Lab 2: Deploying and using RStudio (R) in AWS

In this lab, we'll be deploying an RStudio server onto an EC2 instance. Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

To deploy our EC2 instance, we're going to use a service called CloudFormation. AWS CloudFormation provides a common language for you to describe and provision all the infrastructure resources in your cloud environment. CloudFormation allows you to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all regions and accounts. This file serves as the single source of truth for your cloud environment.

Here's what we hope to accomplish in this lab:

1. Provision an EC2 instance using CloudFormation
2. Execute and run sample code in an RStudio environment
3. Clean up and delete all resources

Deploy RStudio

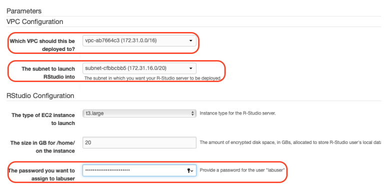
To start, we'll need to deploy an RStudio EC2 instance. We'll use a CloudFormation template to deploy a stack.

Curious?

Want to learn more about CloudFormation? Check out this getting started doc!

Launch Stack

1. Click here to launch the stack:



Parameters
VPC Configuration

Which VPC should this be deployed to?	vpc-ab7664c3 (172.31.0.0/16)	
The subnet to launch RStudio into	subnet-cfbbcbb5 (172.31.16.0/20)	The subnet in which you want your R-Studio server to be deployed.

RStudio Configuration

The type of EC2 instance to launch	t3.large	Instance type for the R-Studio server.
The size in GB for /home/ on the instance	20	The amount of encrypted disk space, in GBs, allocated to store R-Studio user's local dat
The password you want to assign to labuser	•••••••••••••••••••••••	Provide a password for the user "labuser"

2. You will be directed to a page to fill in details about this stack. Take note of the following:

   2a. For VPC configuration, you should only have **1** VPC listed. Select this VPC. *Your VPC ID will be different than the one pictured here!*

   2b. Select a subnet to launch your instance into. Any of the **3** subnets will do.

   2c. Specify a password that you'll use to log in to RStudio. You'll need this password to log in, so make sure you enter it properly and remember it.

   2d. In the **Capabilities** dialogue box at the bottom, be sure to click the checkbox acknowledging that you'll be creating IAM resources.

   2e. Click **Create stack** at the bottom right.

What's Happening?

Well that was easy. So what did we just do?

1. We used a CloudFormation template to launch a stack. The template allowed us to specify parameters to customize the deployment to meet our needs.
2. After we hit **Create stack**, the template is then parsed and sent to the CloudFormation service. This service translates the template into API calls and builds the resources for us.

3. It will take a few minutes for the stack to deploy, so now's a good time to explore the CloudFormation console. So items to note:

   3a. Outputs: You can optionally output information from a stack. When this stack is completed, we'll want to come back here to find the URL for our RStudio server.

   3b. Resources: Any AWS resource this stack created will be available here.

   3c. Events: Any event related to this stack (e.g. updates, new resources getting created, resources deleted, etc.) will be timestamped here.

   3d. Template: The CloudFormation template used to provision this stack.

   3e. Parameters: The parameters and their corresponding values we entered for this stack.

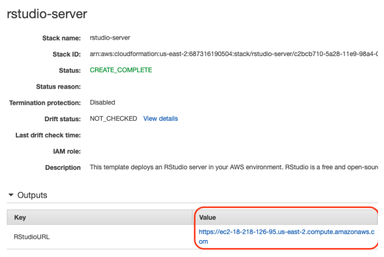
   3f. Tags: Any tags associated with this stack.

   3g. Rolback Triggers: In the event that the stack deployment fails, you can trigger an event (e.g. send a message).

   3h. Policy: You can associate a policy to this stack to enforce permissions, e.g. prevent users from deleting a production database.

3i. Change Sets: A change set allows you to preview how proposed changes to a stack might impact your running resources.

