amazon web services | Certified

Solutions Architect - Associate

lab

lab title

Setting up a NodeJS Web Server on AWS EC2

V1.01

Course title

AWS Certified Solutions Architect Associate

BackSpace

Table of Contents

Contents

Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the lastest version with any updates or corrections.

About the Lab

These lab notes are to support the instructional videos on Setting up a NodeJS Server on EC2 in the BackSpace AWS Certified Solutions Architect course.

We will first use the Identity and Access Management (IAM) service to create a user and a developers group for user. Permissions will be set for the developers group and users inside the group will inherit the permissions. We will also create a role with permissions that will allow our EC2 Linux server to access AWS resources within the account.

We will then:

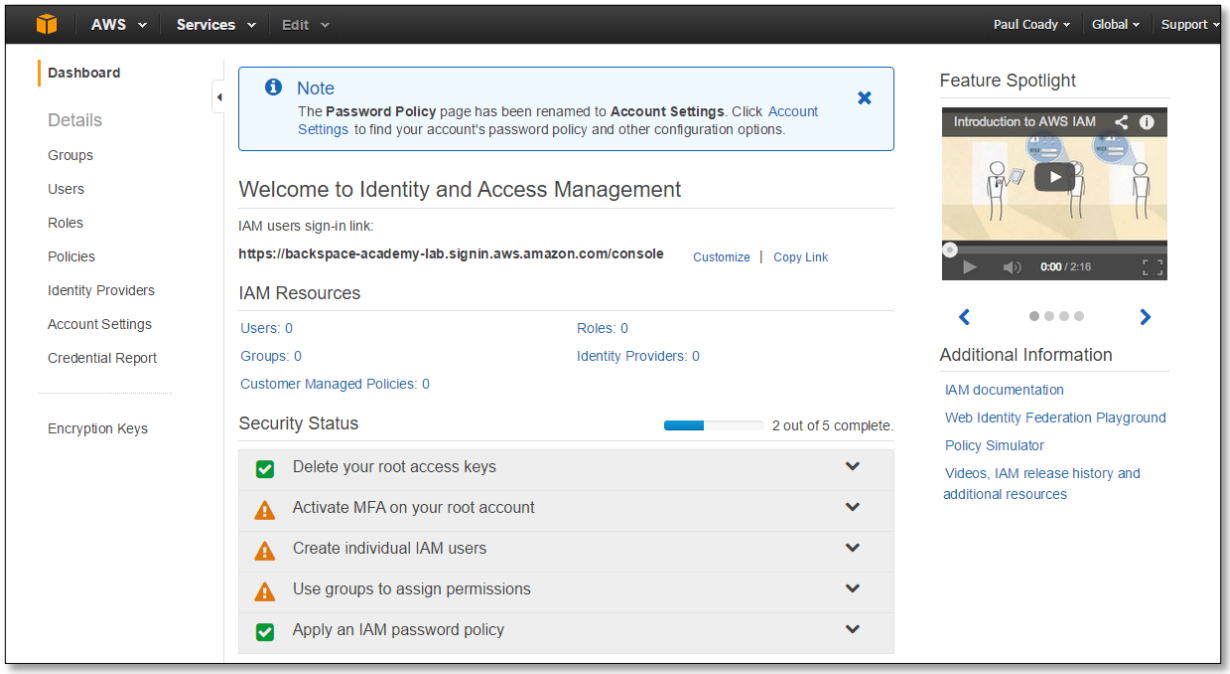
- Create an EC2 Linux instance and connect to that instance using SSH.
- Transfer files using SFTP.

Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the lastest version with any updates or corrections.

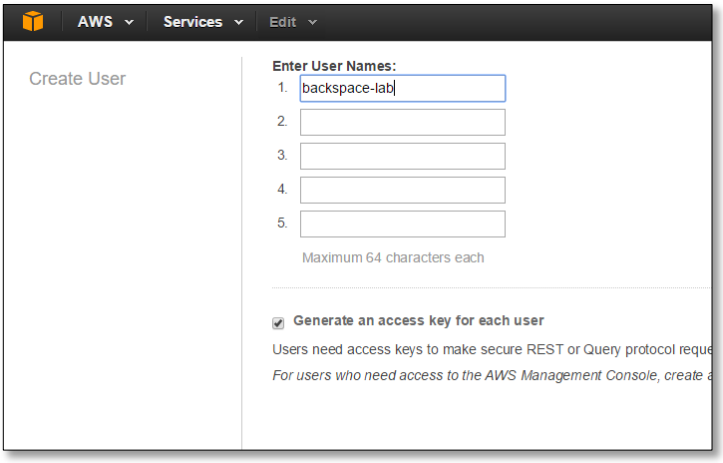
Creating an IAM User, Group and Role

In this section we will use the Identity and Access Management (IAM) service to create a user and a developers group for user. Permissions will be set for the developers group and users inside the group will inherit the permissions. We will also create a role with permissions that will allow our EC2 Linux server to access AWS resources within the account.

