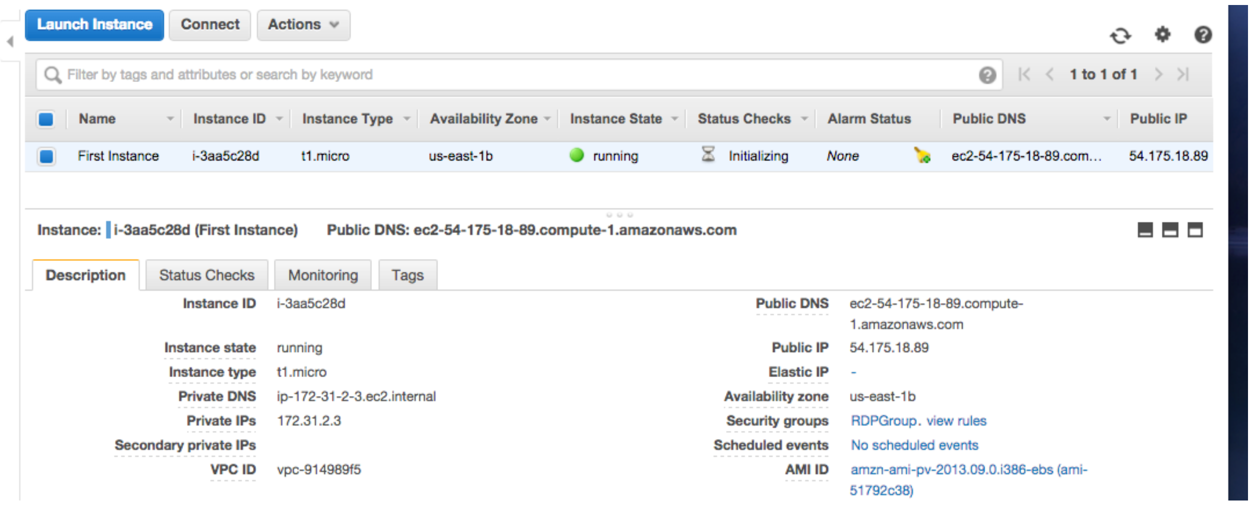
HU Extension Assignment 09 FINAL E-90 Cloud Computing

Problem 1:
Start with **Amazon Linux AMI 2013.09** - ami-51792c38 (32-bit). You can actually start with any Linux or Windows AMI, if you feel comfortable with that AMI. If there is a need, install Apache Web server and PHP framework. Again, you are welcome to work with any other technology. Add either index.php I used in class or some other active page that would demonstrate that a request from your browser ended up on a particular server (AWS instance). If you are using index.php, remove the greeting. Leave only the IP address as the response produced by that file. Create a new AMI based on this instance. Create 3 (three) instances based on your new AMI. Out of those 3, place one instance in the same availability zone as the original instance and the remaining two in another availability zone. When creating new instances, please make sure that you check "Enable Cloud Watch detailed monitoring". Create a load balancer and associate the original and 3 new instances with that load balancer. When creating the load balancer as the URL used for health check does specify /index.php or some other page you have added to your server. Demonstrate that the load balancer distributes incoming requests to different instances with approximately equal frequency. You could for example use just enabled Cloud Watch monitoring to show that after a moderately large number of requests sent to the index.php or a similar page, the usage pattern on all servers looks similar. You can do it all using AWS Console.
Points: [35]

Create a new AWS EC2 instance with Amazon Linux AMI 2013.09 - ami-51792c38 (32-bit)

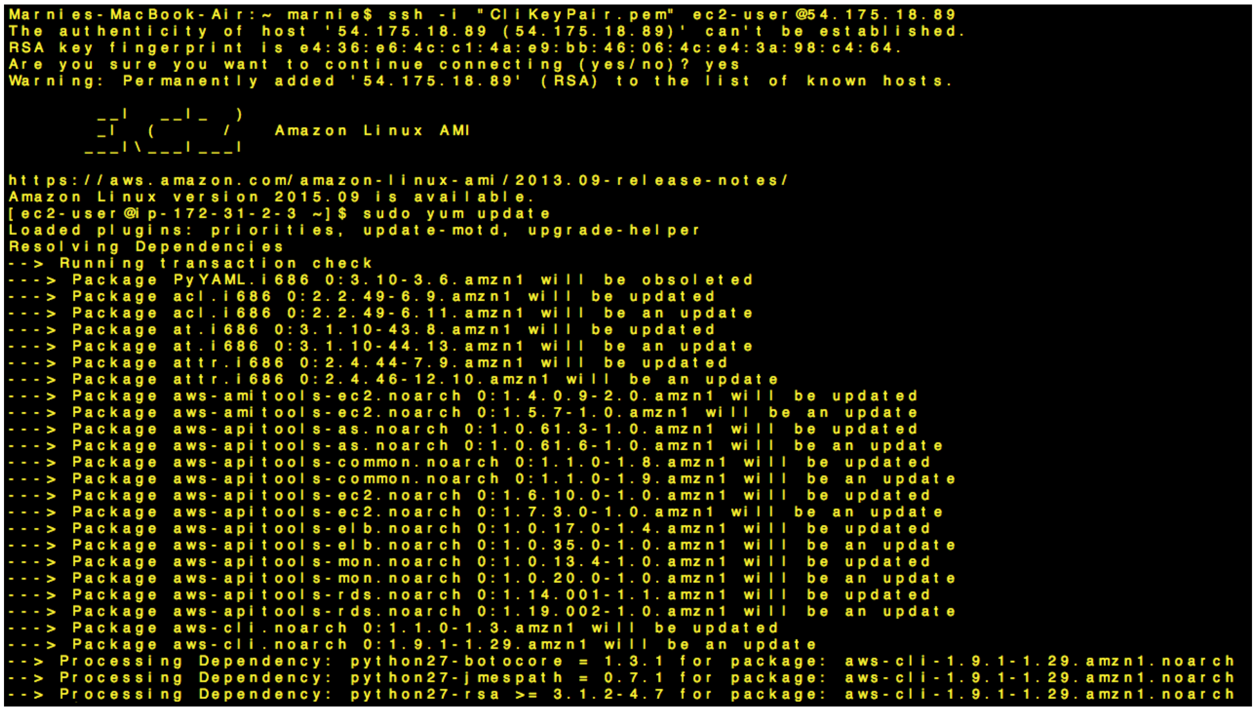


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
First Instance	i-3aa5c28d	t1.micro	us-east-1b	running	Initializing	None	ec2-54-175-18-89.com...	54.175.18.89