4. Explland the **Outputs** dropdown. Copy and paste the value for **RStudioURL**, and open this URL in a new tab / window.



rstudio-server

Stack name:	rstudio-server
Stack ID:	arn:aws:cloudformation:us-east-2:687316190504:stack/rstudio-server/c2bcb710-5a28-11e9-98a4-0
Status:	CREATE_COMPLETE
Status reason:	
Termination protection:	Disabled
Drift status:	NOT_CHECKED View details
Last drift check time:	
IAM role:	
Description	This template deploys an RStudio server in your AWS environment. RStudio is a free and open-sour

Outputs

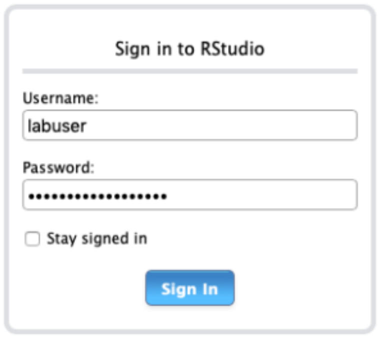
Key	Value
RStudioURL	https://ec2-18-218-126-95.us-east-2.compute.amazonaws.com

Whoa there

Are you getting an error that you can't connect to the server URL? That's because the service is still coming up on the EC2 instance. Give it another minute or so and you should be good to go.

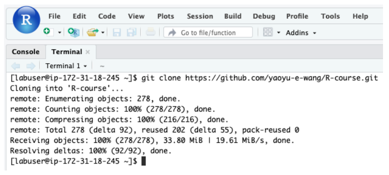
5. We're using a self-signed certificate here, so ignore the SSL warning from your browser.

6. For the username and password, enter **labuser** for the user, and specify the password you entered in **Step 2**.



7. In the RStudio window, click on the **Terminal** tab towards the top left of the screen and run the following:

```
git clone https://github.com/JamesSWiggins/R-course.git
```



```
[labuser@ip-172-31-18-245 ~]$ git clone https://github.com/yaoyu-e-wang/R-course.git
Cloning into 'R-course'...
remote: Enumerating objects: 278, done.
remote: Counting objects: 100% (278/278), done.
remote: Compressing objects: 100% (216/216), done.
remote: Total 278 (delta 92), reused 202 (delta 55), pack-reused 0
Receiving objects: 100% (278/278), 33.80 MiB | 19.61 MiB/s, done.
Resolving deltas: 100% (92/92), done.
[labuser@ip-172-31-18-245 ~]$
```

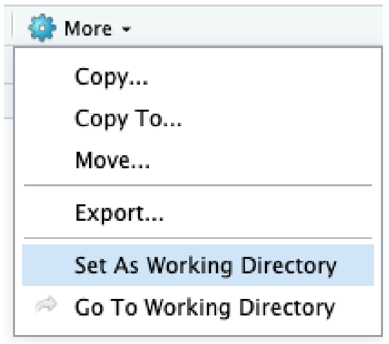
No Terminal Prompt?

If you don't see a prompt when you click the Terminal tab, it may be a browser incompatibility problem. We've had success when using Google Chrome for this lab.

8. Click on the **R-course** folder in the bottom right window.

9. Click on the **Basic Data Analysis** folder.

10. Click on the **More** dropdown, and select **Set As Working Directory**. See image on the right:

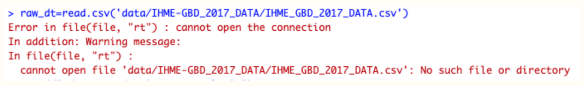


11. Click on **data_preprocessing.R**. This will open the code in the top left window.

12. Click on the Run icon to execute each line within the program.

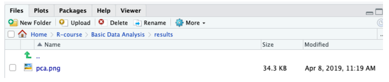
Oops!

If you get an error like the following, double check you set the working directory properly as mentioned in **Step 10**.



```
> raw_dt=read.csv('data/IHME-GBD_2017_DATA/IHME_GBD_2017_DATA.csv')
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'data/IHME-GBD_2017_DATA/IHME_GBD_2017_DATA.csv': No such file or directory
```

13. After you've executed each line in the **data_preprocessing.R** file, click on the **cluster_code.R** file and execute each line here, following the same procedure as above.



