Junior Devops/Sysadmin at Concucorp

Author: Ladislau Felisbino
e-mail: ladis29@gmail.com
phone: +55 (48) 99833-5848

This project consists of an pure ansible code that automates the initial installation and configuration of Drupal and CiviCRM over NGINX and their respective databases in an AWS instance.

Getting Started

The challenge was to do a secure installation of Drupal and CiviCRM on an AWS instance with SSL certificate using letsencrypt.

The server should use NGINX and ve developed on an Ubuntu 16.04 distribution and have Drush installed.

Prerequisites(Softwares)

- Ubuntu 16.04
- Nginx
- MySQL Server
- PHP Group
    - php7.1-fpm
    - php7.1-common
    - php7.1-mbstring
    - php7.1-xmlrpc
    - php7.1-soap
    - php7.1-gd
    - php7.1-xml
    - php7.1-intl
    - php7.1-mysql
    - php7.1-cli
    - php7.1-mcrypt
    - php7.1-ldap
    - php7.1-zip
    - php7.1-curl
- Ansible Packages
    - python-minimal
    - python-mysqldb
    - ansible
- Drush
- Drupal
- CiviCRM
- Certbot/Letsencrypt
- AWS Instance(t2.micro)

Developing and testing the environment

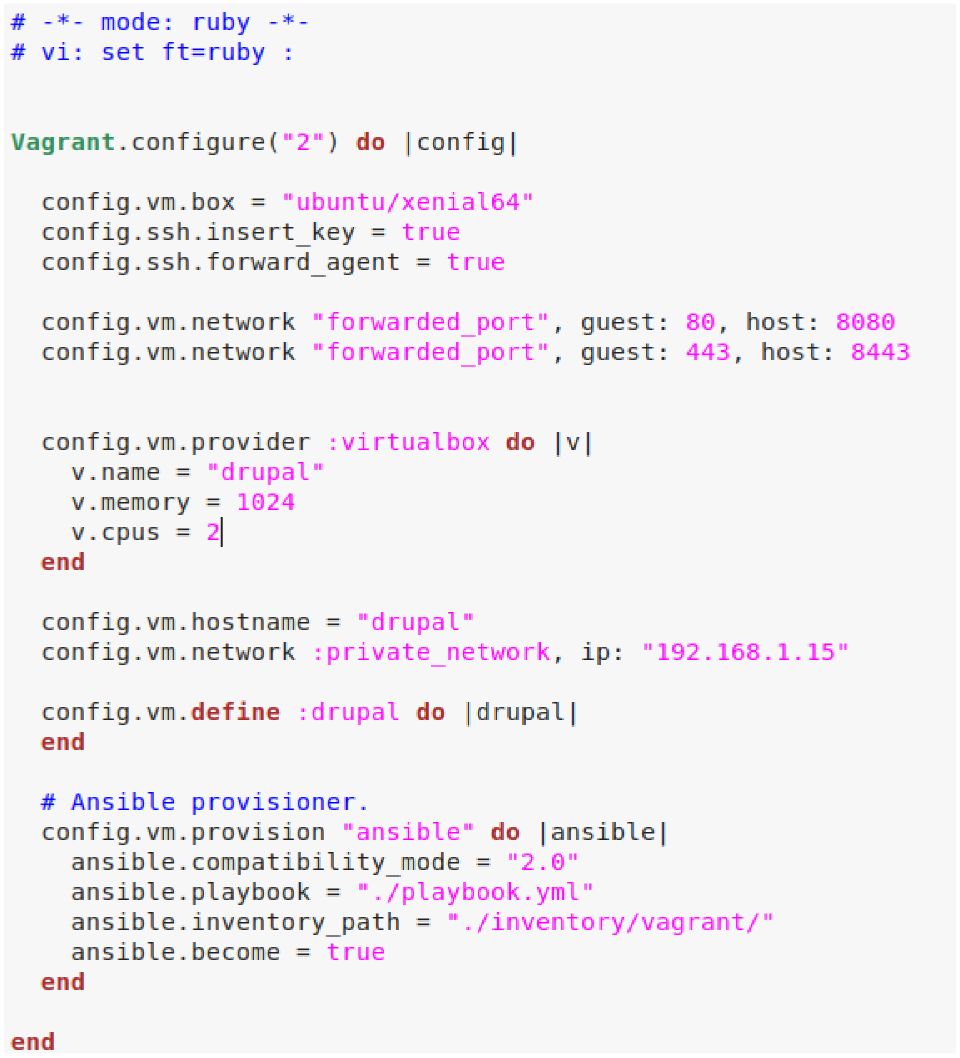
I started and ran each step of the requirements on a local Vagrant image by developing the ansible code and testing it locally, for, just after everything working as expected move up to the instance in the AWS.

The procedure for developing and configuring of the requested tasks occurred as follows:

First Step:

Choose the image to be used and create the script to run the chosen virtual machine in vagrant. By the confidence acquired in the author of the vagrant image during other projects I chose the image "**geerlingguy / ubuntu1604**" to run the vagrant virtual machine.

Below follows the vagrantfile used to create the virtual machine in the vagrant:



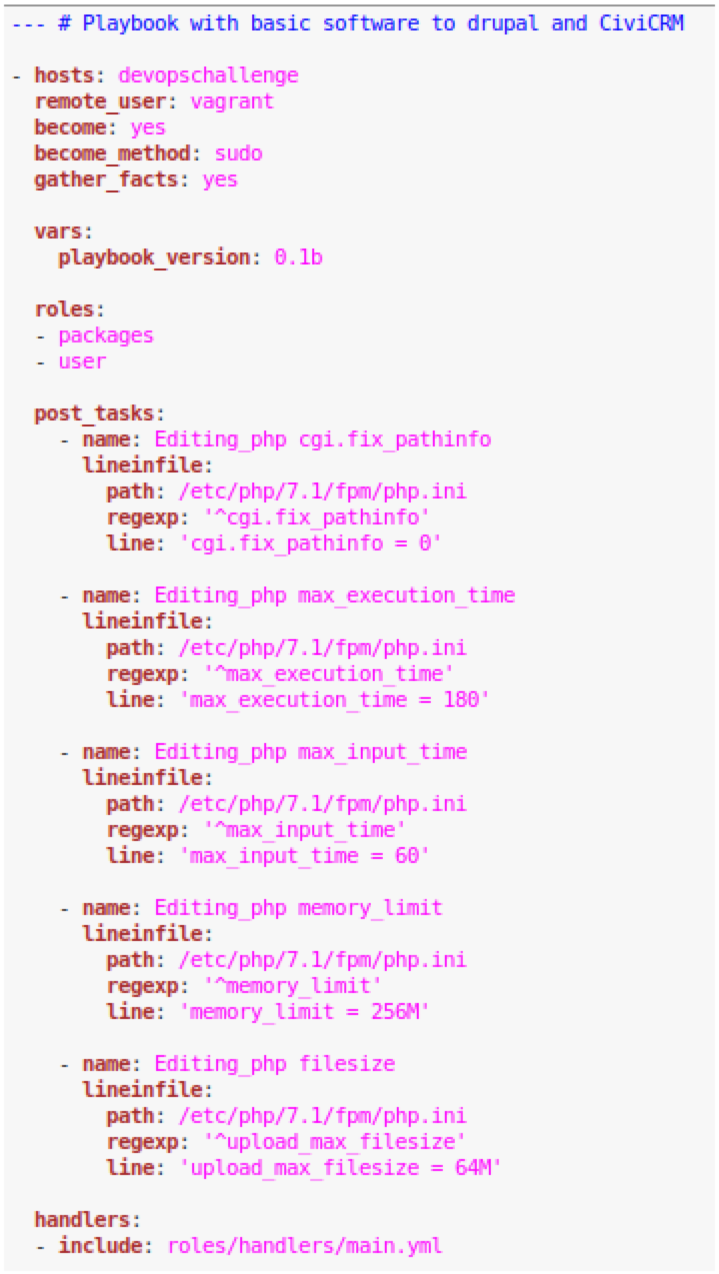
```ruby
# -*- mode: ruby -*-
# vi: set ft=ruby :


Vagrant.configure("2") do |config|

  config.vm.box = "ubuntu/xenial64"
  config.ssh.insert_key = true
  config.ssh.forward_agent = true

  config.vm.network "forwarded_port", guest: 80, host: 8080
  config.vm.network "forwarded_port", guest: 443, host: 8443


  config.vm.provider :virtualbox do |v|
    v.name = "drupal"
    v.memory = 1024
    v.cpus = 2
  end

  config.vm.hostname = "drupal"
  config.vm.network :private_network, ip: "192.168.1.15"

  config.vm.define :drupal do |drupal|
  end

  # Ansible provisioner.
  config.vm.provision "ansible" do |ansible|
    ansible.compatibility_mode = "2.0"
    ansible.playbook = "./playbook.yml"
    ansible.inventory_path = "./inventory/vagrant/"
    ansible.become = true
  end

end
```

