CS312 Project #3

Feburary 25, 2016

Instructions

Please submit all answers as zip or tarball files as mentioned in each task via T.E.A.C.H. This project is due at 4pm Wednesday, March 9th.

This project consists of three separate tasks (Nagios, HAProxy and Docker) that are not connected. Its advised to use a fresh VM for each task. Each task assumes you create an Ansible Role for each. You are free to look at Ansible roles hosted on Ansible Galaxy, however please create the roles from scratch and include attributions of any code you used from those roles.

Questions

Nagios

Create an Ansible Nagios role that does **all** of the following. Please put all new Nagios config files in `/etc/nagios/conf.d` and NRPE config files in `/etc/nrpe.d`. You are free to modify any existing files if needed. Ensure that the Nagios or NRPE daemons are properly reloaded or restarted when config files are changed.

1. Install Nagios, NRPE and all of the Nagios plugins.
2. Start and enable Apache, Nagios and NRPE daemons.

   Go to the nagios page `http://<your ip>/nagios` and login using `nagiosadmin` for both the user and password. Click on '`Services`' under '`Current Status`'. Make sure that all of the checks return 'OK' except for the HTTP check.
3. Create a new HTTP check called `nagios-http` for `localhost` which checks the `/nagios` URL and utilizes the username and password properly so that it returns 'OK'.
4. Create a new host check called `cs312-server` using the IP `140.211.15.183`. Add checks for ping, ssh and an http check for `http://cs312.osuosl.org`.
5. Create a new NRPE config in called `check_all_disks` which checks all of the disks and mounted volumes except for `/sys` and `/proc`.

Create a zip or tarball file named `nagios.zip` or `nagios.tar.gz` which includes the following:

- Nagios Ansible Role
- The output of `journalctl -u nagios -u nrpe` in a log file

HAProxy

Create an Ansible HAProxy role that does **all** of the following. We will be configuring HAProxy to serve a site that uses 8 different application servers that are simulated with a simple python one-liner via systemd. Here are the details about each application server:

- We have two blog applications running on ports 8000 and 8001 that will show 'Blog Page'.
- We have one intranet application running on port 8002 that will show 'Intranet Page'.
- We have five www applications running on ports 8003, 8004, 8005, 8006 and 8007 that will show 'WWW Page'.

Tasks:

6. Install and setup HAProxy. Setup the `global` and `defaults` sections like we did in class. For now don't add any frontends or backends.
7. Configure HAProxy to send logs to `systemd` by default.
8. Setup the HAProxy admin port so we can see stats.
9. Download this script (`http://cs312.osuosl.org/_static/hw/haproxy.sh`) and run it once. This will setup a few simple HTTP servers using python to simulate a cluster of applications. Make sure you see output when you run `journalctl -u cs312*`.
10. Create a frontend on port 80.
11. Create three backends called `blog`, `intranet` and `www` that connect to the ports mentioned above for each app (assume you're using `localhost` as the hostname).
12. For the www backend, give the apps running on port 8006 and 8007 a weight of 100, while the others have a weight of 50.
13. Setup acls for `/blog` that point to the blog app, and `/intranet` that point to the intranet app.
14. Set the www backend as the default backend.

Now try accessing the site. Do `/blog` and `/intranet` show the correct content? Does the main page work properly? Access each URL several times to ensure the weighting is working properly.

Create a zip or tarball file named `haproxy.zip` or `haproxy.tar.gz` which includes the following:

- HAProxy Ansible Role
- The output of `journalctl -u haproxy` in a log file

Docker

Create an Ansible Docker role that does **all** of the following:

15. Install, enable and start Docker and its services
16. Create a `Dockerfile` in `/tmp` which will be used to build the cs312 repo and output it over nginx. Use the following as a base:

```
FROM nginx
MAINTAINER <your email here>

RUN apt-get update

# Install build-essential, python-virtualenv, git and other
# dependencies here

# Clone the cs312 repo
RUN git clone https://github.com/osuosl/cs312

# Run the cs312 build script

# Copy the resulting build to /usr/share/nginx/html
RUN cp -r cs312/build/html/* /usr/share/nginx/html
```

17. Build a Docker image called `cs312/site` using the `Dockerfile` you made in the previous question.
18. Create a systemd unit file for the docker image in the previous question. Here is a base file to get you going:

```
[Unit]
Description=cs312 service
```

```
BindsTo=cs312.service

[Service]
ExecStartPre=-/usr/bin/docker <kill old container here>
ExecStartPre=-/usr/bin/docker <remove old container here>
ExecStart=/usr/bin/docker <run new container here, make sure
    ↪  to expose port 80!>
ExecStop=/usr/bin/docker <stop running container here>
```

19. Start and enable the systemd unit you created in the previous question
20. Verify that you can access the website

    Create a zip or tarball file named `docker.zip` or `docker.tar.gz` which includes the following:

    - Docker Ansible Role
    - The output of `journalctl -u cs312` in a log file
    - The output of `docker ps` in a log file