### Instruction

Group14 members:

Daming Li-15398736, Moravy Oum-16859528, Yaozu Zhang-15398302, Simon Freeman-13036748

This is a document that describes the entire IT engineering project. If readers want to see ‘Testing/demonstration’ only, they can tend to a document named ‘test\_or\_demonstration’. It has several parts. In the first part, it introduces the readers to the selection topic of the fourteenth group and an introduction to the topic. In the second part, it introduces the readers to the tasks of the fourteenth group and the techniques to be used to solve the problem and the problems encountered in the task, and the technical introductions are also used when necessary. In the third part, the readers will see how we implement our concepts to solve problems in assignment2.

This is the GitHub web address: <https://github.com/Damming/2018_Group_14>

First of all, the research topic of the group is Open Street Map. So what is this Open Street Map?

Open Street Map(OSM) is a free open source, editable map service created by the Internet. Open Street Map uses public collective power and unpaid contributions to improve map-related geographic data. OSM is not-for-profit, it returns data to the community for reuse in other products and services. Other maps sell most of the map data to third parties. Maps of OSM are drawn by users based on portable GPS devices, aerial photography photos, other free content, and even local intelligence. The map image and vector data in the website are all shared by the creative name - the same way to share the authorization. [1]

The first step is to create a virtual Ubuntu server on the AWS website, and then all you have to do is configure the server. Here's a tutorial on how to configure Open street map. But the virtual server we use is not as complicated as the one in the tutorial. We only need a part to complete the server configuration.

If the readers wants to see the original installation tutorial, this is the URL: <https://ircama.github.io/osm-carto-tutorials/tile-server-ubuntu/>. When the configuration server is finished, you can start the first part of the task.

The first Assignment:

Our first task was to automatically configure OSM on an Ubuntu server (First task of Assignment1). Here we need to use a shell script. It uses a variety of commands to be written into a file, which is convenient for one-time execution of a program file, mainly for the administrator to set up or manage. It uses commands under Linux/Unix. [2] Script files for this issue can be found in the Ass1\_Task1 folder under GitHub. In this folder, there are two script files called ‘set\_up\_OpenStreetMap.sh’ and ‘set\_up\_OpenStreetMap\_with\_password.sh’ . The first script file is about installing the necessary OSM tool kits on an Ubuntu server. There is a specific toolkit name on the note. The second script file is about how to configure the OSM server and can build an user instance in database.

Our second task was to include the setup of the server and use a configuration management tool instead of a shell script. This required Ansible, which is a simple automated operation and maintenance management tool, based on the Python language. It is built by two key modules, Paramiko and PyYAML. It can be used to automate the deployment of applications and configurations, and implements batch system configuration, batch program deployment, batch run commands, etc.[3]

In this section, our main goal is to translate the script files from the previous section into Ansible's playbook format to automatically configure OSM on other Ubuntu servers. The readers can find a playbook file called ‘prepared\_new\_instance.yml’ in a folder called Ass1\_Task2. This file is important because the playbook file is also modified based on this in the second assignment, which readers will see later. If readers want to what the content is of the playbook, check the notes.

Next is the second assignment:

Our group chooses the following question:

1.How can your system be upgraded in a safe manner and avoiding downtime?

2.What else would be important in a DevOps approach?

If the readers want to see the concepts involved in these questions, they can go to another file to find these concepts.

In the first task, six AWS instances are required. They are 1 assistant server, 1 database server, 3 OpenStreetMap program servers (including 1 backup), and 1 nginx server. Only assistant server needs to be setup manually. The other servers can be setup by Ansible playbook running on the assistant server. Assistant server can also be used to update program server.

In a folder called ‘Ass2\_Q2’, readers can find a folder called Shell Scripts. There are 6 shellscript files in this folder. All of these files should be rewritten in the form of a playbook. The meaning of breaking shellscript into 6 small parts is to facilitate rewriting them to the form of playbook. The rewritten playbook file was placed in a ‘Q2\_playbook.yml’ file called the Playbook folder. Also in this folder, there is a file called ‘update.yml’. It means that Update OpenStreetMap program server then run in assistant\_server. Let me explain these in detail.

We have four servers now. They are assistant\_server which contains Ansible and other tools, database\_server which is postgresql, map\_program\_server which contains map programs and nginx\_server which is Nginx.

There are four steps that set up instances:

Step 1: Use codes from ansible.sh shell script set up assistant\_server.

Step 2: Use Ansible in assistant\_server set up a database\_server and record its IP. This requires codes from database.sh(1), assistant.sh, database.sh(2).

Step 3: Use Ansible in assistant\_server set up two map\_program\_server with database\_server's IP and record the IPs of those two map\_program\_server. This requires codes from map\_server.sh.

Step 4: Use Ansible in assistant\_server set up nginx\_server and config gninx with map\_program\_server's IP. This requires codes from conf\_nginx.sh.

Step 5: Use Ansible in assistant\_server update those two map\_program\_server with their IPs. This requires codes from update.sh.

Now let us focus on the second question:

This section consists of three documents. The idea is as following. We now have an old database. The first thing to do is to back up the data in this database and then delete the original database. Finally, restore a new database with the backed up data. The file name corresponds to the respective step.

Reference:

1. Open Street Map, Retrieve from <https://baike.baidu.com/item/Open%20Street%20Map/11057294.>
2. Shell script, Retrieve from <https://baike.baidu.com/item/Shell%E8%84%9A%E6%9C%AC/572265?fr=aladdin>.
3. Ansible, Retrieve from <https://www.iyunv.com/thread-385359-1-1.html>.