Select the IAM Console

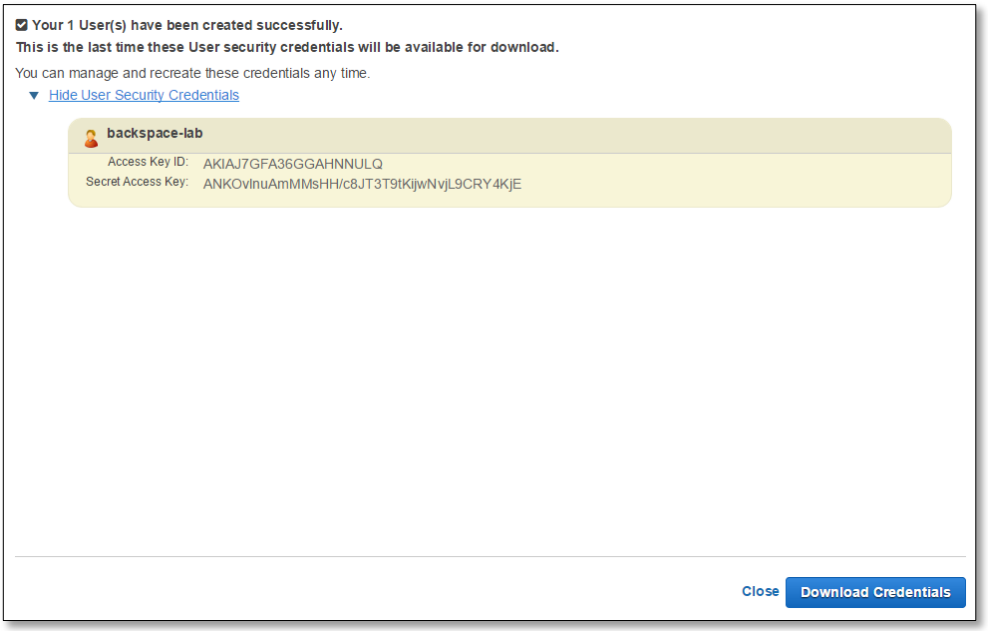


Click “Users” then “Create New Users”. Call the user backspace-lab.



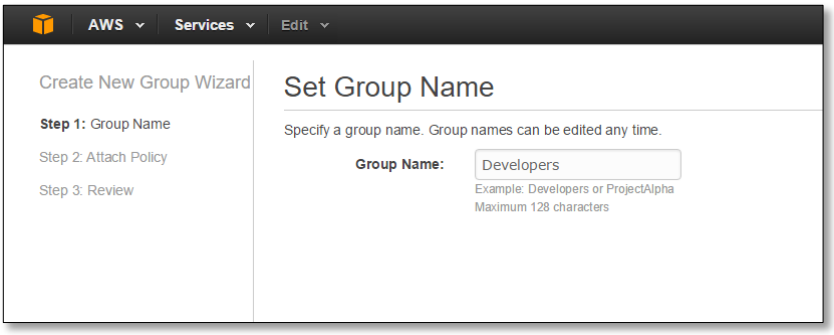
Click Create.

Click Download Credentials. Save this file somewhere we will need it later.



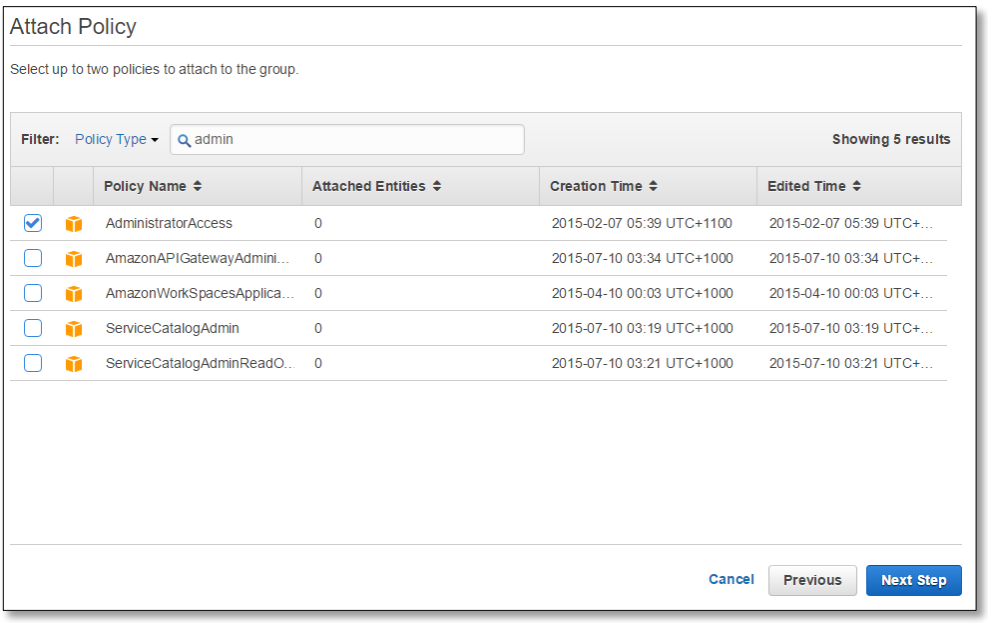
Click Close.

Click on “Groups” then select “Create New Group”. Call the group Developers.



Click “Next Step”.

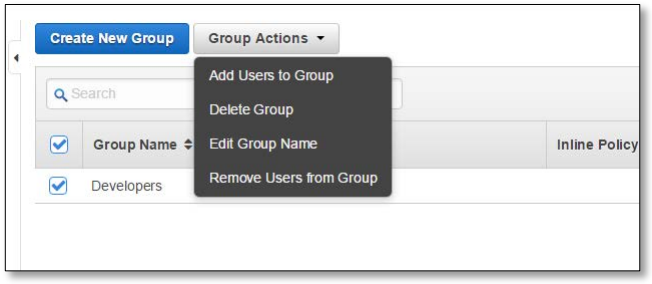
Search for Administrator Access and select.