Instance: i-3aa5c28d (First Instance) Public DNS: ec2-54-175-18-89.compute-1.amazonaws.com

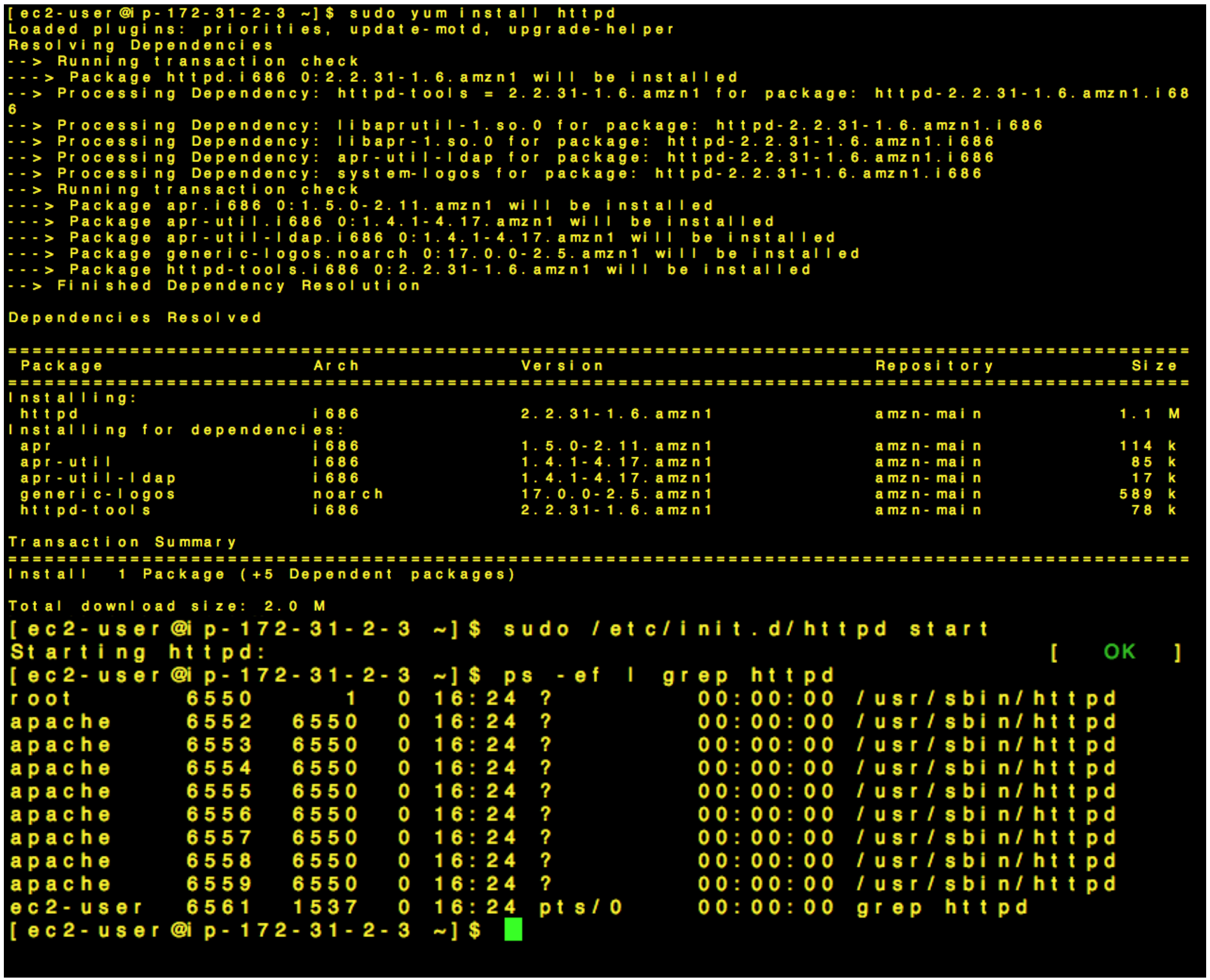
Instance ID	i-3aa5c28d	Public DNS	ec2-54-175-18-89.compute-1.amazonaws.com
Instance state	running	Public IP	54.175.18.89
Instance type	t1.micro	Elastic IP	-
Private DNS	ip-172-31-2-3.ec2.internal	Availability zone	us-east-1b
Private IPs	172.31.2.3	Security groups	RDPGroup. view rules
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-914989f5	AMI ID	amzn-ami-pv-2013.09.0.i386-ebs (ami-51792c38)

