from \_\_future\_\_ import print\_function

import troposphere.ec2 as ec2

import os

from troposphere import Base64, Join, ImportValue

from troposphere import Parameter, Ref, Tags, Template

from troposphere.ec2 import SecurityGroupRule, SecurityGroup

template = Template()

template.add\_description("Node.js, Nginx stack w/PM")

def make\_name(component\_name):

return Join('', [component\_name, ' - ', Ref('AWS::StackName')])

LinuxAmi = template.add\_parameter(Parameter(

"LinuxAmi",

Type="AWS::EC2::Image::Id",

Description="The AMI ID for our Linux Web server instance. (default: Amazon Linux AMI 2017 - ami-489f8e2c)",

Default="ami-489f8e2c",

))

keyname\_param = template.add\_parameter(Parameter(

"KeyPairName",

Description="Name of an existing EC2 KeyPair to enable SSH access to the instance",

Type="AWS::EC2::KeyPair::KeyName",

Default="nld\_aws\_london"

))

instanceSecurityGroup = template.add\_resource(

SecurityGroup(

'InstanceSecurityGroup',

GroupDescription='Enable EC2 HTTP, HTTPS',

SecurityGroupIngress=[

SecurityGroupRule(IpProtocol='tcp', FromPort='22', ToPort='22', CidrIp='0.0.0.0/0'),

SecurityGroupRule(IpProtocol='tcp', FromPort='80', ToPort='80', CidrIp='0.0.0.0/0'),

SecurityGroupRule(IpProtocol='tcp', FromPort='443', ToPort='443', CidrIp='0.0.0.0/0'),

],

VpcId=ImportValue('VPC'),

Tags=Tags(Name=make\_name('Nginx, Node & PM'))

)

)

server\_path = os.path.dirname(os.path.realpath(\_\_file\_\_))

with open(server\_path + '/../includes/server.js', 'r') as serverjs:

server\_app = serverjs.read()

with open(server\_path + '/../includes/pm2config.json', 'r') as pm2:

pm2\_config = pm2.read()

with open(server\_path + '/../includes/nginx.conf', 'r') as nginx:

nginx\_conf = nginx.read()

user\_data\_server\_app = """cat << EOF >> /var/www/html/example-app/server.js

%s

EOF

""" % server\_app

user\_data\_pm2\_config = """cat << EOF >> /var/www/html/example-app/pm2config.json

%s

EOF

""" % pm2\_config

user\_data\_nginx\_conf = """cat << EOF > /etc/nginx/nginx.conf

%s

EOF

""" % nginx\_conf

user\_data\_nginx\_index = """cat << EOF >> /var/www/html/index.html

<h1>Nginx server, running at <code>/var/www/html</code></h1>

EOF

"""

user\_data\_bashrc = """ cat << EOF >> /home/ec2-user/.bashrc

# User

export PATH=./bin:$PATH

alias l='ls -alh'

# .pm2

export USER=ec2-user

export HOME=/home/ec2-user

export PM2\_HOME=/home/ec2-user/.pm2

EOF

"""

ec2\_instance = template.add\_resource(ec2.Instance(

"Ec2Instance",

ImageId=Ref(LinuxAmi),

InstanceType="t2.small",

KeyName=Ref(keyname\_param),

SecurityGroupIds=[Ref(instanceSecurityGroup)],

SubnetId=ImportValue('Subnet1'),

Tags=Tags(Name=make\_name('Nginx, Node & PM')),

UserData=Base64(Join('', [

"#!/bin/bash\n",

"sed -i '/ZONE=\"UTC\"/c\ZONE=\"Europe/London\"' /etc/sysconfig/clock \n",

"ln -sf /usr/share/zoneinfo/Europe/London /etc/localtime\n",

"echo [UserData] Updating Packages\n",

"yum update -y\n",

# Update bashrc with exports

user\_data\_bashrc,

". /home/ec2-user/.bashrc\n",

"mkdir -p /var/www/html/example-app\n",

"groupadd www\n",

"usermod -a -G www ec2-user\n",

"chown -R ec2-user:www /var/www\n",

"chmod 2775 /var/www\n",

"find /var/www -type d -exec chmod 2775 {} +\n",

"find /var/www -type f -exec chmod 0664 {} +\n",

user\_data\_server\_app,

"curl --silent --location https://rpm.nodesource.com/setup\_6.x | sudo bash -\n",

"yum -y install nodejs\n",

"yum -y install nginx\n",

"chkconfig nginx on\n",

user\_data\_nginx\_index,

user\_data\_nginx\_conf,

"service nginx restart\n",

user\_data\_pm2\_config,

"npm install pm2 -g\n",

"pm2 startup -u ec2-user --hp /home/ec2-user\n",

"sudo env PATH=$PATH:/usr/bin /usr/lib/node\_modules/pm2/bin/pm2 startup systemv -u ec2-user --hp /home/ec2-user\n"

"pm2 start /var/www/html/example-app/pm2config.json -u ec2-user --hp /home/ec2-user\n",

"pm2 save\n",

"chown -R ec2-user:ec2-user /home/ec2-user/.pm2\n",

"\n",

]