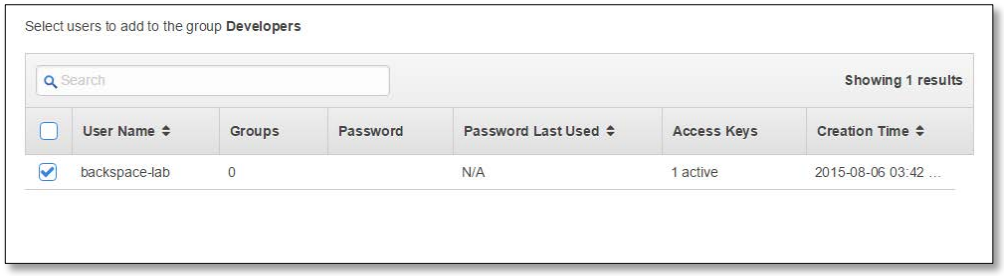
Click “Next Step”.

Click “Create Group”.

Select the new group and select “Add users to group” from Group Actions.



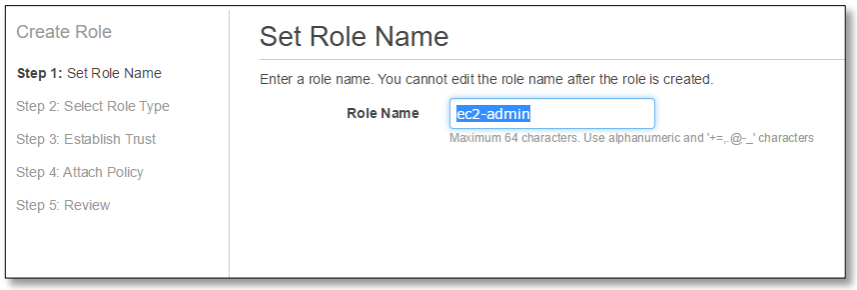
Select the backspace-lab user and click “Add users”



The user is now added to the Developers group and has inherited administrator access from the group.

Click on “Roles” and select “Create new role”.

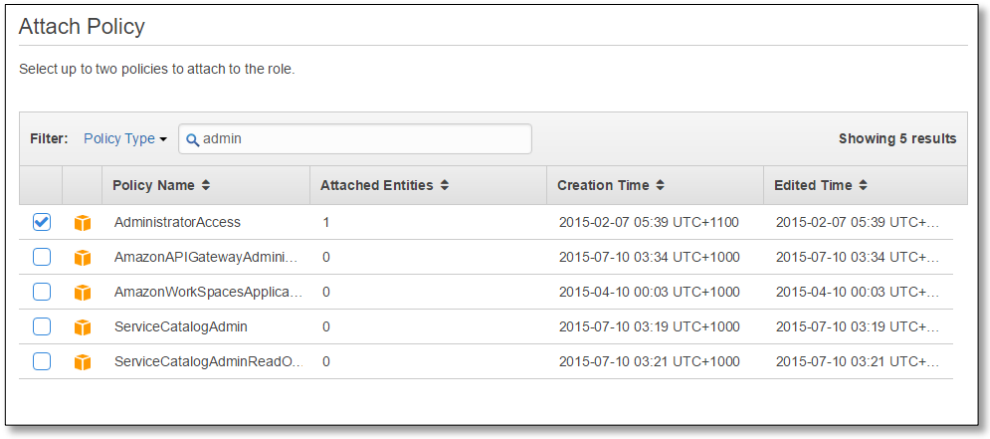
Call the role ec2-admin.



Click “Next Step”.

Select “Amazon EC2 - Allows EC2 instances to call AWS services on your behalf.”

Search for Administrator Access and select.



Now click “Create Role”

You have now created a role that can be assigned to an EC2 instance to access AWS resources.

Creating a Security Group

In this section we will create a security group that can be assigned to our EC2 NodeJS server to restrict access from the internet.

Go to the EC2 console.

Click on “Security Groups” and select “Create Security Group”.

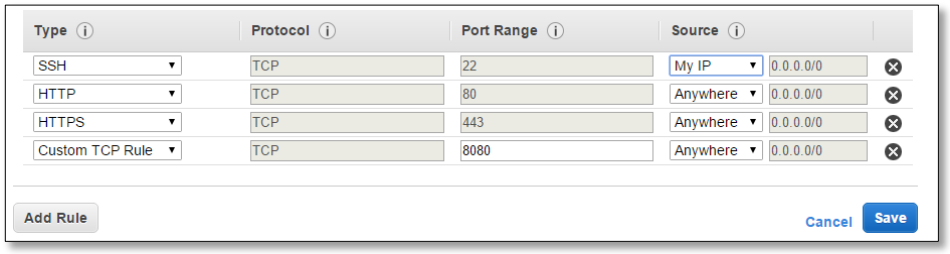
Call your security group WebServerSG.

Select the default VPC.

Select the inbound tab and add the following rules:

Inbound			
Source	**Protocol**	**Port Range**	**Comments**
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
0.0.0.0/0	TCP	8080	Allow inbound HTTP access to the web servers from anywhere
My IP (your home network's public IP address range)	TCP	22	Allow inbound SSH access to Linux instances from your home network (over the Internet gateway)

Outbound			
Destination	**Protocol**	**Port Range**	**Comments**
All traffic	TCP	All	Allow outbound traffic from the EC2 instance



Click “Create” to create the security group.