Home > R-course > Basic Data Analysis > results

Name	Size	Modified
..		
pca.png	34.3 KB	Apr 8, 2019, 11:19 AM

13a. As you step through this code, it will generate graphs along the way. It will generate a folder called **results**, with various charts which you can view.

13b. Use the file browser in the bottom right to view these graphs.

14. Now that you've walked through some example code on your very own RStudio server, it's time to clean up. Since we used CloudFormation to deploy these resources, it's trivial to delete our infrastructure. Simply delete the stack!

14a. Jump back into the CloudFormation console.

14b. Select the check box next to the stack called **rstudio-server**.

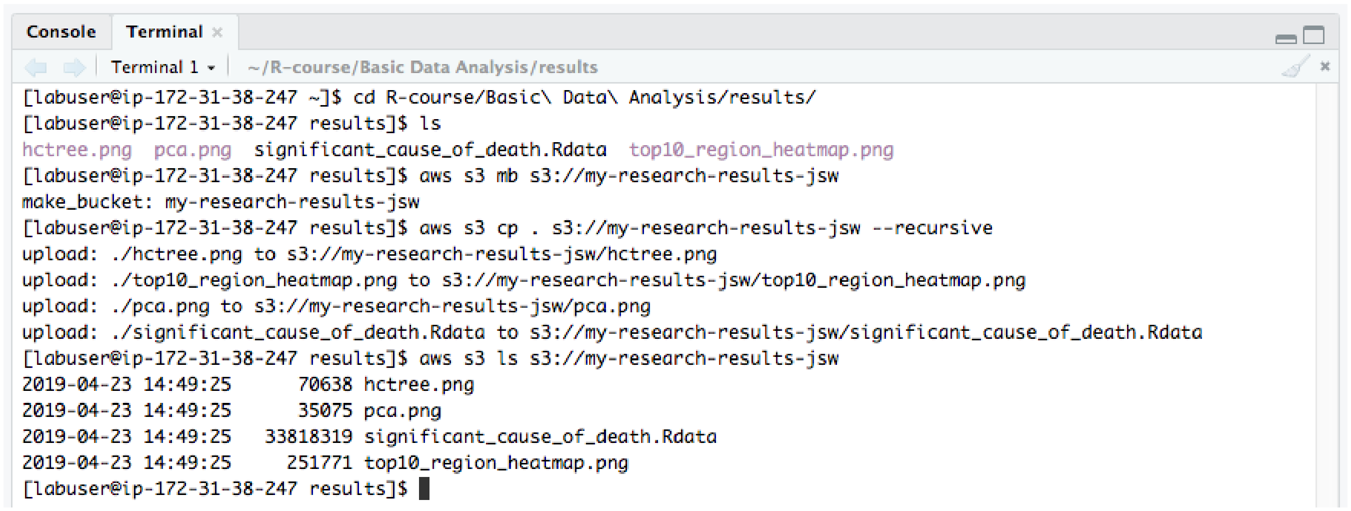
14c. From the **Actions** dropdown button towards the top left, select **Delete stack**.

14d. In the confirmation window, click the **Delete** button.

And that's it! You've just used CloudFormation to deploy an EC2 instance with RStudio already installed. Once deployed, you walked through an example R code. Finally, you cleaned up all these resources by deleting the stack.

Extra Credit

Want to copy the results you generated back up to Amazon S3 for permanent storage? Just click on the **Terminal** tab and type the following commands:



```
[labuser@ip-172-31-38-247 ~]$ cd R-course/Basic\ Data\ Analysis/results/
[labuser@ip-172-31-38-247 results]$ ls
hctree.png  pca.png  significant_cause_of_death.Rdata  top10_region_heatmap.png
[labuser@ip-172-31-38-247 results]$ aws s3 mb s3://my-research-results-jsw
make_bucket: my-research-results-jsw
[labuser@ip-172-31-38-247 results]$ aws s3 cp . s3://my-research-results-jsw --recursive
upload: ./hctree.png to s3://my-research-results-jsw/hctree.png
upload: ./top10_region_heatmap.png to s3://my-research-results-jsw/top10_region_heatmap.png
upload: ./pca.png to s3://my-research-results-jsw/pca.png
upload: ./significant_cause_of_death.Rdata to s3://my-research-results-jsw/significant_cause_of_death.Rdata
[labuser@ip-172-31-38-247 results]$ aws s3 ls s3://my-research-results-jsw
2019-04-23 14:49:25      70638 hctree.png
2019-04-23 14:49:25      35075 pca.png
2019-04-23 14:49:25   33818319 significant_cause_of_death.Rdata
2019-04-23 14:49:25     251771 top10_region_heatmap.png
[labuser@ip-172-31-38-247 results]$
```