Connect to your instance and run sudo yum update



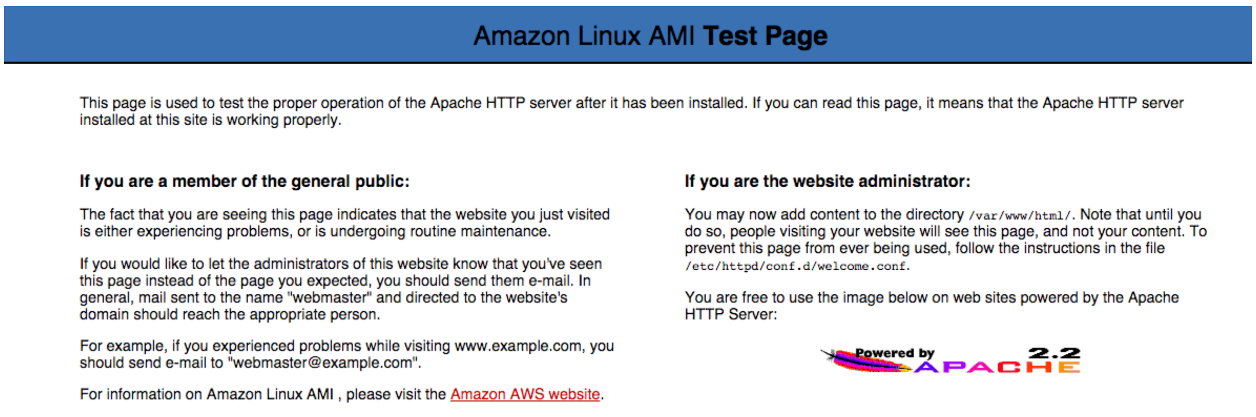
```
Marnies-MacBook-Air:~ marnie$ ssh -i "CliKeyPair.pem" ec2-user@54.175.18.89
The authenticity of host '54.175.18.89 (54.175.18.89)' can't be established.
RSA key fingerprint is e4:36:e6:4c:c1:4a:e9:bb:46:06:4c:e4:3a:98:c4:64.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.175.18.89' (RSA) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2013.09-release-notes/
Amazon Linux version 2015.09 is available.
[ec2-user@ip-172-31-2-3 ~]$ sudo yum update
Loaded plugins: priorities, update-motd, upgrade-helper
Resolving Dependencies
--> Running transaction check
---> Package PyYAML.i686 0:3.10-3.6.amzn1 will be obsoleted
---> Package acl.i686 0:2.2.49-6.9.amzn1 will be updated
---> Package acl.i686 0:2.2.49-6.11.amzn1 will be an update
---> Package at.i686 0:3.1.10-43.8.amzn1 will be updated
---> Package at.i686 0:3.1.10-44.13.amzn1 will be an update
---> Package attr.i686 0:2.4.44-7.9.amzn1 will be updated
---> Package attr.i686 0:2.4.46-12.10.amzn1 will be an update
---> Package aws-amitools-ec2.noarch 0:1.4.0.9-2.0.amzn1 will be updated
---> Package aws-amitools-ec2.noarch 0:1.5.7-1.0.amzn1 will be an update
---> Package aws-apitools-as.noarch 0:1.0.61.3-1.0.amzn1 will be updated
---> Package aws-apitools-as.noarch 0:1.0.61.6-1.0.amzn1 will be an update
---> Package aws-apitools-common.noarch 0:1.1.0-1.8.amzn1 will be updated
---> Package aws-apitools-common.noarch 0:1.1.0-1.9.amzn1 will be an update
---> Package aws-apitools-ec2.noarch 0:1.6.10.0-1.0.amzn1 will be updated
---> Package aws-apitools-ec2.noarch 0:1.7.3.0-1.0.amzn1 will be an update
---> Package aws-apitools-elb.noarch 0:1.0.17.0-1.4.amzn1 will be updated
---> Package aws-apitools-elb.noarch 0:1.0.35.0-1.0.amzn1 will be an update
---> Package aws-apitools-mon.noarch 0:1.0.13.4-1.0.amzn1 will be updated
---> Package aws-apitools-mon.noarch 0:1.0.20.0-1.0.amzn1 will be an update
---> Package aws-apitools-rds.noarch 0:1.14.001-1.1.amzn1 will be updated
---> Package aws-apitools-rds.noarch 0:1.19.002-1.0.amzn1 will be an update
---> Package aws-cli.noarch 0:1.1.0-1.3.amzn1 will be updated
---> Package aws-cli.noarch 0:1.9.1-1.29.amzn1 will be an update
--> Processing Dependency: python27-botocore = 1.3.1 for package: aws-cli-1.9.1-1.29.amzn1.noarch
--> Processing Dependency: python27-jmespath = 0.7.1 for package: aws-cli-1.9.1-1.29.amzn1.noarch
--> Processing Dependency: python27-rsa >= 3.1.2-4.7 for package: aws-cli-1.9.1-1.29.amzn1.noarch
```

Install Apache



```
[ec2-user@ip-172-31-2-3 ~]$ sudo yum install httpd
Loaded plugins: priorities, update-motd, upgrade-helper
Resolving Dependencies
--> Running transaction check
---> Package httpd.i686 0:2.2.31-1.6.amzn1 will be installed
--> Processing Dependency: httpd-tools = 2.2.31-1.6.amzn1 for package: httpd-2.2.31-1.6.amzn1.i68
6
--> Processing Dependency: libaprutil-1.so.0 for package: httpd-2.2.31-1.6.amzn1.i686
--> Processing Dependency: libapr-1.so.0 for package: httpd-2.2.31-1.6.amzn1.i686
--> Processing Dependency: apr-util-ldap for package: httpd-2.2.31-1.6.amzn1.i686
--> Processing Dependency: system-logos for package: httpd-2.2.31-1.6.amzn1.i686
--> Running transaction check
---> Package apr.i686 0:1.5.0-2.11.amzn1 will be installed
---> Package apr-util.i686 0:1.4.1-4.17.amzn1 will be installed
---> Package apr-util-ldap.i686 0:1.4.1-4.17.amzn1 will be installed
---> Package generic-logos.noarch 0:17.0.0-2.5.amzn1 will be installed
---> Package httpd-tools.i686 0:2.2.31-1.6.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================
 Package                  Arch              Version                      Repository        Size
================================================================================================
Installing:
 httpd                    i686              2.2.31-1.6.amzn1             amzn-main        1.1 M
Installing for dependencies:
 apr                      i686              1.5.0-2.11.amzn1             amzn-main        114 k
 apr-util                 i686              1.4.1-4.17.amzn1             amzn-main         85 k
 apr-util-ldap            i686              1.4.1-4.17.amzn1             amzn-main         17 k
 generic-logos            noarch            17.0.0-2.5.amzn1             amzn-main        589 k
 httpd-tools              i686              2.2.31-1.6.amzn1             amzn-main         78 k

Transaction Summary
================================================================================================
Install  1 Package (+5 Dependent packages)

Total download size: 2.0 M
[ec2-user@ip-172-31-2-3 ~]$ sudo /etc/init.d/httpd start
Starting httpd:                                            [  OK  ]
[ec2-user@ip-172-31-2-3 ~]$ ps -ef | grep httpd
root      6550     1  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6552  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6553  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6554  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6555  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6556  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6557  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6558  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
apache    6559  6550  0 16:24 ?        00:00:00 /usr/sbin/httpd
ec2-user  6561  1537  0 16:24 pts/0    00:00:00 grep httpd
[ec2-user@ip-172-31-2-3 ~]$
```

