Deploying s2p in jasmin cloud

s2p has IP address 192.171.139.78
domain name is TBD.
For now I can get in with `ssh -A root@192.171.139.78`.

Setup Users etc

Modify hosts file for the system and user we want to create. This is mine at the moment:

```
all:
    hosts:
        192.171.139.78
    vars:
        ansible_user: root
        ansible_port: 22
        user: s2p
        keyname: id_deployrepo_url: git@github.com:bnlawrence
        repo: staff2proj
        home_dir: /home/{{ user }}
        repo_dir: "{{ home_dir }}/{{ repo }}"
        django_dir: "{{ repo_dir }}/django"
        static_dir: "{{ home_dir }}/static"
        django_project: staff2proj
```

Play the zeroth play book:

This sets up users and keys that are necessary to talk to Git.

```
./ansible.sh centos-0-system.yaml
```

You need to take the key that is produced after the message, something like

```
ssh-rsa AAAA ..lines of text .. ansible-generated on your_host
```

and load it into the staff2proj deploy keys on github. (Go to your repo, go to settings, go to deploy keys, and copy and paste your key in there.)

Play the "get setup playbooks"

Make sure your `requirements.txt` is complete and up to date in your repo first!

```
./ansible.sh centos-1-packages.yaml
./ansible.sh centos-2-getapp.yaml
./ansible.sh centos-3-venv.yaml
./ansible.sh centos-4-config-django.yaml
```

Setup Django Superuser

There is a special yaml which has django initial users and passwords for your repo in it, you need to edit out the templates (things starting with `YOUR_`) and run it.

```
./ansible.sh centos-5-superuser-EDITME.yaml
```

Check things

At this point we can put the service up temporarily on port 8000. The port is probably not open to the world.

SSH into your server.

You can check if the port is not open by looking at the output of `iptables-save | grep 8000` which will likely be empty. (You need to do this as route or from an account which you can sudo the command.)

We can open it temporarily and check it's open with

```
firewall-cmd --add-port=8000/tcp
iptables-save | grep 8000
```

Now you can try and run gunicorn on port 8000. Do this as the user you have setup for your django stuff.

```
source django/bin/activate
cd YourProject
gunicorn --bind 0.0.0.0:8000 YourProject.wsgi
```

and you should be able to see something sensible on `https://your_ip_address:8000`.

`<CTRL-C>` in terminal window when done.

Now start setting up the gunicorn services

```
 ./ansible.sh centos-6-gunicorn.yaml
 ./ansible.sh centos-7-gunicorn.yaml
```

This is currently not working.

```
MSG:
Could not find the requested service gunicorn: host
```

When I run `systemctl start gunicorn` I get this:

```
[root@s2p system]# systemctl start gunicorn
Failed to start gunicorn: Unit not found.
```

I note the instructions here suggest that a problem could occur if

- The project files are owned by the root user instead of a sudo user

Which they currently are. So I have created a new ansible script (centos-0a-...) to make my user a sudo user, and ran it.

I then manually changed ownership for the django and project directories. I am now trying again. Manually that seems to work:

```
[root@s2p s2p]# systemctl start gunicorn
[root@s2p s2p]# systemctl enable gunicorn
Created symlink from /etc/systemd/system/multi-user.target.wants/gunicorn.service to
/etc/systemd/system/gunicorn.service.
```

Everything looks good right now. We will have to fix this in the base scripts, but I'm not doing that right now.

NB: I think the gunicorn.socket that is sitting in /home/s2p is redundant, we might want to not bother with putting it here. To check.

Now for Nginx.

We are going to blow away the old nginx configuration file.

```
./ansible.sh centos-8-nginx.yaml
```

At this point the firewall is probably not allow port 80 out. We can manually fix that for the moment and check it:

```
firewall-cmd --add-port=80/tcp
iptables-save | grep 80
```

If you've still got port 8000 open, this might be a good time to close it:

```
firewall-cmd --remove-port=8000/tcp
```

because we want port 80 open permanently so we save the current state:

```
firewall-cmd --runtime-to-permanent
```

Is it working? No it's not. We have a 502! This looks like a permissions problem. Again, we follow the instructions.

Yes, the nginx.log shows the problem:

```
2022/07/23 15:47:15 [crit] 25124#25124: *5 connect() to
unix:/home/s2p/staff2proj/gunicorn.sock failed (13: Permission denied) while connecting to
upstream, client: 200.58.95.187, server: _, request: "GET / HTTP/1.1", upstream:
"http://unix:/home/s2p/staff2proj/gunicorn.sock:/", host: "192.171.139.78:80"
```

which is that nginx can't get to the socket:

```
namei -nom /home/s2p/staff2proj/
f: /home/s2p/staff2proj/
 dr-xr-xr-x root root /
 drwxr-xr-x root root home
 drwx------ s2p  s2p  s2p
 drwxr-xr-x s2p  s2p  staff2proj
```

Problem solved by

```
usermod -a -G nginx s2p
usermod -a -G s2p nginx
chmod -R g+rx /home/s2p
```

Again, we should fix the ansible to get this right from the off, but not right now.