NWEN 406: Project #1

Due on Friday, August 7, 2015

Boxiong Tan

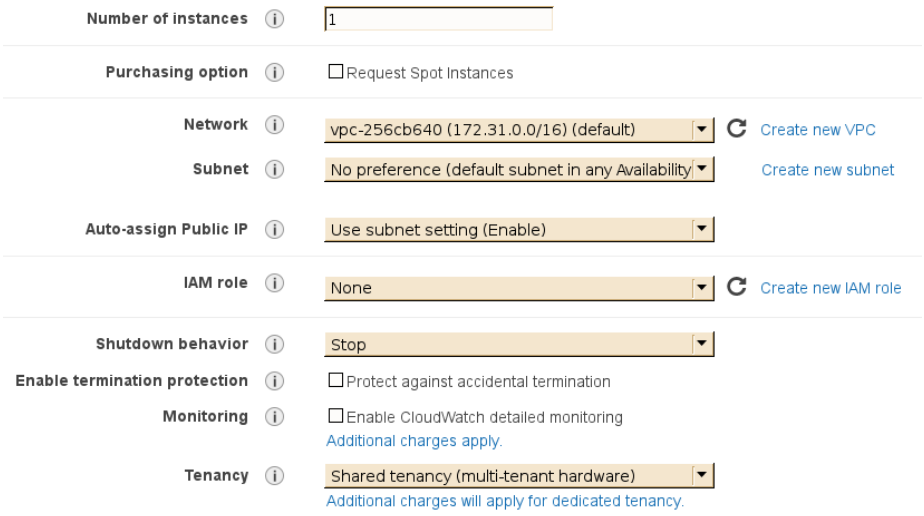
Project Objectives

1. Obtain an Amazon AWS micro-instance and configure it.
2. Write a server which could receive and send messages.

Micro-instance Configuration

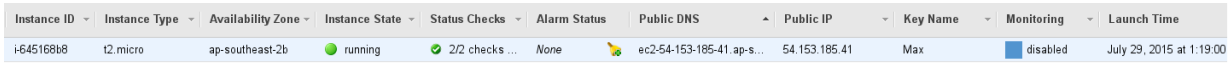
Major steps of configuration:

1. Choose an Amazon Machine Image (AMI) → **Ubuntu Server**
2. Choose an instance Type → **t2.micro**
3. Configure instance Details



4. Add storage → **8 GB General Purpose (SSD)**
5. Tag Instance → **"Max":"MyServer"**
6. Configure Security Group → **All traffic**

After configuration, launch the micro-instance.



Web Server

Programming Environment

Programming language: **Python v2.7**
Web Server framework: **Flask v0.10.1**

Description

The Restful Web server runs in the background and listening port 80. When receiving a POST request, it first reply the request by code 202. Then start a thread to handle the message.
For the sake of simplicity, our team decides to pass a json message between members. This message mainly contains a string message, which each member must edited it when received the message. Each member could handle this string message with a simple hashing or string mashing algorithm.

Web Server [Description] continued on next page. . .

Algorithm 1 Web server

Listen to HTTP request on myIP:80/api
if get POST request **then**
  Start a thread handle the received data
  reply the request with “code 202”
end if
Thread unpack the received data
Thread identify the input message and reverse the message string
Thread edit the index number and count
Thread Pack the edited message as a dictionary
Thread add edited message dictionary to “max” message list
repeat
  Thread pop up the first address in the order listed
  for three times **do**
    and try to send the json package to it.
  end for
until Send successfully or order list is empty

This is an example of starting message:

```
{
    "value": "thebeginningvalue",
    "count": 0,
    "audit": {},
    "order": [
        "52.25.73.1",
        "52.25.246.56",
        "52.25.246.34",
        "52.25.185.43",
        "52.25.56.34",
        "52.25.127.88"
    ]
}
```

The string message allows characters for input and output are 0-9a-zA-Z. The output at least one character and maximum of 255. i.e. /[0-9a-zA-Z]1,255/

Experimental Results

Sending JSON to next address: 52.27.64.194

```
{
    "audit": {
        "Alex": [
            {
                "index": 1,
                "input": "blasomething",
                "output": "Areyouoneofthoseboyswhopreferscarstowomen",
                "time": "2015-08-05 22:17:45 GMT + 0"
            }
```

```
        ],
        "max": [
            {
                "index": 0,
                "input": "",
                "output": "blasomething",
                "time": "Wed, 05, Aug 2015 22:17:54 GMT"
            }
        ]
    },
    "count": 2,
    "order": [
        "52.27.228.163",
        "54.68.184.120"
    ],
    "value": "Areyouoneofthoseboyswhopreferscarstowomen"
}
```


This example shows, “max” starts sending the message to “Alex”. “Alex” further forward this message to 52.27.64.194.