Creating an EC2 instance

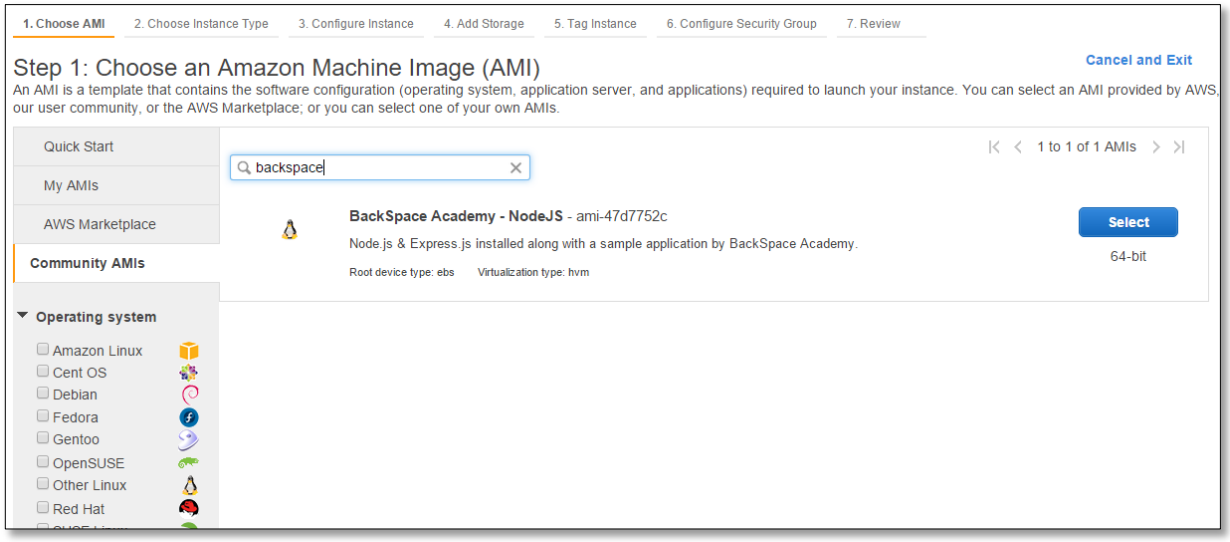
In this section we will create an EC2 instance from an AMI containing NodeJS. We will also bootstrap our instance to run a Linux bash script to set up firewall settings and update the operating system. We will also assign the IAM role and security group we created earlier.

Go to “Instances”

Click Launch Instance.

Select the Community AMIs tab.

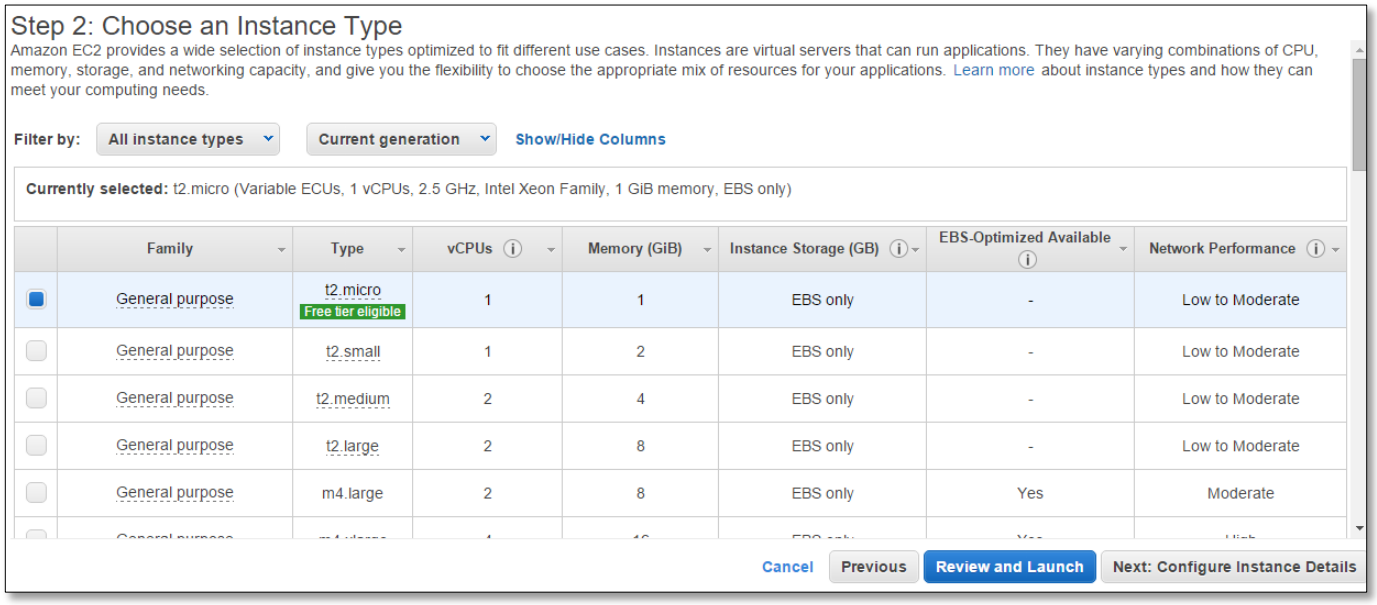
Search for the BackSpace NodeJS AMI.



Click Select.

Select a t2 micro instance.