Extra Reading

One of the nice features about CloudFormation is that is allows you to describe your infrastructure end state in a configuration file that's easy to read. For the curious, here is the raw template used to deploy the RStudio server in this lab:

```
# Copyright 2018 Amazon.com, Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may not use this file except in
compliance with the License.
# A copy of the License is located at
#    http://aws.amazon.com/apache2.0/
# or in the "license" file accompanying this file. This file is distributed on an "AS IS" BASIS, WITHOUT
WARRANTIES OR CONDITIONS OF ANY KIND,
# either express or implied. See the License for the specific language governing permissions and limitations
under the License.


AWSTemplateFormatVersion: '2010-09-09'
Description: This CloudFormation Template deploys a complete OHDSI environment.  It
  depends on the OHDSI-VPC CloudFormation Template.



Parameters:
#  VpcId:
#    Type: AWS::EC2::VPC::Id
#    Description: VPC this server will reside in
  VPCSubnet:
    Description: The subnet in which you want your R-Studio server to be deployed.
    Type: AWS::EC2::Subnet::Id
  VPCId:
    Type: AWS::EC2::VPC::Id
  HomeDirectorySize:
    Description: The amount of encrypted disk space, in GBs, allocated to store R-Studio user's local data.
    Type: Number
    Default: 20
  InstanceType:
    Type: String
    Description: Instance type for the R-Studio server.
    AllowedValues:
      - t2.medium
      - t2.large
      - t2.xlarge
      - t2.2xlarge
      - t3.medium
      - t3.large
      - t3.xlarge
      - t3.2xlarge
      - m4.large
      - m4.xlarge
      - m4.2xlarge
      - m4.4xlarge
      - m4.10xlarge
      - m4.16xlarge
      - m5.large
      - m5.xlarge
      - m5.2xlarge
      - m5.4xlarge
      - m5.12xlarge
      - m5.24xlarge
      - c4.large
      - c4.xlarge
      - c4.2xlarge
      - c4.4xlarge
      - c4.8xlarge
      - c5.large
      - c5.xlarge
      - c5.2xlarge
      - c5.4xlarge
      - c5.9xlarge
      - c5.18xlarge
      - r4.large
      - r4.xlarge
      - r4.2xlarge
      - r4.4xlarge
      - r4.8xlarge
      - r4.16xlarge
      - r5.large
      - r5.xlarge
      - r5.2xlarge
      - r5.4xlarge
      - r5.8xlarge
      - r5.16xlarge
      - g2.2xlarge
      - g2.8xlarge
      - p2.xlarge
      - p2.8xlarge
      - p2.16xlarge
      - g3.4xlarge
      - g3.8xlarge
      - g3.16xlarge
    ConstraintDescription: Valid instance type in the t2, t3, m5, c5, r4, g2, p2, and g3 families
    Default: t2.xlarge
  UserList:
    Description: Provide a comma separated list of usernames and passwords (user1,pass1,user2,pass2) to
create on the R-Studio Server.
    Type: 'String'
    NoEcho: true
  KeyPair:
    Description: The EC2 Key Pair to use for the Atlas/WebAPI EC2 Instances.
    Type: AWS::EC2::KeyPair::KeyName
  AccessCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/([0-9]|[1-2][0-9]|3[0-2]))$
    Description: 'The CIDR IP range that is permitted to access your OHDSI servers. Note: A value of 0.0.0.0
/0 will allow access from ANY IP address.'
```

```
    Type: String
    Default: 0.0.0.0/0


Mappings:
  RegionMap:
    us-east-1:
      AMI: ami-467ca739
    us-east-2:
      AMI: ami-976152f2
    us-west-1:
      AMI: ami-46e1f226
    us-west-2:
      AMI: ami-6b8cef13
    ca-central-1:
      AMI: ami-2f39bf4b
    eu-west-1:
      AMI: ami-9cbe9be5
    eu-west-2:
      AMI: ami-c12dcda6
    eu-west-3:
      AMI: ami-cae150b7
    eu-central-1:
      AMI: ami-1b316af0
    sa-east-1:
      AMI: ami-f09dcc9c
    ap-south-1:
      AMI: ami-b46f48db
    ap-southeast-1:
      AMI: ami-64260718
    ap-southeast-2:
      AMI: ami-60a26a02
    ap-northeast-1:
      AMI: ami-28ddc154
    ap-northeast-2:
      AMI: ami-efaf0181


Resources:

  PublicSGSSL:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security Group for RStudio SSL
      VpcId: !Ref VPCId
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: '443'
          ToPort: '443'
          CidrIp: !Ref 'AccessCidr'

      Tags:
        - Key: Name
          Value: RStudio SSL Security Group

#IAM Roles for the RStudio Server
  RStudioRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - "arn:aws:iam::aws:policy/service-role/AmazonEC2RoleforSSM"
        - "arn:aws:iam::aws:policy/AmazonSageMakerFullAccess"
        - "arn:aws:iam::aws:policy/ComprehendMedicalFullAccess"
  RStudioRolePolicies:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: root
      PolicyDocument:
        Statement:
          - Effect: Allow
            Action:
              - "s3:GetObject"
              - "s3:ListObjects"
              - "s3:ListBucket"
              - "s3:PutObject"
              - "s3:CreateBucket"
            Resource: '*'
          - Effect: Allow
            Action:
              - "iam:GetRole"
              - "s3:ListAllMyBuckets"
            Resource: '*'
      Roles:
        - !Ref 'RStudioRole'
  RStudioInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
      Roles:
        - !Ref 'RStudioRole'


  RStudioInstance:
    Type: AWS::EC2::Instance
    Metadata:
      AWS::CloudFormation::Init:
        config:
          files:
              /etc/awslogs/awslogs.conf:
                content: !Sub |
                  [general]
                  state_file = /var/lib/awslogs/agent-state

                  [ohdsi-build-log]
                  file = /var/log/secure
                  log_group_name = RStudio-Audit-Log
                  log_stream_name = ${AWS::StackName}-var-log-secure
                mode: 000664
                owner: root
                group: root
              /etc/awslogs/awscli.conf:
                content: !Sub |
                  [plugins]
                  cwlogs = cwlogs
                  [default]
                  region = ${AWS::Region}
                mode: 000664
                owner: root
                group: root
              /etc/nginx/nginx.conf:
                content: !Sub |
                  #
                  # Ngxinx configuration file for secure websocket applications.
                  #
                  # - Listens on 80 (HTTP) and 443 (HTTPS)
                  # - Redirects all port 80 traffic to port 443
                  # - Reverse proxies requests to RStudio on port 8787.
                  #
                  events { }
                  http {
                  upstream node {
                    # Directs to the process with least number of connections.
                    least_conn;
                    # One failed response will take a server out of circulation for 20 seconds.
                    server 127.0.0.1:8787 fail_timeout=20s;
                  }

                  server {
                    # Listen on 80 and 443
                    listen 80;
                    listen 443 ssl;
                    # Self-signed certificate.
                    ssl_certificate /etc/pki/tls/certs/server.crt;
                    ssl_certificate_key /etc/pki/tls/certs/server.key;
                    # Certificate chained with a certificate authority bundle.
                    # ssl_certificate /etc/ssl/certs/example.com.chained.crt;
                    # ssl_certificate_key /etc/ssl/private/example.com.key;

                    # Redirect all non-SSL traffic to SSL.
                    if ($ssl_protocol = "") {
                      rewrite ^ https://$host$request_uri? permanent;
                    }

                    # Split off traffic to RStudio backend, and make sure that websockets