Second Step:

The next step in the challenge was to install the necessary software and dependencies that would be the basis for Drupal and CiviCRM to run properly.

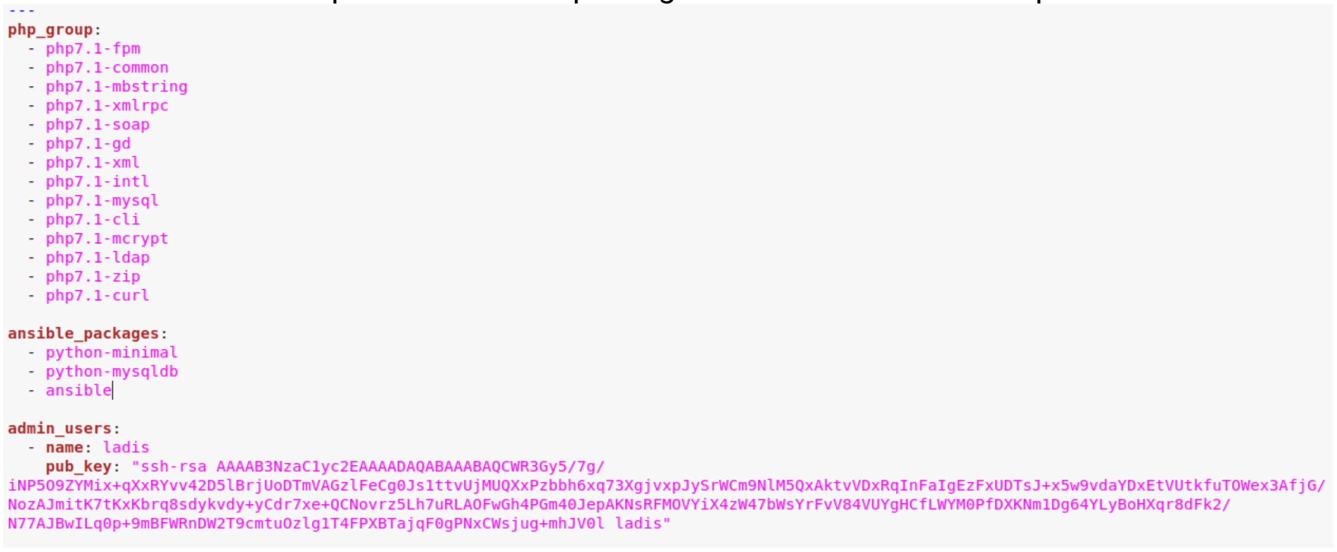
Following the best coding practices ansible I developed a script broken in roles and inventory to better organize the code.

The main part of the ansible code was the playbook and at this step it contained only the basics and initial packages, the configuration of php.ini according to the requirements of Drupal and the user with administrative access to the system:



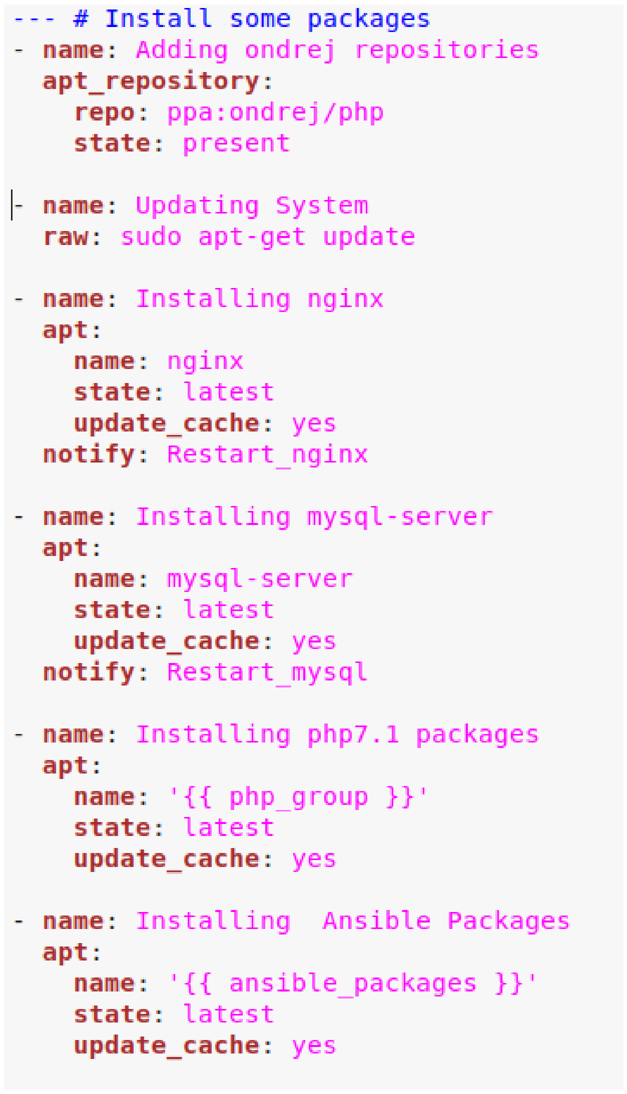
```yaml
--- # Playbook with basic software to drupal and CiviCRM

- hosts: devopschallenge
  remote_user: vagrant
  become: yes
  become_method: sudo
  gather_facts: yes

  vars:
    playbook_version: 0.1b

  roles:
  - packages
  - user

  post_tasks:
    - name: Editing_php cgi.fix_pathinfo
      lineinfile:
        path: /etc/php/7.1/fpm/php.ini
        regexp: '^cgi.fix_pathinfo'
        line: 'cgi.fix_pathinfo = 0'

    - name: Editing_php max_execution_time
      lineinfile:
        path: /etc/php/7.1/fpm/php.ini
        regexp: '^max_execution_time'
        line: 'max_execution_time = 180'

    - name: Editing_php max_input_time
      lineinfile:
        path: /etc/php/7.1/fpm/php.ini
        regexp: '^max_input_time'
        line: 'max_input_time = 60'

    - name: Editing_php memory_limit
      lineinfile:
        path: /etc/php/7.1/fpm/php.ini
        regexp: '^memory_limit'
        line: 'memory_limit = 256M'

    - name: Editing_php filesize
      lineinfile:
        path: /etc/php/7.1/fpm/php.ini
        regexp: '^upload_max_filesize'
        line: 'upload_max_filesize = 64M'

  handlers:
  - include: roles/handlers/main.yml
```