Click “Next: Configure Instance Details”



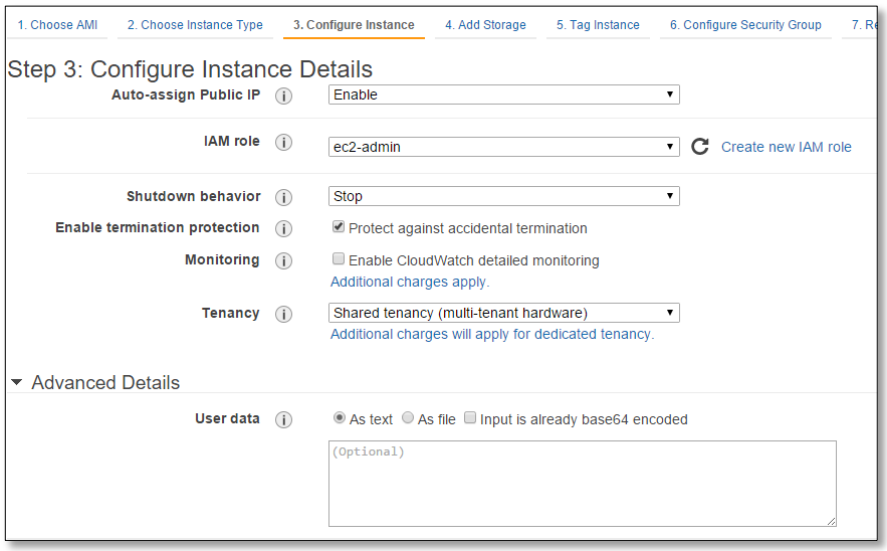
Select the default VPC

Enable “Auto assign public IP”

Select IAM role “ec2-admin”

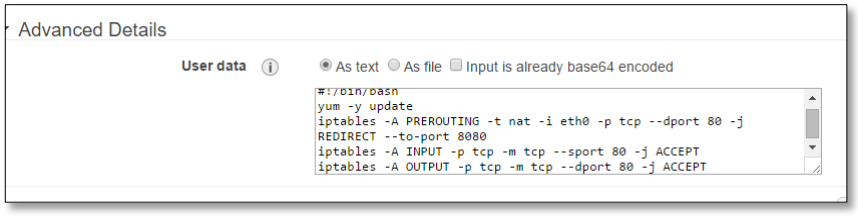
Check “Protect against accidental termination”

Expand the “Advanced Details” section.



In “User Data” we now have to add our bash script to set up the firewall settings that is run when the instance is launched:

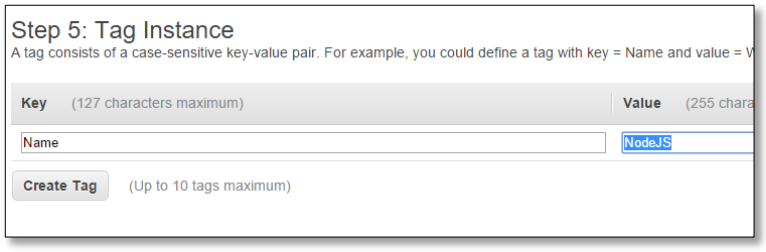
```
#!/bin/bash
yum -y update
iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
iptables -A INPUT -p tcp -m tcp --sport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT
```



Click “Next add storage”

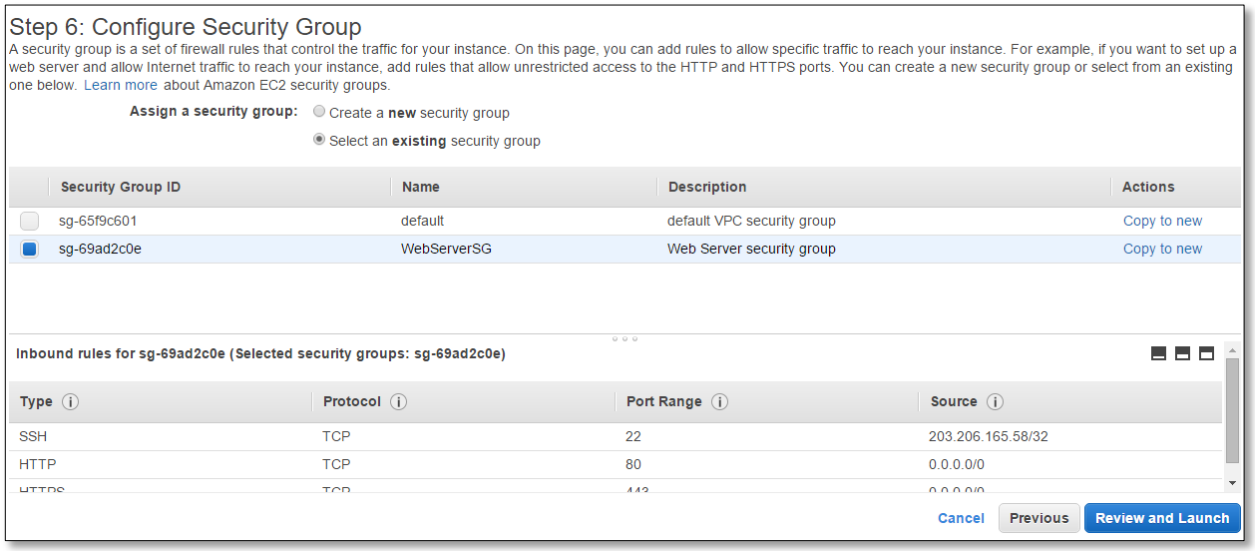
Click “Next tag instance”