                    # are managed correctly.
                    location / {
                      proxy_pass http://localhost:8787/;
                      proxy_http_version 1.1;
                      proxy_set_header Upgrade websocket;
                      proxy_set_header Connection upgrade;
                    }

                  }
                  }
                mode: 000664
                owner: root
                group: root
    Properties:
      InstanceType: !Ref InstanceType
      KeyName: !Ref 'KeyPair'
      ImageId: !FindInMap
        - RegionMap
        - !Ref 'AWS::Region'
        - AMI
      IamInstanceProfile: !Ref RStudioInstanceProfile
      NetworkInterfaces:
        - AssociatePublicIpAddress: "true"
          DeviceIndex: "0"
          GroupSet:
            - Ref: "PublicSGSSL"
          SubnetId:
            Ref: "VPCSubnet"
      Tags:
        - Key: "Name"
          Value: !Sub "RStudio-${AWS::StackName}"
      BlockDeviceMappings:
          - DeviceName: "/dev/xvda"
            Ebs:
              VolumeType: "gp2"
              DeleteOnTermination: "true"
              VolumeSize: 50
          - DeviceName: "/dev/sdm"
            Ebs:
              VolumeType: "gp2"
              DeleteOnTermination: "true"
              Encrypted: "true"
              VolumeSize: !Ref HomeDirectorySize
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          RSTUDIO_URL="https://download2.rstudio.org/rstudio-server-rhel-1.1.463-x86_64.rpm"
          SHINY_URL="https://download3.rstudio.org/centos6.3/x86_64/shiny-server-1.5.9.923-x86_64.rpm"
          RSTUDIOPORT=8787
          users=${UserList}
          MIN_USER_ID=400 # default is 500 starting from 1.0.44, EMR hadoop user id is 498

          # Install SSM client
          yum install -y https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/linux_amd64/amazon-
ssm-agent.rpm
          restart amazon-ssm-agent

          sudo mkfs -t ext4 /dev/sdm
          mkdir /mnt/tmp
          sudo mount /dev/sdm /mnt/tmp
          cp -a /home/. /mnt/tmp
          umount /dev/sdm
          mount /dev/sdm /home
          echo "/dev/sdm /home ext4 defaults,nofail 0 2" >> /etc/fstab

          if [ ! -f /etc/pki/tls/certs/server.key ]; then
          openssl genrsa 2048 > server.key
          openssl req -new -key server.key -out csr.pem -subj "/C=US/ST=WA/L=Seattle/O=anon/OU=anon
/CN=selfsigned/emailAddress=selfsigned"
          openssl x509 -req -days 365 -in csr.pem -signkey server.key -out server.crt
          cp server.crt server.key /etc/pki/tls/certs/
          rm -f server.crt server.key csr.pem
          else
          echo "Already have a self-signed private key.  This must be an application redeployment"
          fi

          date > /tmp/rstudio_sparklyr_emr5.tmp
          export MAKE='make -j 8'
          sudo yum install -y nginx xorg-x11-xauth.x86_64 xorg-x11-server-utils.x86_64 xterm libXt libX11-
devel libXt-devel libcurl-devel git compat-gmp4 compat-libffi5 openssl-devel awslogs

          /opt/aws/bin/cfn-init --verbose --stack ${AWS::StackName} --resource RStudioInstance --region
${AWS::Region}
          sudo service awslogs start

          sudo yum install R R-core R-core-devel R-devel libxml2-devel -y
          if [ -f /usr/lib64/R/etc/Makeconf.rpmnew ]; then
            sudo cp /usr/lib64/R/etc/Makeconf.rpmnew /usr/lib64/R/etc/Makeconf
          fi
          if [ -f /usr/lib64/R/etc/ldpaths.rpmnew ]; then
            sudo cp /usr/lib64/R/etc/ldpaths.rpmnew /usr/lib64/R/etc/ldpaths
          fi

          mkdir /mnt/r-stuff
          cd /mnt/r-stuff

          #pushd .
          #mkdir R-latest
          #cd R-latest
          #wget https://cran.r-project.org/src/base/R-3/R-3.5.0.tar.gz