The variable script with the basic packages needed to fulfill the requested tasks:



```yaml
---
php_group:
  - php7.1-fpm
  - php7.1-common
  - php7.1-mbstring
  - php7.1-xmlrpc
  - php7.1-soap
  - php7.1-gd
  - php7.1-xml
  - php7.1-intl
  - php7.1-mysql
  - php7.1-cli
  - php7.1-mcrypt
  - php7.1-ldap
  - php7.1-zip
  - php7.1-curl

ansible_packages:
  - python-minimal
  - python-mysqldb
  - ansible

admin_users:
  - name: ladis
    pub_key: "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCWR3Gy5/7g/
iNP5O9ZYMix+qXxRYvv42D5lBrjUoDTmVAGzlFeCg0Js1ttvUjMUQXxPzbbh6xq73XgjvxpJySrWCm9NlM5QxAktvVDxRqInFaIgEzFxUDTsJ+x5w9vdaYDxEtVUtkfuTOWex3AfjG/
NozAJmitK7tKxKbrq8sdykvdy+yCdr7xe+QCNovrz5Lh7uRLAOFwGh4PGm40JepAKNsRFMOVYiX4zW47bWsYrFvV84VUYgHCfLWYM0PfDXKNm1Dg64YLyBoHXqr8dFk2/
N77AJBwILq0p+9mBFWRnDW2T9cmtuOzlg1T4FPXBTajqF0gPNxCWsjug+mhJV0l ladis"
```

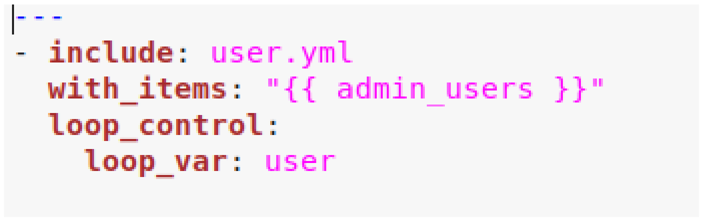
BETWEEN THE “ROLES” SCRIPTS
The Tasks Script:



```yaml
--- # Install some packages
- name: Adding ondrej repositories
  apt_repository:
    repo: ppa:ondrej/php
    state: present

- name: Updating System
  raw: sudo apt-get update

- name: Installing nginx
  apt:
    name: nginx
    state: latest
    update_cache: yes
  notify: Restart_nginx

- name: Installing mysql-server
  apt:
    name: mysql-server
    state: latest
    update_cache: yes
  notify: Restart_mysql

- name: Installing php7.1 packages
  apt:
    name: '{{ php_group }}'
    state: latest
    update_cache: yes

- name: Installing  Ansible Packages
  apt:
    name: '{{ ansible_packages }}'
    state: latest
    update_cache: yes
```

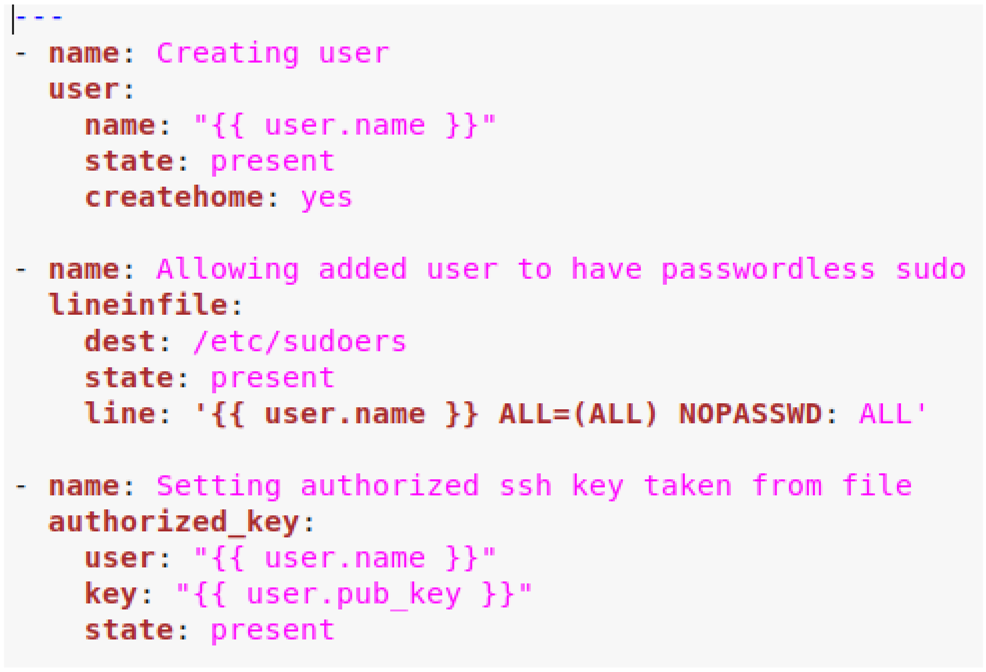
The script for creation of users with administrative access to the system, in this case with a single user, however I decided to keep the structure that I normally use to take advantage of the segmentation of the variable and the possibility of easy re-fitting of the same script.

Main



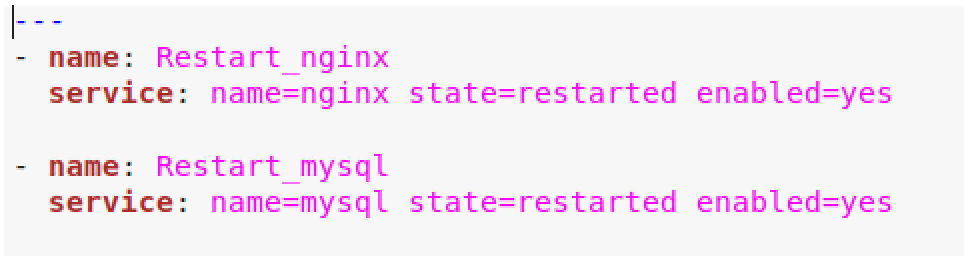
```yaml
---
- include: user.yml
  with_items: "{{ admin_users }}"
  loop_control:
    loop_var: user
```

Users



```yaml
---
- name: Creating user
  user:
    name: "{{ user.name }}"
    state: present
    createhome: yes

- name: Allowing added user to have passwordless sudo
  lineinfile:
    dest: /etc/sudoers
    state: present
    line: '{{ user.name }} ALL=(ALL) NOPASSWD: ALL'

- name: Setting authorized ssh key taken from file
  authorized_key:
    user: "{{ user.name }}"
    key: "{{ user.pub_key }}"
    state: present
```

And the handlers section that runs after all to ensure that some softwares will run properly.



```yaml
---
- name: Restart_nginx
  service: name=nginx state=restarted enabled=yes

- name: Restart_mysql
  service: name=mysql state=restarted enabled=yes
```

I ran the virtual machine and everything worked fine. Time to next step.