Give it the name NodeJS

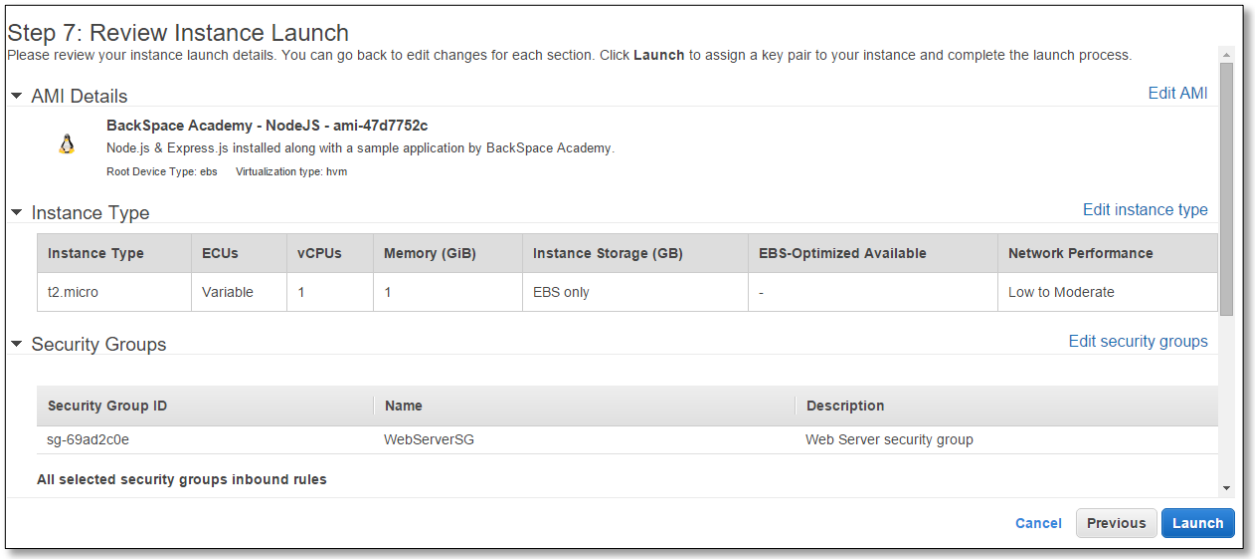


Click “Next configure security group”

Select your existing WebServerSG you created earlier.



Click “Review and Launch”



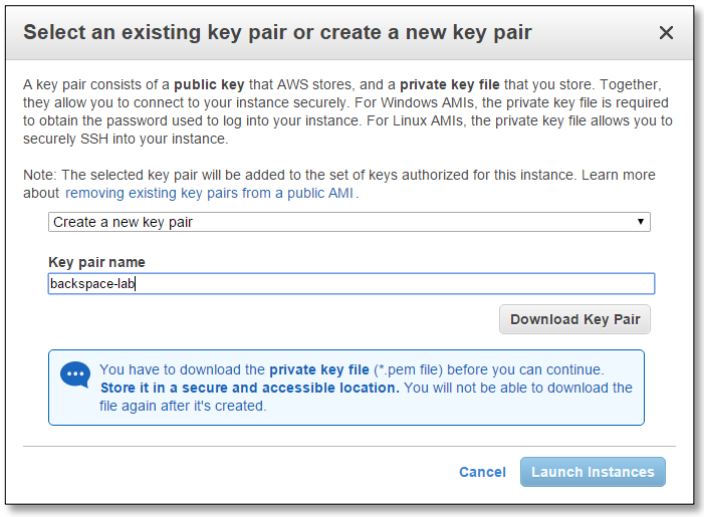
Click “Launch”

Select “Create a new key pair”

Call the key pair backspace-lab.

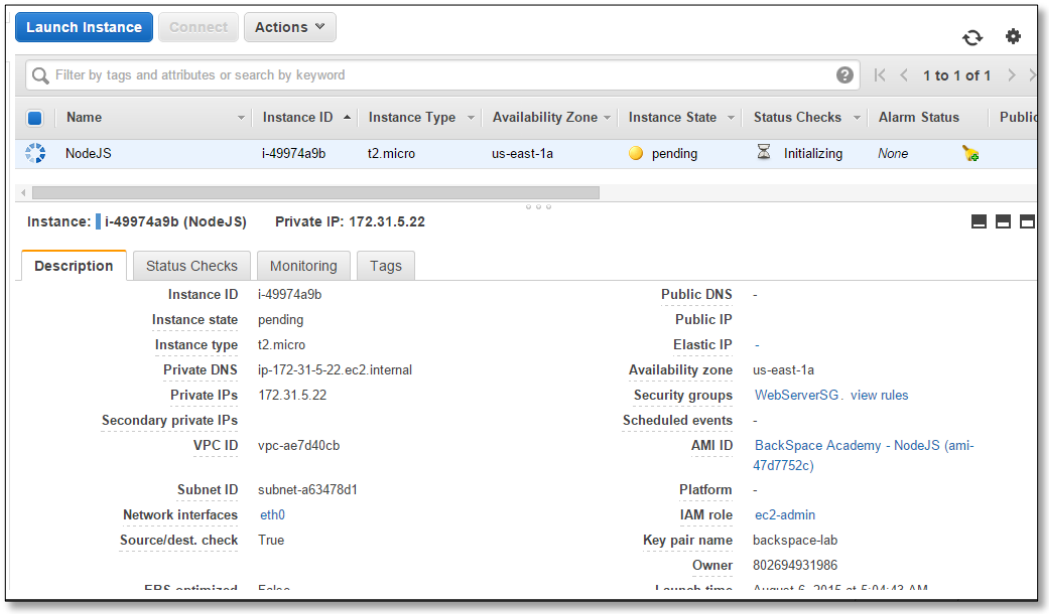
Create a directory on your windows system at C:\KeyPairs

Download the key backspace-lab.pem file to C:\KeyPairs



Click “Launch Instance”

Click “View Instance”



You have now created an EC2 server ready to go with NodeJS, Express and the AWS SDK.