Test access to Web Server



Amazon Linux AMI **Test Page**

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

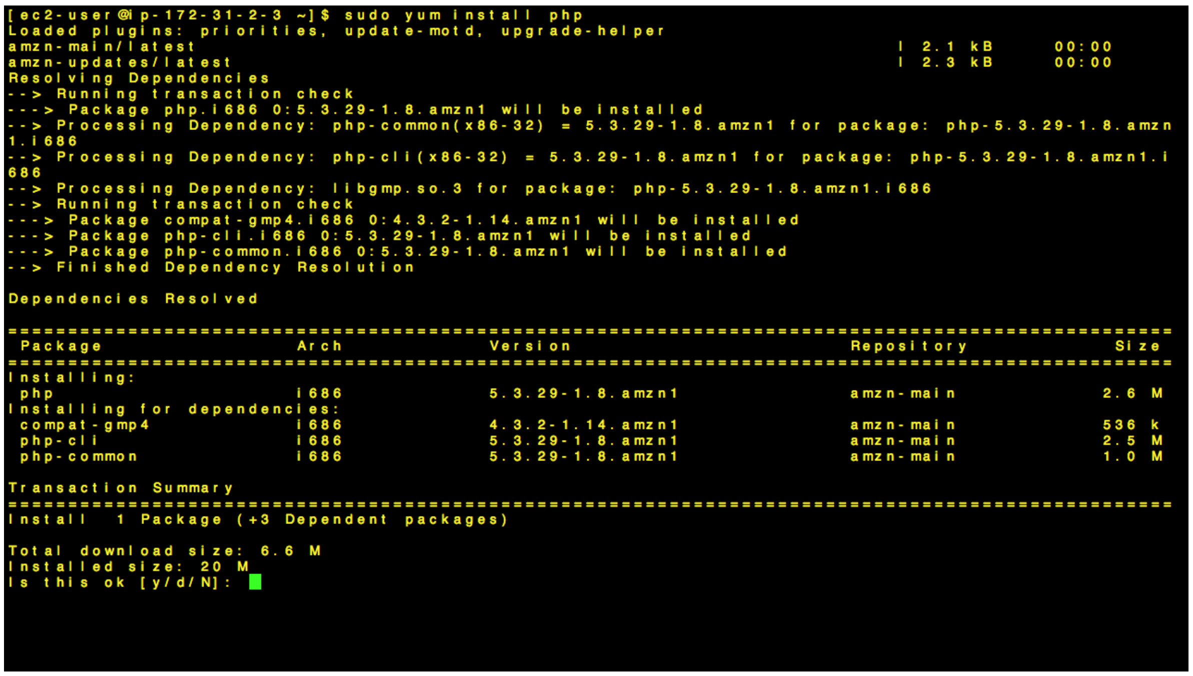
For information on Amazon Linux AMI , please visit the Amazon AWS website.

If you are the website administrator:

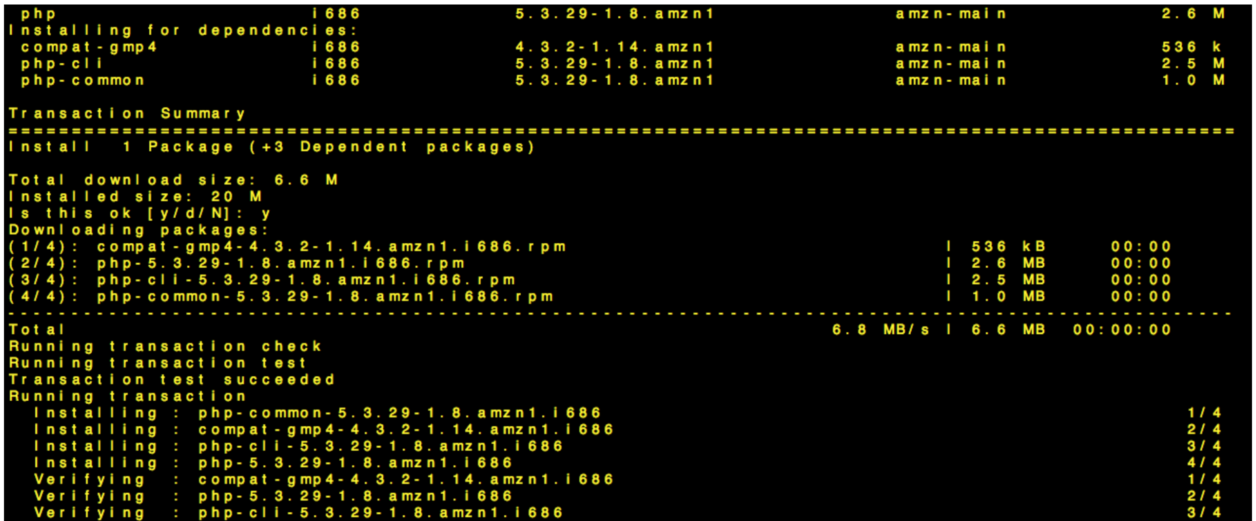
You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:

Install PHP

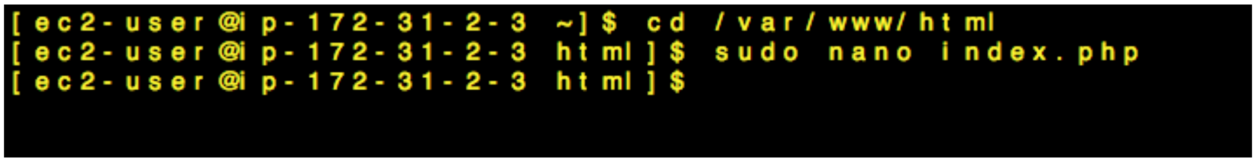


```
[ec2-user@ip-172-31-2-3 ~]$ sudo yum install php
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-main/latest                                                        | 2.1 kB     00:00
amzn-updates/latest                                                     | 2.3 kB     00:00
Resolving Dependencies
--> Running transaction check
---> Package php.i686 0:5.3.29-1.8.amzn1 will be installed
--> Processing Dependency: php-common(x86-32) = 5.3.29-1.8.amzn1 for package: php-5.3.29-1.8.amzn
1.i686
--> Processing Dependency: php-cli(x86-32) = 5.3.29-1.8.amzn1 for package: php-5.3.29-1.8.amzn1.i
686
--> Processing Dependency: libgmp.so.3 for package: php-5.3.29-1.8.amzn1.i686
--> Running transaction check
---> Package compat-gmp4.i686 0:4.3.2-1.14.amzn1 will be installed
---> Package php-cli.i686 0:5.3.29-1.8.amzn1 will be installed
---> Package php-common.i686 0:5.3.29-1.8.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=================================================================================================
 Package                 Arch            Version                       Repository           Size
=================================================================================================
Installing:
 php                     i686            5.3.29-1.8.amzn1              amzn-main           2.6 M
Installing for dependencies:
 compat-gmp4             i686            4.3.2-1.14.amzn1              amzn-main           536 k
 php-cli                 i686            5.3.29-1.8.amzn1              amzn-main           2.5 M
 php-common              i686            5.3.29-1.8.amzn1              amzn-main           1.0 M

Transaction Summary
=================================================================================================
Install  1 Package (+3 Dependent packages)

Total download size: 6.6 M
Installed size: 20 M
Is this ok [y/d/N]:
```

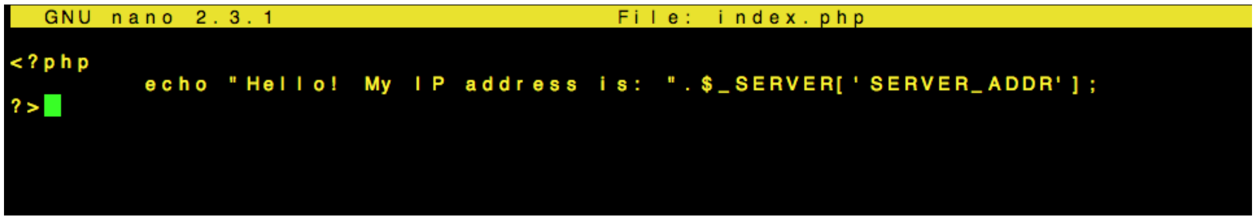


```
 php                     i686            5.3.29-1.8.amzn1              amzn-main           2.6 M
Installing for dependencies:
 compat-gmp4             i686            4.3.2-1.14.amzn1              amzn-main           536 k
 php-cli                 i686            5.3.29-1.8.amzn1              amzn-main           2.5 M
 php-common              i686            5.3.29-1.8.amzn1              amzn-main           1.0 M

Transaction Summary
=================================================================================================
Install  1 Package (+3 Dependent packages)

Total download size: 6.6 M
Installed size: 20 M
Is this ok [y/d/N]: y
Downloading packages:
(1/4): compat-gmp4-4.3.2-1.14.amzn1.i686.rpm                     | 536 kB   00:00
(2/4): php-5.3.29-1.8.amzn1.i686.rpm                             | 2.6 MB   00:00
(3/4): php-cli-5.3.29-1.8.amzn1.i686.rpm                         | 2.5 MB   00:00
(4/4): php-common-5.3.29-1.8.amzn1.i686.rpm                      | 1.0 MB   00:00
-------------------------------------------------------------------------------------------------
Total                                                   6.8 MB/s | 6.6 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : php-common-5.3.29-1.8.amzn1.i686                                              1/4
  Installing : compat-gmp4-4.3.2-1.14.amzn1.i686                                             2/4
  Installing : php-cli-5.3.29-1.8.amzn1.i686                                                 3/4
  Installing : php-5.3.29-1.8.amzn1.i686                                                     4/4
  Verifying  : compat-gmp4-4.3.2-1.14.amzn1.i686                                             1/4
  Verifying  : php-5.3.29-1.8.amzn1.i686                                                     2/4
  Verifying  : php-cli-5.3.29-1.8.amzn1.i686                                                 3/4
```

Create an index.php page to display each instance's private IP address when viewed through a browser

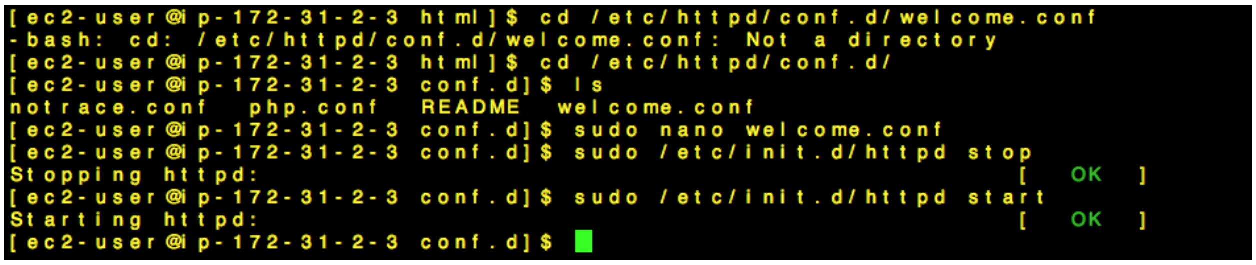


```
[ec2-user@ip-172-31-2-3 ~]$ cd /var/www/html
[ec2-user@ip-172-31-2-3 html]$ sudo nano index.php
[ec2-user@ip-172-31-2-3 html]$
```