Third Step:

Now I should install Drupal, create it's database table, user and access it via Browser. Although I read many things about it, I had never worked with Drupal before and this was the part that was most pleasing me here.
I did some research and decided to follow the installation guidelines of the following site:

https://websiteforstudents.com/install-drupal-cms-on-ubuntu-16-04-lts-with-nginx-mariadb-php-7-1-and-lets-encrypt-ssl-tls/

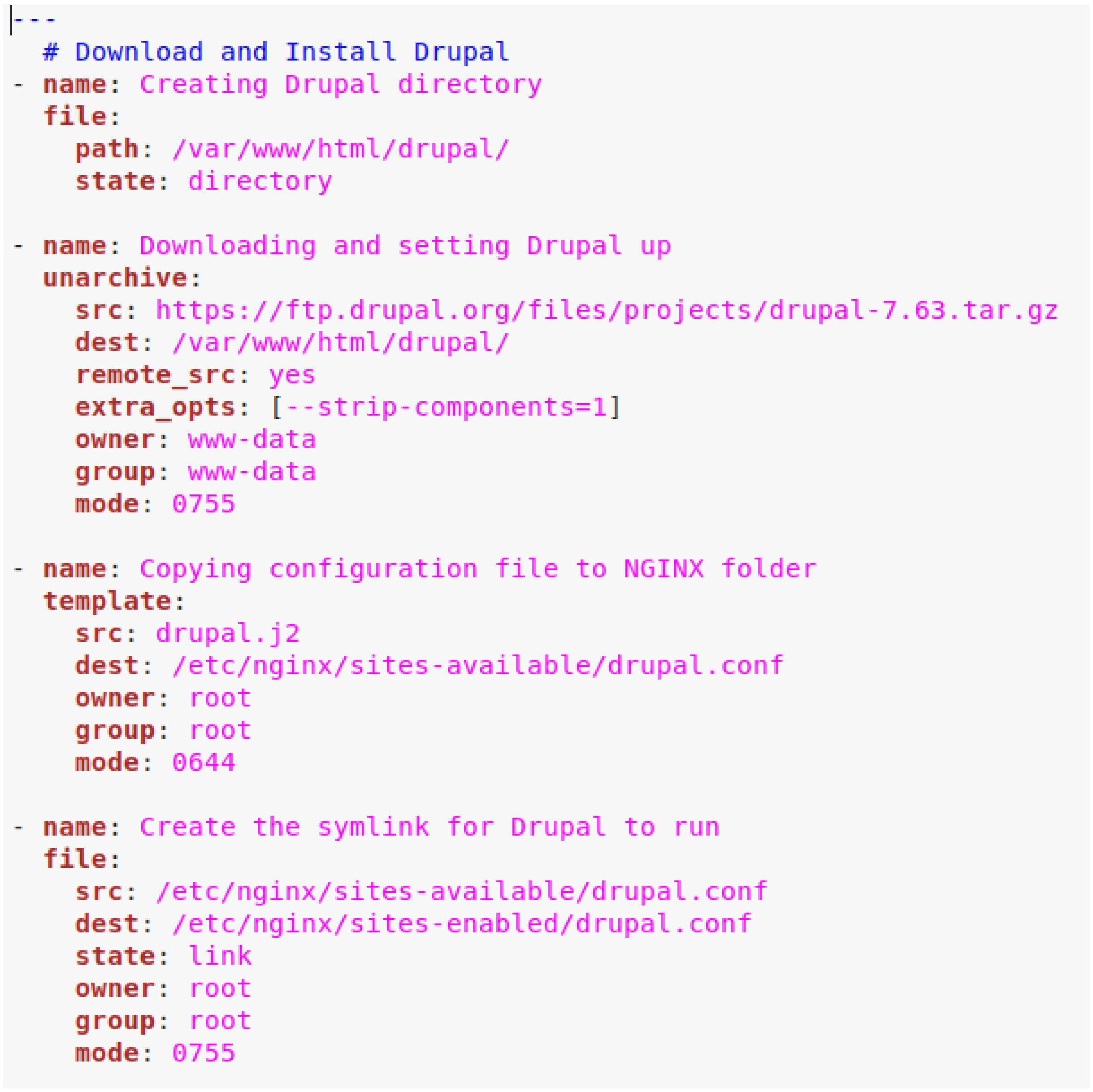
I've created another two roles in my playbook, now the "role" session of my playbook was as follows:

```
roles:
- packages
- user
- database
- drupal
```

I created a new "role" script for the configuration of the database and its respective user and decided that would use the same user for Drupal and CiviCRM databases.

```
---
- name: Create Drupal DB
  mysql_db:
    name: drupal
    state: present
    login_user: root
    login_password: root123

- name: Create a new DB user to drupal and civicrm and give him all access
  mysql_user:
    name: admin
    password: admin123
    priv: '*.*:ALL,GRANT'
    state: present
    login_user: root
    login_password: root123
```

Then I created another role” script for the Drupal configuration with the settings targeted at the site above suiting to my case :



```yaml
---
  # Download and Install Drupal
- name: Creating Drupal directory
  file:
    path: /var/www/html/drupal/
    state: directory

- name: Downloading and setting Drupal up
  unarchive:
    src: https://ftp.drupal.org/files/projects/drupal-7.63.tar.gz
    dest: /var/www/html/drupal/
    remote_src: yes
    extra_opts: [--strip-components=1]
    owner: www-data
    group: www-data
    mode: 0755

- name: Copying configuration file to NGINX folder
  template:
    src: drupal.j2
    dest: /etc/nginx/sites-available/drupal.conf
    owner: root
    group: root
    mode: 0644

- name: Create the symlink for Drupal to run
  file:
    src: /etc/nginx/sites-available/drupal.conf
    dest: /etc/nginx/sites-enabled/drupal.conf
    state: link
    owner: root
    group: root
    mode: 0755
```

And created the jinja2 template that would be copied to nginx to redirect traffic to the drupal index

```
server {
    listen 80;
    listen [::]:80;
    server_name juniordevopschallenge.com;


    location @rewrite {
                rewrite ^/(.*)$ /index.php?q=$1;
        }

    location ~ [^/]\.php(/|$) {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php7.1-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ ^/sites/.*/files/styles/ { # For Drupal >= 7
                try_files $uri @rewrite;
        }

    location ~ ^(/[a-z\-]+)?/system/files/ { # For Drupal >= 7
        try_files $uri /index.php?$query_string;
        }


}
```

Once more I ran the virtual machine and everything worked fine.

Time to next step.