Connecting to your EC2 instance using SSH

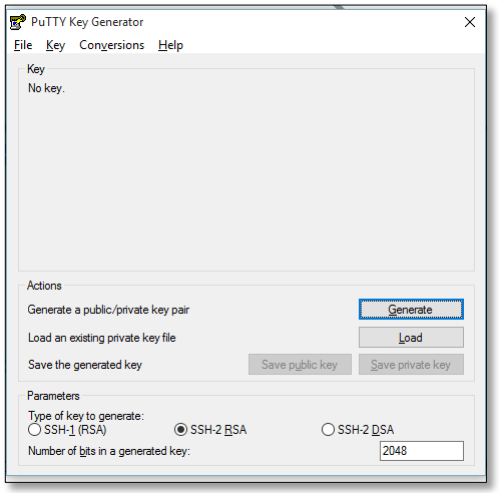
In this section we will connect from our Windows desktop to our EC2 instance using SSH and Putty. Mac OSX and Linux have SSH support without installing an additional client software.

Go to http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html and download the following executable files:

Putty (putty.exe)

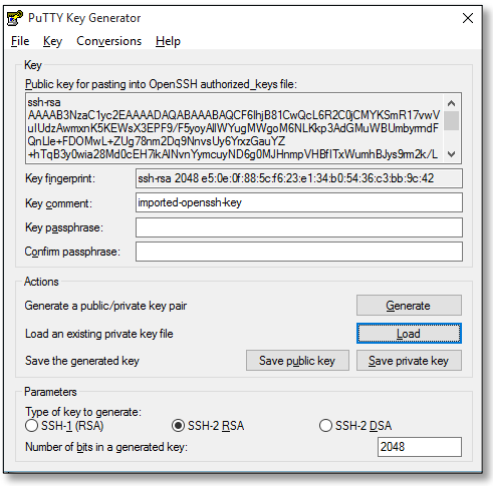
Putty Key Generator (puttygen.exe)

When they have downloaded run puttygen.exe



We need to convert our backspace-lab.pem to a ppk file suitable for Putty.

Click load and select “All files” and select the backspace-lab.pem from C:\KeyPairs



Click “Save Private Key”

Click “yes” to save without passphrase

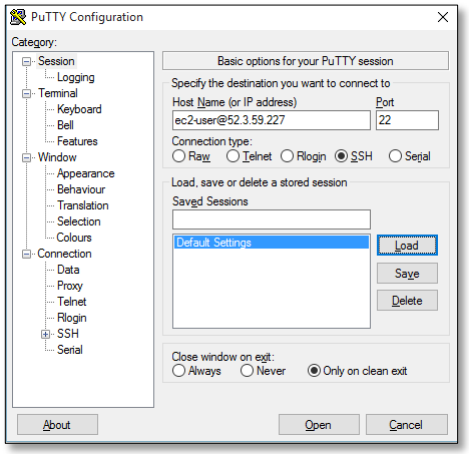
Save as backspace-lab to C:\KeyPairs

Close Puttygen

Go back to the EC2 console and copy your instances Public IP

Now run Putty.exe

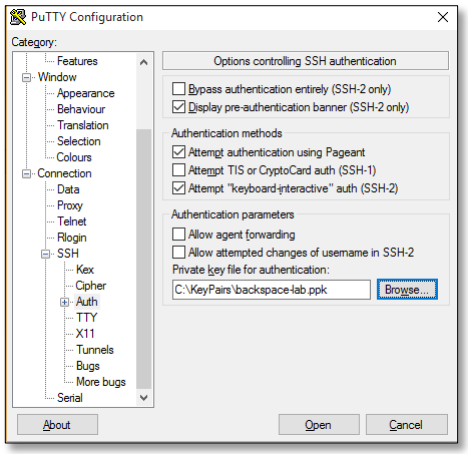
Input the hostname as ec2-user@(your Public IP) and port as 22



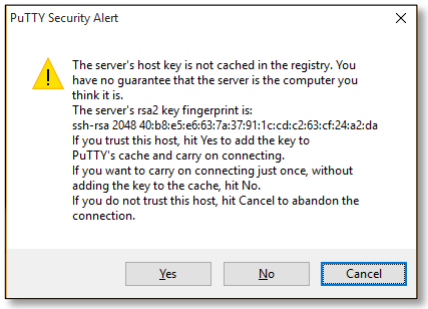
Click on SSH in the directory tree to expand.

Click on Auth in the directory tree.