          #tar -xzf R-3.5.0.tar.gz
          #Trying out Python
          sudo yum install -y python36 python36-devel python36-pip
          sudo yum install -y gcc gcc-c++ gcc-gfortran
          sudo yum install -y readline-devel cairo-devel libpng-devel libjpeg-devel libtiff-devel postgresql-
devel
          #cd R-3*
          #./configure --with-readline=yes --enable-R-profiling=no --enable-memory-profiling=no --enable-R-
shlib --with-pic --prefix=/usr --with-x --with-libpng --with-jpeglib --with-cairo --enable-R-shlib --with-
recommended-packages=yes
          #make -j 8
          #sudo make install
          #sudo su << BASH_SCRIPT
          #echo 'export PATH=${!PWD}/bin:$PATH' >> /etc/profile
          #BASH_SCRIPT
          #popd

          #sudo sed -i 's/make/make -j 8/g' /usr/lib64/R/etc/Renviron

          # set unix environment variables
          sudo su << BASH_SCRIPT
          export JAVA_HOME=/etc/alternatives/jre
          ' >> /etc/profile
          BASH_SCRIPT
          sudo sh -c "source /etc/profile"

          # fix hadoop tmp permission
          #sudo chmod 777 -R /mnt/var/lib/hadoop/tmp

          # fix java binding - R and packages have to be compiled with the same java version as hadoop
          sudo R CMD javareconf


          # install rstudio
          RSTUDIO_FILE=$(basename $RSTUDIO_URL)
          wget $RSTUDIO_URL
          sudo yum install --nogpgcheck -y $RSTUDIO_FILE
          # change port - 8787 will not work for many companies
          sudo sh -c "echo 'www-port=$RSTUDIOPORT' >> /etc/rstudio/rserver.conf"
          sudo sh -c "echo 'auth-minimum-user-id=$MIN_USER_ID' >> /etc/rstudio/rserver.conf"
          sudo perl -p -i -e "s/= 5../= 100/g" /etc/pam.d/rstudio
          sudo rstudio-server stop || true
          sudo rstudio-server start

          sudo service nginx restart
          sudo chkconfig nginx on

          #sudo R --no-save << R_SCRIPT
          #install.packages(c('reshape2', 'tidyverse', 'devtools', 'gdtools', 'png'), repos="http://cran.
rstudio.com")
          #R_SCRIPT


          SHINY_FILE=$(basename $SHINY_URL)
              wget $SHINY_URL
              sudo yum install --nogpgcheck -y $SHINY_FILE

              sudo R --no-save <<R_SCRIPT
          install.packages(c('shiny','rmarkdown'),
          repos="http://cran.rstudio.com")
          R_SCRIPT

          sudo rm -f /tmp/rstudio_sparklyr_emr5.tmp


          sudo yum install -y cairo-devel
          #sudo yum install -y python-scipy
          #sudo pip install scipy
          #sudo pip install sklearn
          #sudo pip install torch torchvision
          #sudo pip install boto3==1.7.52
          #sudo pip install sagemaker
          #sudo pip install mxnet
          #sudo pip install pandas

          region=`curl http://169.254.169.254/latest/dynamic/instance-identity/document|grep region|awk -F\"
'{print $4}'`
          count=1
          for i in $(echo $users | sed "s/,/ /g")
          do
              if [ `expr $count % 2` -eq "1" ]; then
                username=$i
                let count+=1
                continue
              else
                sudo adduser $username
                sudo sh -c "echo '$i' | passwd --stdin $username"
                sudo -u $username mkdir /home/$username/.aws
                sudo -u $username bash -c 'echo "[default]
          region = '$region'" > /home/'$username'/.aws/config'

                # add the first users in the list to the sudoers file.
                if [ "$count" = "2" ]; then
                  echo "$username  ALL=(ALL:ALL) ALL" >> /etc/sudoers
                fi

                let count+=1
              fi
          done


Outputs:
  RStudioURL:
    Value: !Join ['', ['https://', !GetAtt 'RStudioInstance.PublicDnsName']]
```