Fourth Step:

The installation of the CiviCRM was the great mystery of the challenge, I have to confess that I had never heard of it, but although unknown(for me) it was not difficult to find reference material on the subject and I choose to follow the site above.

https://www.rosehosting.com/blog/deploy-civicrm-in-conjunction-with-drupal-on-an-ubuntu-14-04-vps/

The CiviCRM configuration is purely manual, I’ve automated only the package download in the appropriate directory
As usual, another item on roles section of playbook

```
roles:
- packages
- user
- database
- drupal
- civicrm
```

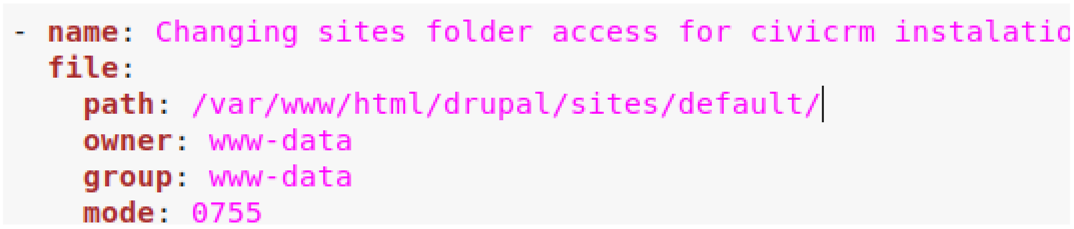
New “role” script for CiviCRM installation:

```
---
  - name: Adding civicrm directory to Drupal modules
    file:
      path: /var/www/html/drupal/sites/all/modules/civicrm
      state: directory

  - name: Downloading CiviCRM
    unarchive:
      src: https://download.civicrm.org/civicrm-5.9.1-drupal.tar.gz
      dest: /var/www/html/drupal/sites/all/modules/civicrm
      remote_src: yes
      extra_opts: [--strip-components=1]
      owner: www-data
      group: www-data
      mode: 0755
```

And new database table on databases role script:

```
- name: Create CiviCRM DB
  mysql_db:
    name: civicrm
    state: present
    login_user: root
    login_password: root123
```

To ensure the correct configuration of CiviCRM, it was necessary to ensure access permissions to the directory "/ var / www / html / drupal / sites / default /" :



```
- name: Changing sites folder access for civicrm instalatio
  file:
    path: /var/www/html/drupal/sites/default/
    owner: www-data
    group: www-data
    mode: 0755
```

And since this directory belongs to Drupal I decided that the change of directory access should be done in Drupal's own role script.

And again I was able to run the ansible script on the virtual machine and everything worked properly.

Fifth step:

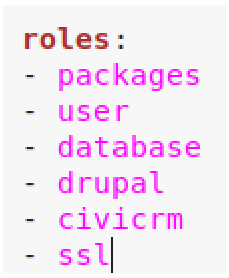
This was the time to install the digital certificate and make the site secure. However all attempts to install and run the certificate in vagrant were not successful, after following and studying several manuals for more than two days I enveloped this article:

https://letsencrypt.org/getting-started/

Where they explain that: "In order to get a certificate for your website's domain from Let's Encrypt, you have to demonstrate **control over the domain**"

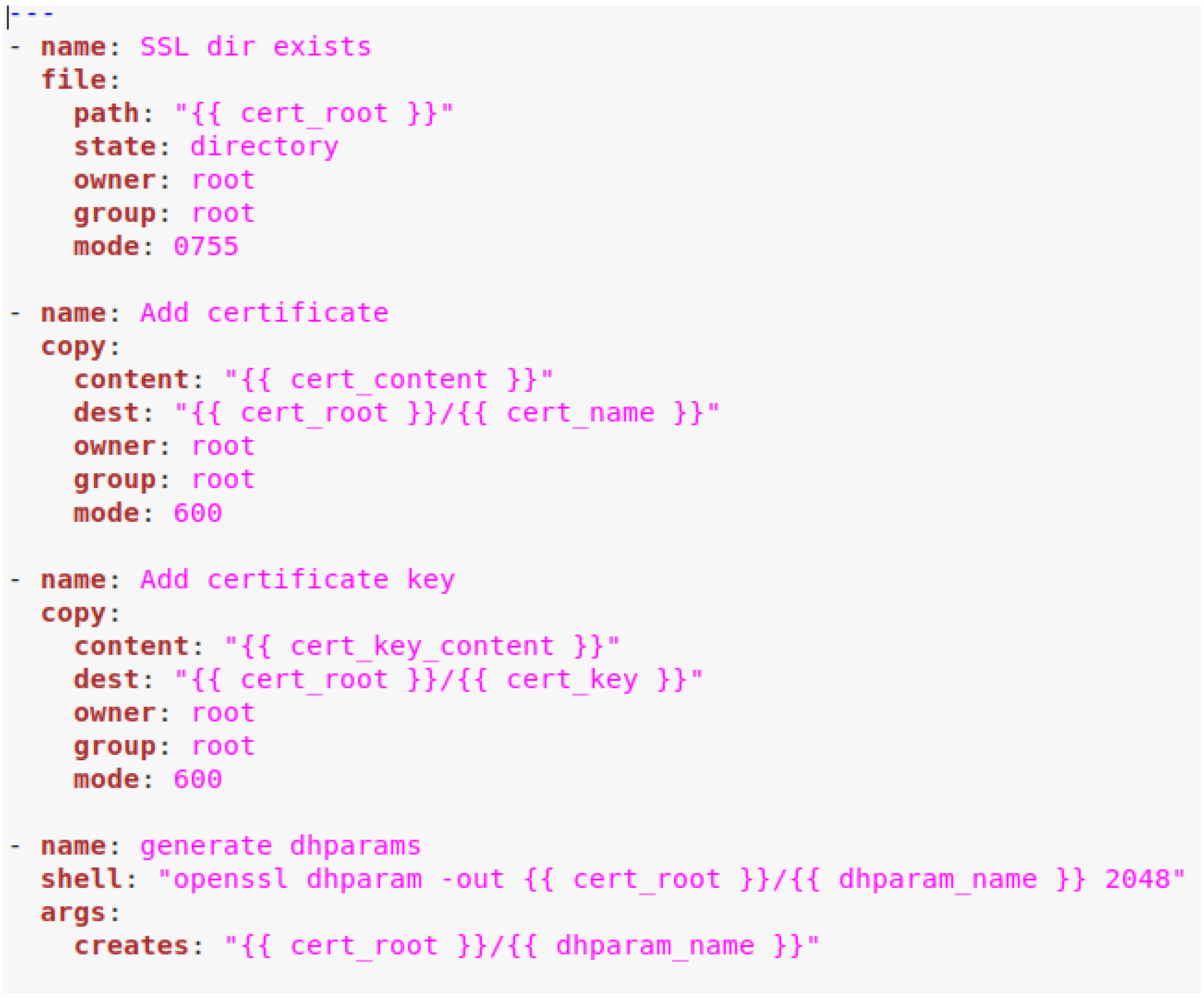
I am inexperienced in matters related to websites since I have always worked on managing infrastructure environments, but I understood that it would be impracticable to generate a valid digital certificate without having a properly registered domain, I then decided to perform a self-certification, which would not be valid as a letsencrypt certificate that was generated and validated by the certificate authority but was a way to accomplish the task even if partially.

So as done in the other steps I've created a new entry in the roles' session of my playbook.



```
roles:
- packages
- user
- database
- drupal
- civicrm
- ssl
```

New “role” script where I create the SSL dir, add two certificates(pem and key) locally generated and using them both generate dhparams.



```yaml
---
- name: SSL dir exists
  file:
    path: "{{ cert_root }}"
    state: directory
    owner: root
    group: root
    mode: 0755

- name: Add certificate
  copy:
    content: "{{ cert_content }}"
    dest: "{{ cert_root }}/{{ cert_name }}"
    owner: root
    group: root
    mode: 600

- name: Add certificate key
  copy:
    content: "{{ cert_key_content }}"
    dest: "{{ cert_root }}/{{ cert_key }}"
    owner: root
    group: root
    mode: 600

- name: generate dhparams
  shell: "openssl dhparam -out {{ cert_root }}/{{ dhparam_name }} 2048"
  args:
    creates: "{{ cert_root }}/{{ dhparam_name }}"
```