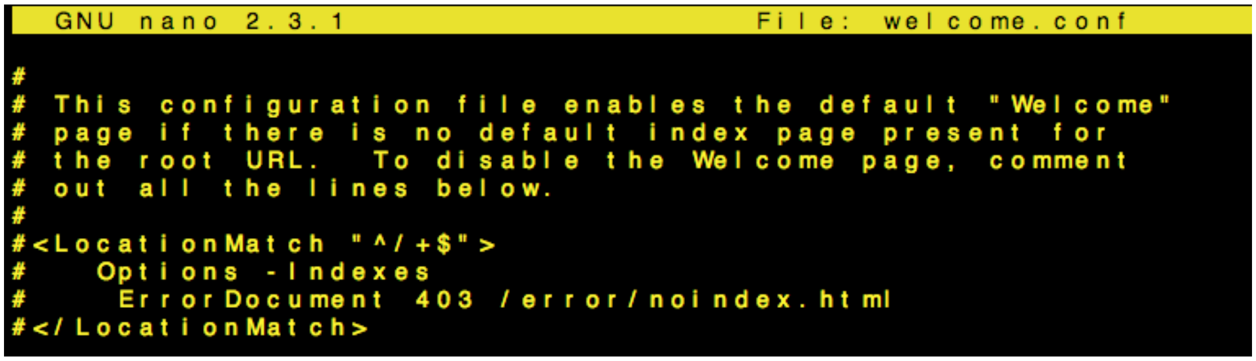


```
  GNU nano 2.3.1                         File: index.php

<?php
        echo "Hello! My IP address is: ".$_SERVER['SERVER_ADDR'];
?>
```

Configure Server to make index.php default home page

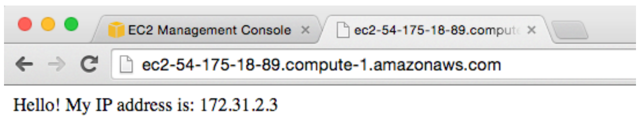


```
[ec2-user@ip-172-31-2-3 html]$ cd /etc/httpd/conf.d/welcome.conf
-bash: cd: /etc/httpd/conf.d/welcome.conf: Not a directory
[ec2-user@ip-172-31-2-3 html]$ cd /etc/httpd/conf.d/
[ec2-user@ip-172-31-2-3 conf.d]$ ls
notrace.conf  php.conf  README  welcome.conf
[ec2-user@ip-172-31-2-3 conf.d]$ sudo nano welcome.conf
[ec2-user@ip-172-31-2-3 conf.d]$ sudo /etc/init.d/httpd stop
Stopping httpd:                                            [  OK  ]
[ec2-user@ip-172-31-2-3 conf.d]$ sudo /etc/init.d/httpd start
Starting httpd:                                            [  OK  ]
[ec2-user@ip-172-31-2-3 conf.d]$
```



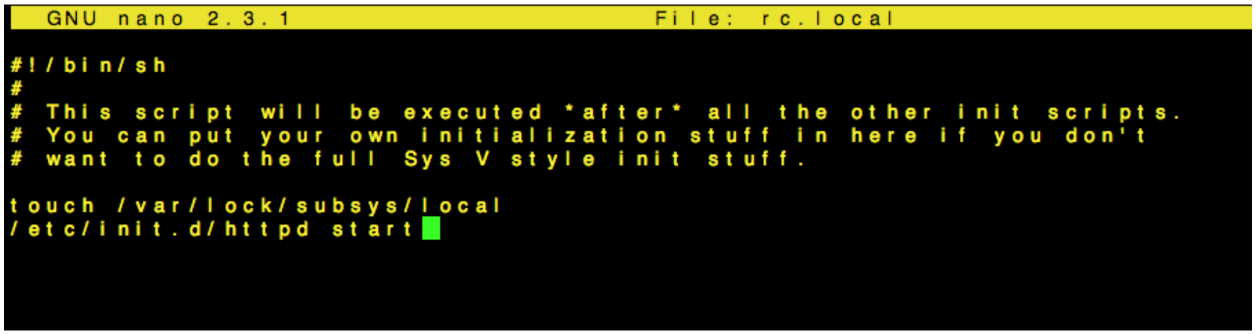
```
  GNU nano 2.3.1                         File: welcome.conf

#
# This configuration file enables the default "Welcome"
# page if there is no default index page present for
# the root URL.  To disable the Welcome page, comment
# out all the lines below.
#
#<LocationMatch "^/+$">
#    Options -Indexes
#     ErrorDocument 403 /error/noindex.html
#</LocationMatch>
```

Test page on server again



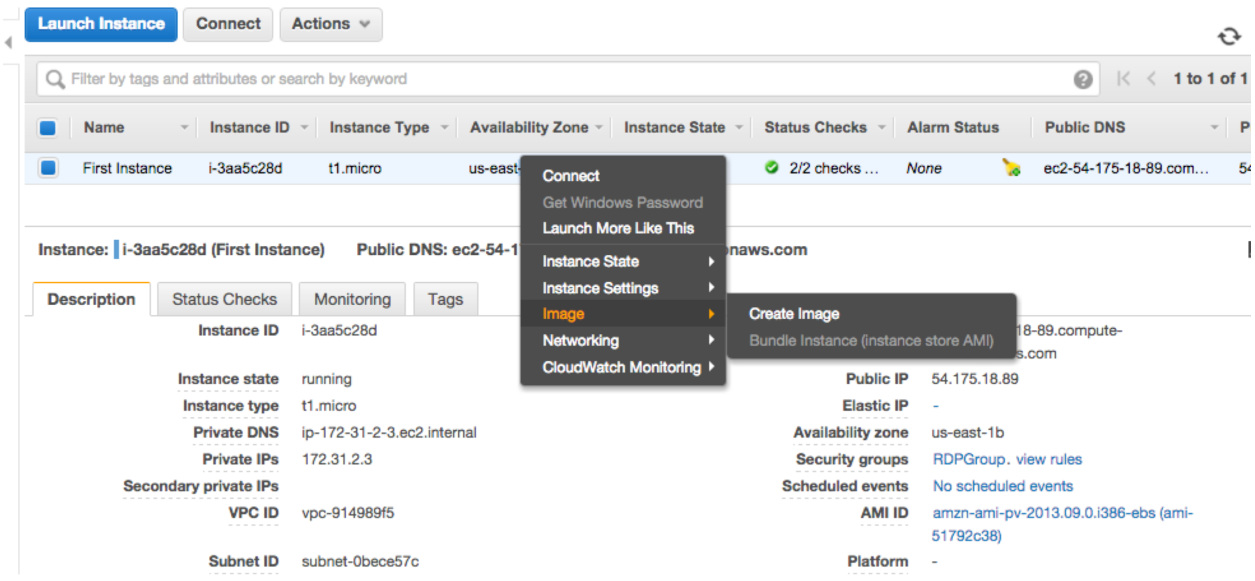
Hello! My IP address is: 172.31.2.3

Configure the server so that apache starts at boot time



```
  GNU nano 2.3.1                    File: rc.local

#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local
/etc/init.d/httpd start
```