))

))

eip\_association = template.add\_resource(ec2.EIPAssociation(

"EIPEC2Association",

EIP=ImportValue('ElasticIP'),

InstanceId=Ref(ec2\_instance),

))

with open(os.path.realpath(\_\_file\_\_)[:-2] + 'json', 'w') as the\_file:

print(template.to\_json(), file=the\_file)

print('Generated: ' + \_\_file\_\_)

{

"Description": "Node.js, Nginx stack w/PM",

"Parameters": {

"KeyPairName": {

"Default": "nld\_aws\_london",

"Description": "Name of an existing EC2 KeyPair to enable SSH access to the instance",

"Type": "AWS::EC2::KeyPair::KeyName"

},

"LinuxAmi": {

"Default": "ami-489f8e2c",

"Description": "The AMI ID for our Linux Web server instance. (default: Amazon Linux AMI 2017 - ami-489f8e2c)",

"Type": "AWS::EC2::Image::Id"

}

},

"Resources": {

"EIPEC2Association": {

"Properties": {

"EIP": {

"Fn::ImportValue": "ElasticIP"

},

"InstanceId": {

"Ref": "Ec2Instance"

}

},

"Type": "AWS::EC2::EIPAssociation"

},

"Ec2Instance": {

"Properties": {

"ImageId": {

"Ref": "LinuxAmi"

},

"InstanceType": "t2.small",

"KeyName": {

"Ref": "KeyPairName"

},

"SecurityGroupIds": [

{

"Ref": "InstanceSecurityGroup"

}

],

"SubnetId": {

"Fn::ImportValue": "Subnet1"

},

"Tags": [

{

"Key": "Name",

"Value": {

"Fn::Join": [

"",

[

"Nginx, Node & PM",

" - ",

{

"Ref": "AWS::StackName"

}

]

]

}

}

],

"UserData": {

"Fn::Base64": {

"Fn::Join": [

"",

[

"#!/bin/bash\n",

"sed -i '/ZONE=\"UTC\"/c\\ZONE=\"Europe/London\"' /etc/sysconfig/clock \n",

"ln -sf /usr/share/zoneinfo/Europe/London /etc/localtime\n",

"echo [UserData] Updating Packages\n",

"yum update -y\n",

" cat << EOF >> /home/ec2-user/.bashrc\n\n# User\nexport PATH=./bin:$PATH\nalias l='ls -alh'\n\n# .pm2\nexport USER=ec2-user\nexport HOME=/home/ec2-user\nexport PM2\_HOME=/home/ec2-user/.pm2\nEOF\n",

". /home/ec2-user/.bashrc\n",

"mkdir -p /var/www/html/example-app\n",

"groupadd www\n",

"usermod -a -G www ec2-user\n",

"chown -R ec2-user:www /var/www\n",

"chmod 2775 /var/www\n",

"find /var/www -type d -exec chmod 2775 {} +\n",

"find /var/www -type f -exec chmod 0664 {} +\n",

"cat << EOF >> /var/www/html/example-app/server.js\nvar http = require('http');\nvar port = 9500; // Must change IPTables also\nvar server = http.createServer(function(request, response) {\n response.setHeader('Content-Type', 'text/html');\n response.end('<h1>Node.js server is running!</h1>');\n});\nserver.listen(port, function() {\n console.log('Node.js server listening on port: ' + port);\n});\n\nEOF\n",

"curl --silent --location https://rpm.nodesource.com/setup\_6.x | sudo bash -\n",

"yum -y install nodejs\n",

"yum -y install nginx\n",

"chkconfig nginx on\n",

"cat << EOF >> /var/www/html/index.html\n<h1>Nginx server, running at <code>/var/www/html</code></h1>\nEOF\n",