At this point I’ve created the “inventory” directory, inside of it the “group_vars” directory, inside group_vars I’ve created “devopschallenge” (my domain name) directory and put my “vars.yml” file there. Here I’ve created 6 new variables as follows:

```
domain_name: "juniordevopschallenge.com"

cert_root: /etc/ssl/
cert_name: juniordevopschallenge.com.pem
cert_key: juniordevopschallenge.com.key
dhparam_name: dhparam.pem


cert_content: |
  -----BEGIN CERTIFICATE-----
  MIIExDCCAqwCCQDOy/+05Se4vTANBgkqhkiG9w0BAQsFADAkMSIwIAYDVQQDDBlq
  dW5pb3JkZXZvcHNjaGFsbGVuZ2UuY29tMB4XDTE5MDEyMjIyNTk0NloXDTIwMDEy
  MjIyNTk0NlowJDEiMCAGA1UEAwwZanVuaW9yZGV2b3BzY2hhbGxlbmdlLmNvbTCC
  AiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBALp6lXUjaYCvkKWUhlJKK/tf
  uy9Q5U/QcHhVSNtB66ioe2YcfZVbfKiGCmSXvWF5QQ/GqENZy8iqUIGeGQBVjsWD
  NFjNgvhxlKrwJmHbviLPNYjbCXH08CTQDnbUIwExKlFHffVVDVtfcrCiAFQIHXZb
  zpJdeH066q+iKbmsMb095yz2xENHq8wyeBnihuT+pCZMnlqxS7qkbazwSIvK3mik
  dTOSAlIH6LPajlltvNUylarwyO8lP4VHu+nWfEw3U+Ydr+lBN+BDeX39CDYWTvnx
  3wQrThN7IfeoSqZ7RUs6bCU/PJiAszWZt6BNpiar578ULbVnYkhx3GOiyuj0kGIu
  ylhRllPe/7yb8l5T7lSxVt3sDiExh9uc7wg3cfNlHTXyEML+3NTjqB00ztarqX0m
  UTHjnwbX7YtL55fPHOLCloMmxR4T2eNRjjklJ0+64nRpzsy+2US2rkR8TQZj+cm9
  wPWlpfeLi5U48mmKQN/I8NM4VjlFzW3bQsbRpOgb0G6jxPDAwoGY4iAhIvM6b/m9
  bW/Q9xq36fzMG/JFvd6Nu58K3KW62MuKeEE9gcE+5ljr954PlmhkRN3bHLPVugwF
  T+4NfMufC3lYvzp20Yf2dAW6hgkI8q2/nFZLxMvlu8l3hOA0+II835ToREcQE/ZW
  jWtpjQAo//+Je1CXoQxXAgMBAAEwDQYJKoZIhvcNAQELBQADggIBAIdiX38JPs+N
  IocoWWDW+euELfmCdBe38QjZUpEHqKvVgqY6D1HWqknZk8saAmuk5LO8wJFXBi+P
  gyN4Q5MYM+JE+SNZ2Ac8KTzgr6J6jfLnMV7d6UKSw/MXhUR5fj5ehcZy1PXIefdz
  YRx7TkGpR+ChGQ6mFFiD+dgEmVOyZp8SLlXI+1wAVuyeDVpK8osqPN4BIAbBWOe+
  dhvvbJERU9USDoQsMzYXB9wQ9P27IuFUao/yS/DcbbmLm/N/6KsG050GG4v0yXRA
  7xAQ3++l7VglfLKRh/p2qn2sEYGQrwCohuxPX0yXdpVS60xLjJwBqEwUCMHQTCng
  Kt/C3c+ANIbVMwJy2Wnz5ylaqBZUwB3mDUii/RcDwd+1aNHXzwW80EayNkpBvluz
  0Fhuqipsy880+MCNAosp6NNJPYPlhWkixLRsc5C+Uz2q8FD85Y7xWaIBUBwqdg2x
  adomPvrRRWNBvq1T61j+yRp38lGBRp0jgG8eJz2/WhiN04FKco9aQQ/vfdeMy5SZ
  w81xR7FKY96GK1xDaTN0mscXs1YUNCxniCznr84abKtfSDIskvwjhJYVlsvtIhUz
  lkLzOKl6ratlb+6ZIR5H5y68ioDBJZBLQcVnotKzZcIfdEgEyouLzWubmshg+7Rn
  RXek+7eTJOP9JyMqeHIYkJ1C7PDyM8RT
  -----END CERTIFICATE-----

cert_key_content: |
  -----BEGIN RSA PRIVATE KEY-----
  MIIJKQIBAAKCAgEAunqVdSNpgK+QpZSHUkor+l+7L1DlT9BweFVI20HrqKh7Zhx9
  lVt8qIYKZJe9YXlBD8aoQlnLyKpQgZ4ZAFWOxYM0WM2C+HHUqvAmYdu+Is8liNsJ
  cfTwJNAOdtQjATEqUUd99VUNW19ysKIAVAgddlvOkll4fTrqr6IpuawxvT3nLPbE
  Q0erzDJ4GeKG5P6kJkyeWrFLuqRtrPBIi8reaKR1M5ICUgfos9qPWW281TLVqvDI
  7yU/hUe76dZ8TDdT5h2v7UE34EN5ff0INhZO+fHfBCtOE3sh96hKpntFSzpsJT88
  mICzNZm3oE2mJqvnvxQttWdiSHHcY6LK6PSQYi7KWFGXU97/vJvyXlPuVLFW3ewO
  ITGH25zvCDdx82UdNfIQwv7clOOoHTTOlqupfSZRMeOfBtfti0vnl88c4sLWgybF
  HhPZ41GOOSUnT7ridGnOzL7ZRLauRHxNBmP5yb3A9aWl94uLlTjyaYpA38jw0zhW
  OUXNbdtCxtGk6BvQbqPE8MDCgZjiICEi8zpv+bltb9D3Grfp/Mwb8kW93o27nwrc
  pbrYy4p4QT2BwT7mWOv3ng+WaGRE3dscs9W6DAVP7g18y58LeVi/OnbRh/Z0BbqG
  CQjyrb+cVkvEy+W7yXeE4DT4gjzflOhERxAT9laNa2mNACj//4l7UJehDFcCAwEA
  AQKCAgBSJBUZmBOs/6izhwlkjg95lt2ZJgUcdzBTkR2alxr7G9vfSsV2uOncQc7q
  KHzfJs4l6NfNcwx0w7Dap41TvFw7XGP+iegbD+khss71ZBoIs0Vdlz492Cq/zHXS
  mfchW0rMqrUtD93mVdDrRTDn0vtHW4FOr6WXZBBkdQX0J4lxVocyIzCogjWGqOSG
  YmYREKmlxSIryNUWzb8R4nXSPfZiGa8WnEYxZBJ4xtlGBzSapN3BgnbAAKROTR7V
  DgmKeQSX5JPN6mB2hMJLYN3xFTqMlqIp7/lgU2jO8m6PjtUjIgTpE0XZ1zcZcTCV
  iB6cAjT+0iCgO33rclwk9lxx6Oh/1SpT5+Qu19XXEpfo2NLU6o7zeBdkBjaLlamI
  r8n9+emVkwbWsU4BqeJXErdOLx9MLqZUdqAmH7RKogxpfsxd6e7gK/BDi/900lUl
  ydSlmUAv6fVNAX1Gh5UuX5Xesjx8XMts2JX+7pG8pVqlAy+F9pawGdIk9aWay71X
  iJxXxAUlxoWT74dr6zazvRc/dbK5Pcu0dDdBHXqs6DzL/mZieA09dzHJARgY97Fb
  1EslTHQB3GQdgMsFMyRld+HT+v7vPIF1K8shnXc084t+E14kVDMLhuhq4P7o+BjW
  UfaYymgIpYTI7MBrquZtUt1F1E5x9jT9AktkI7KuBN0VlMXCAQKCAQEA92x/bYzz
  KX3/bdS75BUT4RDl3ITYm5uRI1OTyCa4Qho6vemw1dacxzcFMokhJWMSjpstiuYq
  u86tmE0sIhKLZTG3y2OHn9N5h9UjMogyAMV0yTJGr0aUOM/3selaBuFaKrKVpSad
  VOG7GdzcVKpdyHv9EW3yOoYzcmPxzKwjkKUQQo5ccVFIwn5yhgOaYEEk/nMEX7R6
  X85UR0B4bPn2bb7/6LqlSksOKxDUmAqkOi6COo1DDLpBMRgZ3sWz0jDx0h/irWXL
  g8DwyBT1ALTPPwWADqmNAtuN9u94565v4kJ5SM3R9hRjKe27gr5HNtcXOCJ6ZWmK
  NSKycAKWu/7HFwKCAQEAwPFLRIXm3PsZA0Zcccls9rOBsN9EIy81QtOfD2PMHGHf
  dZgs3ob9sKSXIWU8S7vDZZErbZL4IkVw5SkpxrzuLizqnj3fRkZ3z2Fjtd8l1SmU
  qS9Qr75Rz1BuTfzR/5BFoqpmopQSfApXoPPnQYmD09zQ5cvyRK8xLNDBmVnlB8Xw
  WTzfHEwxwVTTIw8z2Me5UlrrJEDpce7xURa/TsJEb7rhAChQXhk7Z+WH6NXYK9T5
  15DKRzWhfMzYKqwZHtvEbyD9UUKMG8fMZfQAuiUqoZYumzCkqXUkeZezwfkbMpvM
  2iM5FPyqBehyez6fvdDN6MIaRrct4cjIgowJVXMswQKCAQAs9d1OBe/l2UyCI8ck
  0iOzoUNzx/wogqNkHPrcwxR2hpeoDQAiS4ZfuQLLxRltlsv3eAFVnS+kwxDOfS+I
  3X/RGrDAV8y8YUxTP4rO0urrMvZe3TD1BslGbZwgCzxdxbOB0MPap+KtGvvJ3lV3
  WSIDnWxGL4ll0w8kOT8mJx2rWiNTRNVNjiy/kwSyj5MM7KmRR0tDLOOT50VvUSrQ
  +rcqCCwumUQR9QOVFdDb9AsVu3/UrfjCJlJUXoHM9nYPuu8mdL7XLYlG3teiNBSl
  VSXX2OMFRxAltcndVnTM/Oi938B0V1pQ8jF23AlX38XHFFzWyYUeSohf0XdFXWMQ
  tGJvAoIBAQCCzZLne5fNzHVioPrfNBNhEufdDx/Ucpa7HVjCSDgCLeAeivIwUnf+
  nzbOLyG3fi2zl70HZhHOuiCrmLp3v/Qr3cuZR/zsWa2z7CR7EjNF1hCwuiELZA12
  bOK7AGElzxtGchtusM/vQ9uwhGoNJjs3EIaWo7M1GUhPR59YZYIWqPyOoxPmzUkV
  9nnFVPMjThO3X2f2hgM9eG4lOTStIkrfLGSJhMsnYBfEEfoXwZHx9UsD92cK67L1
  NrJ+C3pkTz1W0Ziq1GDEHKnXnJo5mCTUs0oZb9rR/1ahEFWPdWAY6ULxx9tU/Q4P
  P00NfKEzQDG4UI8bzefPIB2/U0yon5KBAoIBAQCnMioEKV4IyALtundgqOPkhAFh
  MzcqJxapgI+46mEsIQJC/u5b5+DBIol8Ym844XMe75fCBDrznI8jWHh0h4OvPg02
  QcgcgljpVF3rx9COuXvFAslvpBhYUXWM7A9jFr+0ScDII+8TRoHPM2KTVabQSjCa
  9c4XYL9oOFm4os5t/HmlTLlGbca2zlucuWaHIp8BwVcl67P1RPZPal5sVXKAVQVn
  KTtVZu/D3cwLnEfZepXlbmRZ8rK9jGGMvOaTZd2lp9kwMOM6cY3r95SIqgpCaYxb
  3glu04pZrvAizdq3C6egv870RhuwZSdlofrVsSYln+sPNJ5kOS/YMZ1I0v5l
  -----END RSA PRIVATE KEY-----
```

