pg_dump`, refer to this doc

- Given a dump file, run the following command to restore it into a new database

  NOTE: replace `dump_file_path` with real path

  ```
  sudo -u postgres pg_restore --verbose --no-owner --role=gateway --dbname=gateway_prod --jobs=8
  {dump_file_path}
  ```

Nginx configuration

1. Open Nginx configuration file:

   ```
   sudo nano /etc/nginx/sites-available/default
   ```

2. Replace with following file content:

   - if SSL is configured, then use: `deployment_docs/nginx_ssl.conf`
   - otherwise, use `deployment_docs/nginx.conf`
     - Note that: need to remove `config.force_ssl = true` in `config/environments/production.rb`

3. **Review**: you might want to check the configuration to replace the domain name etc as needed.

Start Gateway

1. In gateway application directory, run `Rake assets:precompile`
2. Start puma: `bundle exec puma -C config/puma.rb`
3. `sh deploy/restart`

Future Deployment

Re-deployment after code changes

1. go to `gateway` directory: `cd gateway`
2. run `git pull -r origin master`
3. if there is database schema change, run `rake db:migrate`
4. run other rake tasks if any
5. `bundle`
6. `Rake assets:precompile`
7. run `sh deploy/restart`

After Server Reboot

1. Log in as `deploy` user.
2. go to `gateway` directory: `cd gateway`
3. run `sh deploy/restart`