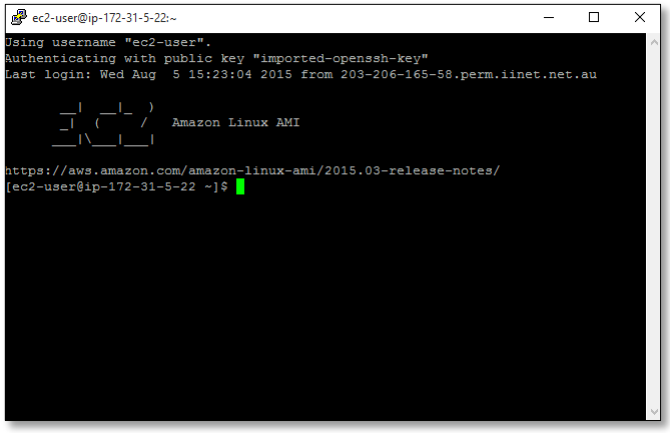
Click on “browse” and select the backspace-lab.ppk file



Click “Open”



Click “Yes”

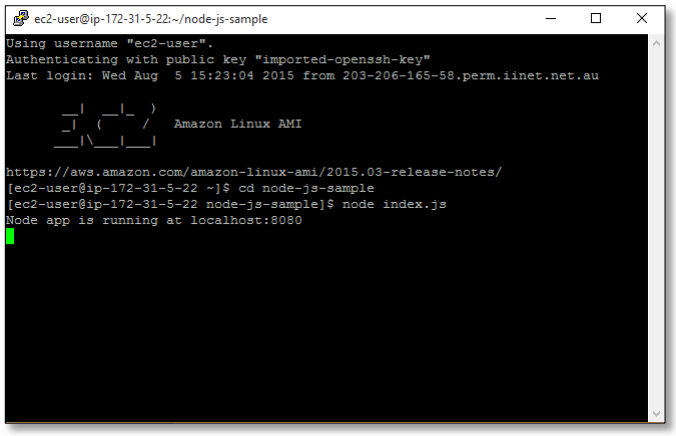


You are now connected to your EC2 instance.

Now run the sample NodeJS app with the following commands:

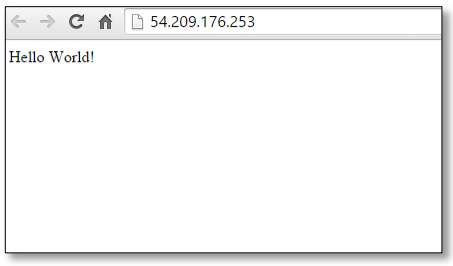
cd node-js-sample

node index.js



Your NodeJS app is now running.

Point your browser to your instance Public IP address and you will see the standard “Hello World!”

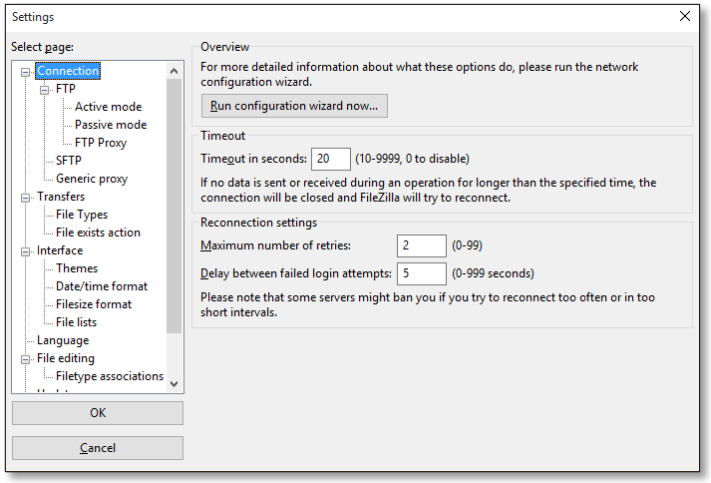


Transferring files to an EC2 instance using SFTP

In this section we will set up FileZilla to allow us to transfer files to our EC2 instance using SFTP. The instructions are for Windows although FileZilla is available for Mac OSX and Linux also.

Open FileZilla

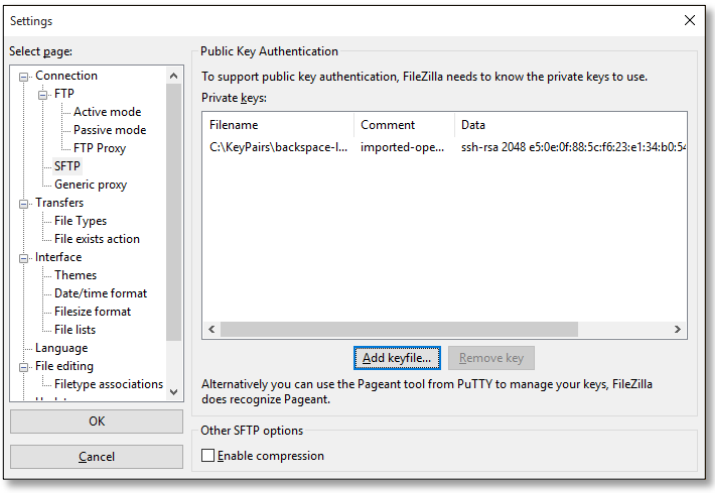
Go to “Edit” -> “Settings”



Click on “SFTP”

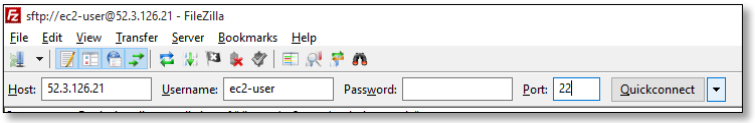
Click “Add Keyfile”

Select the backspace-lab.ppk (not pem) file.



Click OK

Enter your EC2 instance public IP, username ec2-user and port 22.



Click “Quick Connect”

You will then be connected to the EC2 instance.

Navigate to the node-js-sample folder to see the sample app.