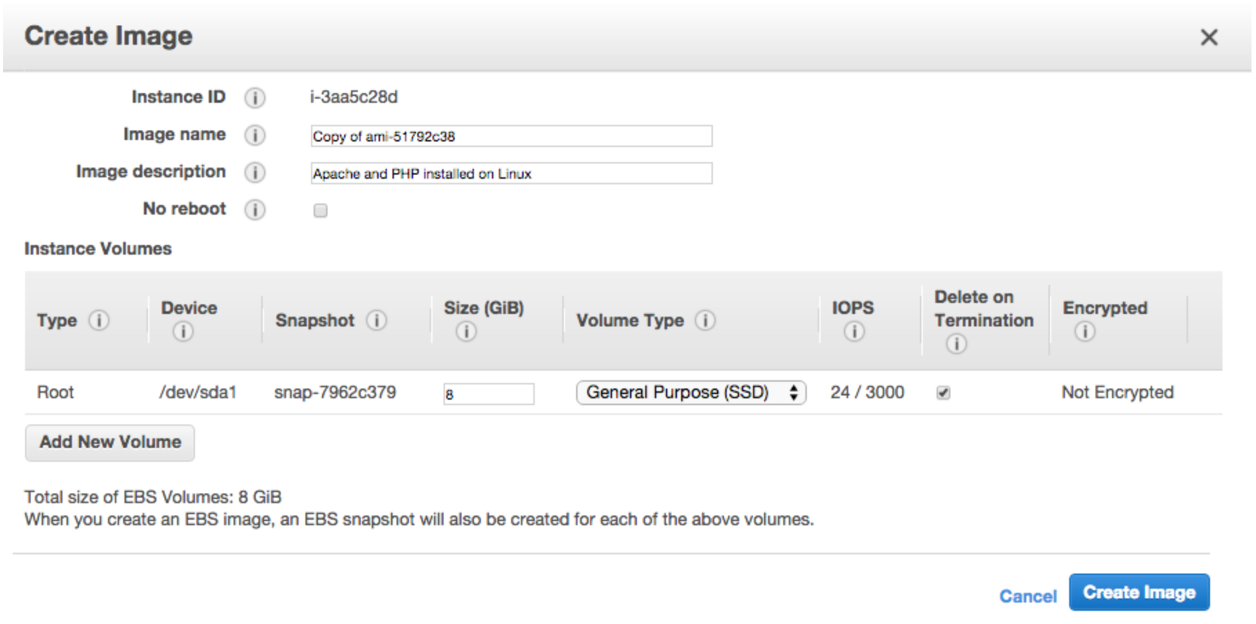
Create an AMI based on this configured instance



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
First Instance	i-3aa5c28d	t1.micro	us-east		2/2 checks ...	None	ec2-54-175-18-89.com...

Instance: i-3aa5c28d (First Instance) Public DNS: ec2-54-1 naws.com

Instance ID	i-3aa5c28d		18-89.compute- s.com
Instance state	running	Public IP	54.175.18.89
Instance type	t1.micro	Elastic IP	-
Private DNS	ip-172-31-2-3.ec2.internal	Availability zone	us-east-1b
Private IPs	172.31.2.3	Security groups	RDPGroup. view rules
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-914989f5	AMI ID	amzn-ami-pv-2013.09.0.i386-ebs (ami-51792c38)
Subnet ID	subnet-0bece57c	Platform	-



Create Image

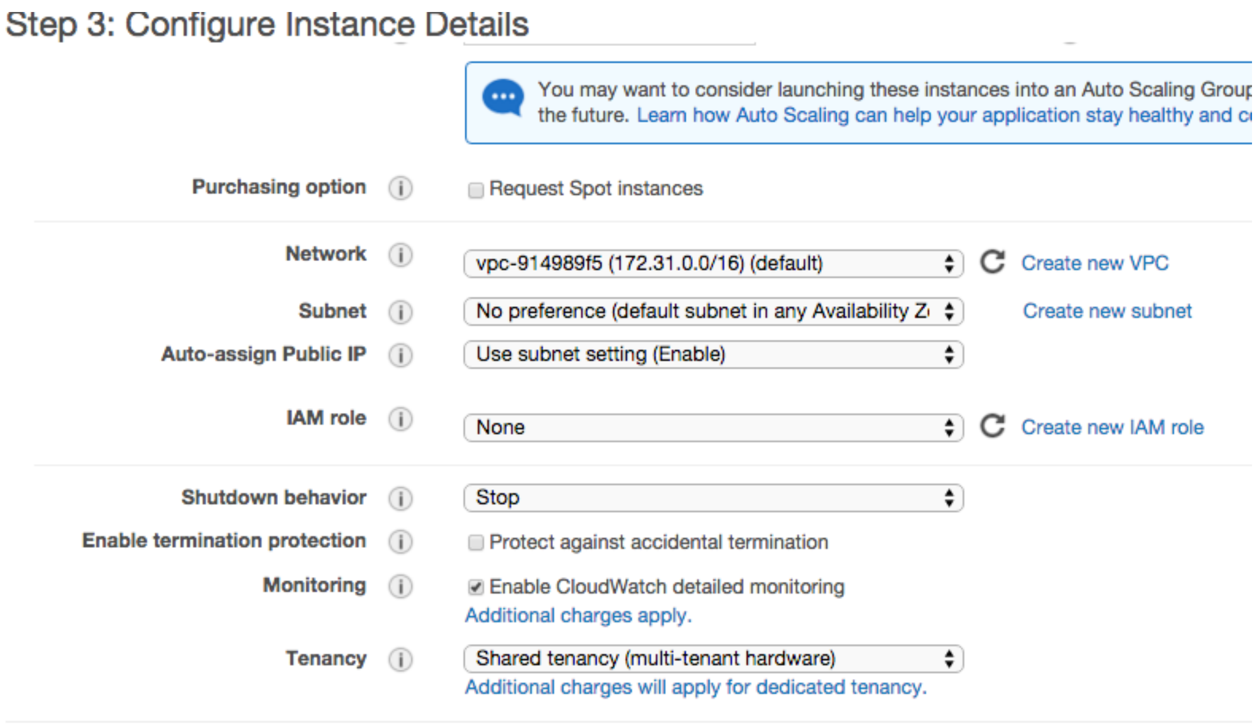
Instance ID	i-3aa5c28d
Image name	Copy of ami-51792c38
Image description	Apache and PHP installed on Linux
No reboot	☐