To enable access to the site I still had to change the Drupal configuration and targeting file, which looks like this:

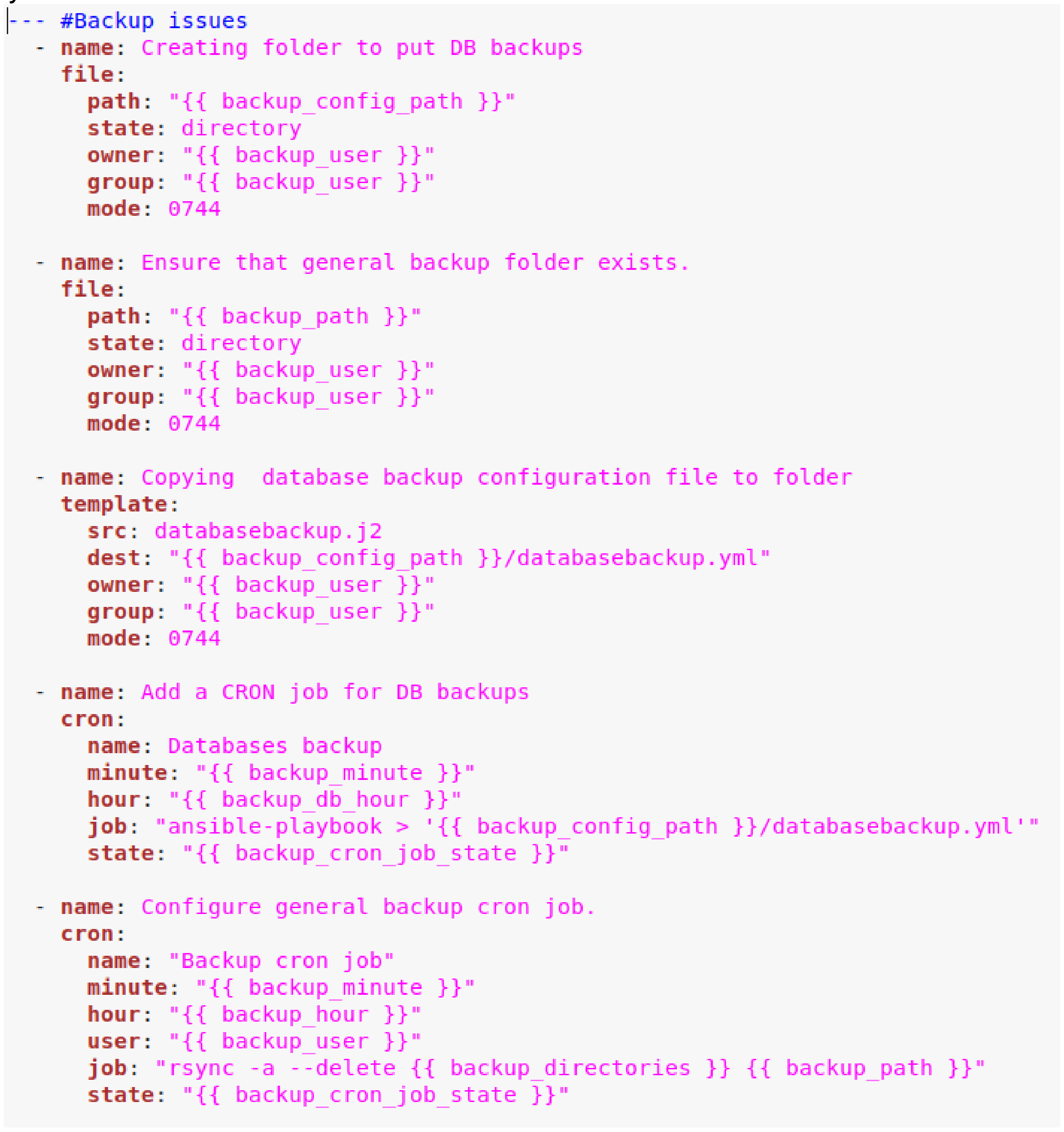
```
server {
    listen 80;
    listen [::]:80;
    server_name juniordevopschallenge.com;

    if ($scheme != "https") {
        return 301 https://$host$request_uri;
    }

}

server {
    listen 443 ssl;
    ssl_certificate {{ cert_root }}/{{ cert_name }} ;
    ssl_certificate_key {{ cert_root }}/{{ cert_key }} ;
    ssl_dhparam {{ cert_root }}/{{ dhparam_name }};

    root /var/www/html/drupal;
    index  index.php index.html index.htm;
    server_name juniordevopschallenge.com;

    location / {
    try_files $uri /index.php?$query_string;
    }

    location @rewrite {
               rewrite ^/(.*)$ /index.php?q=$1;
        }

    location ~ [^/]\.php(/|$) {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/var/run/php/php7.1-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ ^/sites/.*/files/styles/ { # For Drupal >= 7
               try_files $uri @rewrite;
        }

    location ~ ^(/[a-z\-]+)?/system/files/ { # For Drupal >= 7
        try_files $uri /index.php?$query_string;
        }


}
```