"cat << EOF > /etc/nginx/nginx.conf\n# For more information on configuration, see:\n# \* Official English Documentation: http://nginx.org/en/docs/\n# \* Official Russian Documentation: http://nginx.org/ru/docs/\n\nuser nginx;\nworker\_processes auto;\nerror\_log /var/log/nginx/error.log;\npid /var/run/nginx.pid;\n\n# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.\ninclude /usr/share/nginx/modules/\*.conf;\n\nevents {\n worker\_connections 1024;\n}\n\nhttp {\n log\_format main '$remote\_addr - $remote\_user [$time\_local] \"$request\" '\n '$status $body\_bytes\_sent \"$http\_referer\" '\n '\"$http\_user\_agent\" \"$http\_x\_forwarded\_for\"';\n\n access\_log /var/log/nginx/access.log main;\n\n sendfile on;\n tcp\_nopush on;\n tcp\_nodelay on;\n keepalive\_timeout 65;\n types\_hash\_max\_size 2048;\n\n include /etc/nginx/mime.types;\n default\_type application/octet-stream;\n\n # Load modular configuration files from the /etc/nginx/conf.d directory.\n # See http://nginx.org/en/docs/ngx\_core\_module.html#include\n # for more information.\n include /etc/nginx/conf.d/\*.conf;\n\n index index.html index.htm;\n\n upstream node\_app {\n server 127.0.0.1:9500;\n }\n\n server {\n listen 80 default\_server;\n listen [::]:80 default\_server;\n server\_name localhost;\n root /var/www/html;\n\n # Load configuration files for the default server block.\n include /etc/nginx/default.d/\*.conf;\n\n location / {\n proxy\_pass http://node\_app;\n }\n\n # redirect server error pages to the static page /40x.html\n #\n error\_page 404 /404.html;\n location = /40x.html {\n }\n\n # redirect server error pages to the static page /50x.html\n #\n error\_page 500 502 503 504 /50x.html;\n location = /50x.html {\n }\n\n # proxy the PHP scripts to Apache listening on 127.0.0.1:80\n #\n #location ~ \\.php$ {\n # proxy\_pass http://127.0.0.1;\n #}\n\n # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000\n #\n #location ~ \\.php$ {\n # root html;\n # fastcgi\_pass 127.0.0.1:9000;\n # fastcgi\_index index.php;\n # fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;\n # include fastcgi\_params;\n #}\n\n # deny access to .htaccess files, if Apache's document root\n # concurs with nginx's one\n #\n #location ~ /\\.ht {\n # deny all;\n #}\n }\n\n# Settings for a TLS enabled server.\n#\n# server {\n# listen 443 ssl http2 default\_server;\n# listen [::]:443 ssl http2 default\_server;\n# server\_name \_;\n# root /var/www/html;\n#\n# ssl\_certificate \"/etc/pki/nginx/server.crt\";\n# ssl\_certificate\_key \"/etc/pki/nginx/private/server.key\";\n# # It is \*strongly\* recommended to generate unique DH parameters\n# # Generate them with: openssl dhparam -out /etc/pki/nginx/dhparams.pem 2048\n# #ssl\_dhparam \"/etc/pki/nginx/dhparams.pem\";\n# ssl\_session\_cache shared:SSL:1m;\n# ssl\_session\_timeout 10m;\n# ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;\n# ssl\_ciphers HIGH:SEED:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!RSAPSK:!aDH:!aECDH:!EDH-DSS-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA:!SRP;\n# ssl\_prefer\_server\_ciphers on;\n#\n# # Load configuration files for the default server block.\n# include /etc/nginx/default.d/\*.conf;\n#\n# location / {\n# }\n#\n# error\_page 404 /404.html;\n# location = /40x.html {\n# }\n#\n# error\_page 500 502 503 504 /50x.html;\n# location = /50x.html {\n# }\n# }\n\n}\n\nEOF\n",

"service nginx restart\n",

"cat << EOF >> /var/www/html/example-app/pm2config.json\n{\n\t\"apps\": [{\n\t\t\"name\"\t: \"example-app\",\n\t\t\"script\": \"/var/www/html/example-app/server.js\",\n\t\t\"watch\": [\"views\", \"public\", \"routes\"],\n\t\t\"instances\" : \"max\",\n \t\"exec\_mode\" : \"cluster\",\n\t\t\"env\": {\n\t\t\t\"NODE\_ENV\": \"development\"\n\t\t},\n\t\t\"env\_production\": {\n\t\t\t\"NODE\_ENV\": \"production\"\n\t\t},\n\t\t\"log\_date\_format\": \"MM-DD-YYYY HH:mm Z\",\n\t\t\"error\_file\": \"/home/ec2-user/.pm2/logs/myapp\_err.log\",\n\t\t\"out\_file\": \"/home/ec2-user/.pm2/logs/myapp\_out.log\"\n\t}]\n}\n\nEOF\n",

"npm install pm2 -g\n",

"pm2 startup -u ec2-user --hp /home/ec2-user\n",

"sudo env PATH=$PATH:/usr/bin /usr/lib/node\_modules/pm2/bin/pm2 startup systemv -u ec2-user --hp /home/ec2-user\npm2 start /var/www/html/example-app/pm2config.json -u ec2-user --hp /home/ec2-user\n",

"pm2 save\n",

"chown -R ec2-user:ec2-user /home/ec2-user/.pm2\n",

"\n"

]

]

}

}

},

"Type": "AWS::EC2::Instance"

},

"InstanceSecurityGroup": {

"Properties": {

"GroupDescription": "Enable EC2 HTTP, HTTPS",

"SecurityGroupIngress": [

{

"CidrIp": "0.0.0.0/0",

"FromPort": "22",

"IpProtocol": "tcp",

"ToPort": "22"

},

{

"CidrIp": "0.0.0.0/0",

"FromPort": "80",

"IpProtocol": "tcp",

"ToPort": "80"

},

{

"CidrIp": "0.0.0.0/0",

"FromPort": "443",

"IpProtocol": "tcp",

"ToPort": "443"

}

],

"Tags": [

{

"Key": "Name",

"Value": {

"Fn::Join": [

"",

[

"Nginx, Node & PM",

" - ",

{

"Ref": "AWS::StackName"

}

]

]

}

}

],

"VpcId": {

"Fn::ImportValue": "VPC"

}

},

"Type": "AWS::EC2::SecurityGroup"

}

}

}