As I did not publish a valid domain, my site would not be found by any DNS server, so to access it I would need to inform the ip address of the virtual machine in the /etc/hosts file on my computer.

After the necessary changes and despite giving the warning that the site could be potentially dangerous, since the certificate was self-signed, the site was properly accessed using the "https" protocol.

Sixth step:

It was time to set up the backup but I had no time anymore, Today is the day I got send the project with this document. So I've decided to create a simple backup script using cron and rsync.

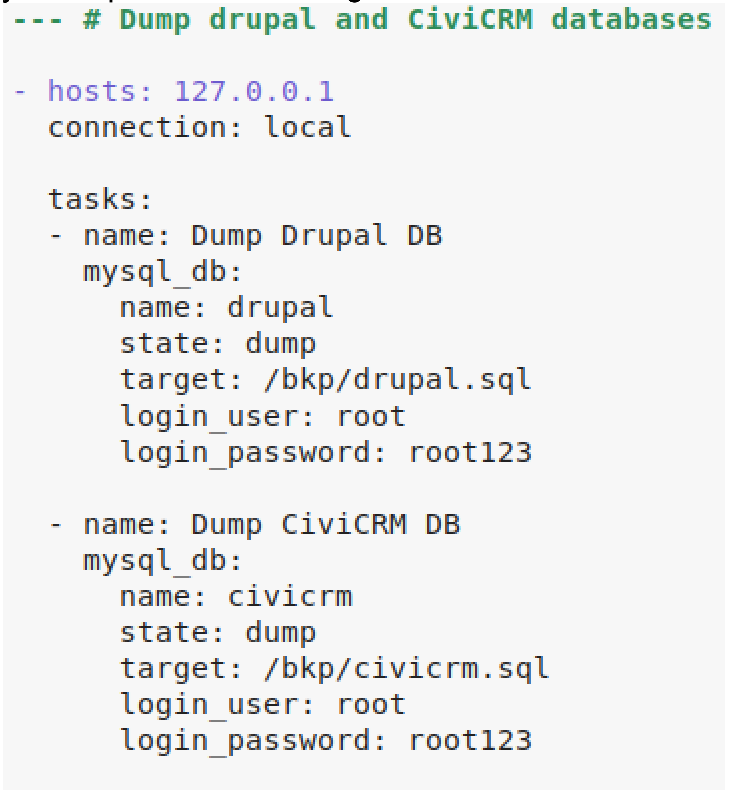


```yaml
--- #Backup issues
  - name: Creating folder to put DB backups
    file:
      path: "{{ backup_config_path }}"
      state: directory
      owner: "{{ backup_user }}"
      group: "{{ backup_user }}"
      mode: 0744

  - name: Ensure that general backup folder exists.
    file:
      path: "{{ backup_path }}"
      state: directory
      owner: "{{ backup_user }}"
      group: "{{ backup_user }}"
      mode: 0744

  - name: Copying  database backup configuration file to folder
    template:
      src: databasebackup.j2
      dest: "{{ backup_config_path }}/databasebackup.yml"
      owner: "{{ backup_user }}"
      group: "{{ backup_user }}"
      mode: 0744

  - name: Add a CRON job for DB backups
    cron:
      name: Databases backup
      minute: "{{ backup_minute }}"
      hour: "{{ backup_db_hour }}"
      job: "ansible-playbook > '{{ backup_config_path }}/databasebackup.yml'"
      state: "{{ backup_cron_job_state }}"

  - name: Configure general backup cron job.
    cron:
      name: "Backup cron job"
      minute: "{{ backup_minute }}"
      hour: "{{ backup_hour }}"
      user: "{{ backup_user }}"
      job: "rsync -a --delete {{ backup_directories }} {{ backup_path }}"
      state: "{{ backup_cron_job_state }}"
```

This is the defaults scripts with the backup vars:

```
---
# DB Backup cron job options.
backup_db_hour: "1"

# Backup cron job options.
backup_cron_job_state: present
backup_hour: "3"
backup_minute: "00"

# User under which backup jobs will run.
backup_user: root

# Path to where backups configuration will be stored.
backup_path: /home/ladis/backup
backup_config_path: /bkp/config

# Directories to back up. {{ backup_user }} must have read access to these dirs.
backup_directories:
  - /bkp
  - /var/www/html
  - /etc
```

Here the template Jinja 2 responsible for doing the databases dum for backup:



```
--- # Dump drupal and CiviCRM databases

- hosts: 127.0.0.1
  connection: local

  tasks:
  - name: Dump Drupal DB
    mysql_db:
      name: drupal
      state: dump
      target: /bkp/drupal.sql
      login_user: root
      login_password: root123

  - name: Dump CiviCRM DB
    mysql_db:
      name: civicrm
      state: dump
      target: /bkp/civicrm.sql
      login_user: root
      login_password: root123
```

Seventh step:

After everithyng done it's time to create an AWS instance, After mannualy created the AWS instance I ran the ansible playbook on it by typing the command:

ansible-playbook -s -i inventory/aws/hosts playbook.yml

To access the site I've informed the ip address of the AWS instance on the /*etc*/hosts file on my computer like this:

```
18.188.192.146 www.juniordevopschallenge.com
18.188.192.146 juniordevopschallenge.com
```

Then, on the webbrowser I've typed https://www.juniordevopschallenge.com.

After access the site I've configured Drupal site and it's database and activated CiviCRM module.