Instance Volumes

Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/sda1	snap-7962c379	8	General Purpose (SSD)	24 / 3000	✓	Not Encrypted

Total size of EBS Volumes: 8 GiB
When you create an EBS image, an EBS snapshot will also be created for each of the above volumes.

Create 3 instances based on this new AMI with CloudWatch Monitoring enabled

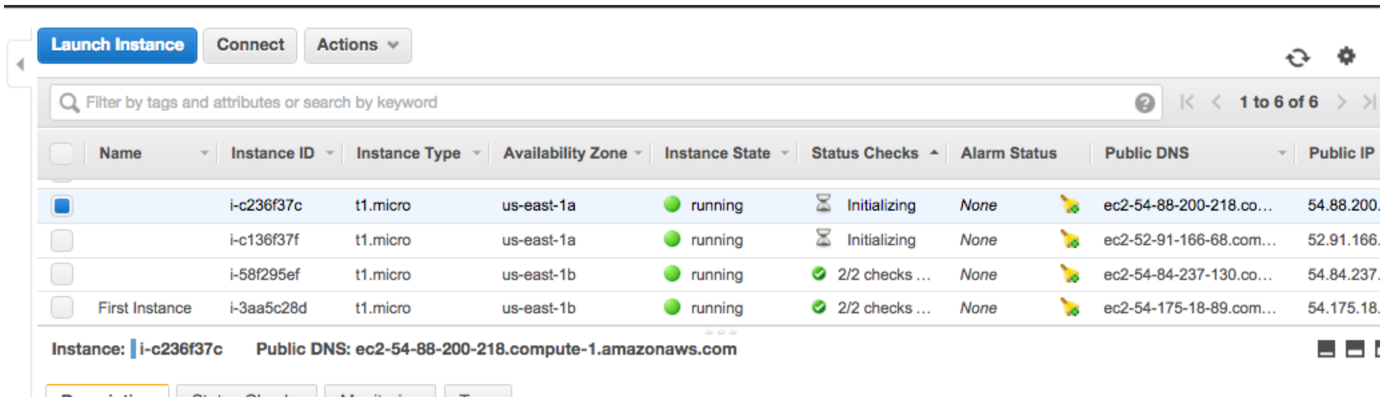


Step 3: Configure Instance Details

You may want to consider launching these instances into an Auto Scaling Group the future. Learn how Auto Scaling can help your application stay healthy and c

Purchasing option	Request Spot instances
Network	vpc-914989f5 (172.31.0.0/16) (default)
Subnet	No preference (default subnet in any Availability Z
Auto-assign Public IP	Use subnet setting (Enable)
IAM role	None
Shutdown behavior	Stop
Enable termination protection	Protect against accidental termination
Monitoring	☑ Enable CloudWatch detailed monitoring. Additional charges apply.
Tenancy	Shared tenancy (multi-tenant hardware). Additional charges will apply for dedicated tenancy.

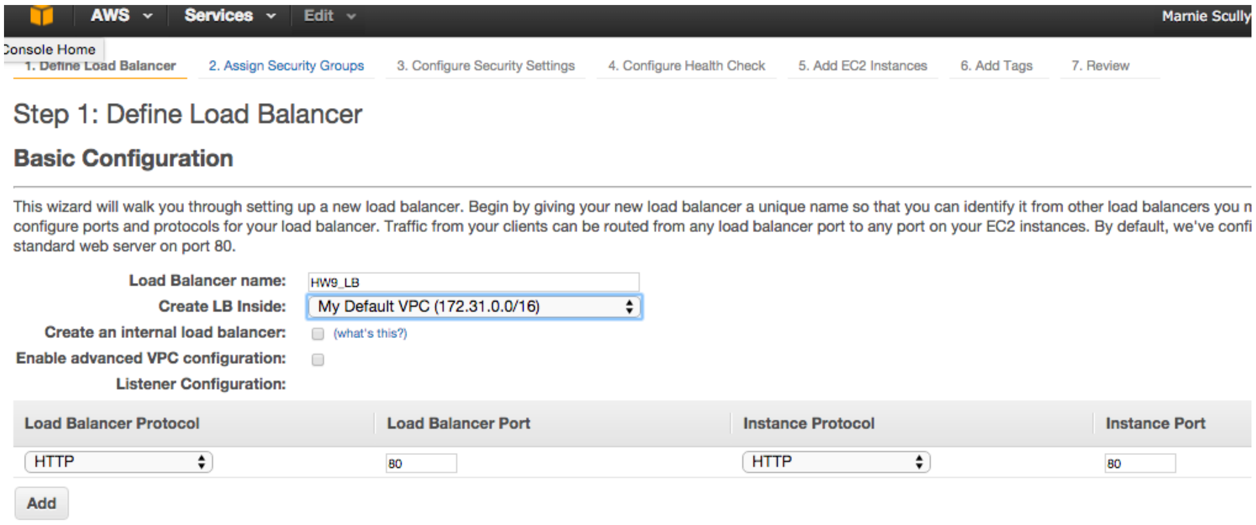
Place one instance in the same availability zone as the original instance and place the other two instances in another availability zone within the same region



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
	i-c236f37c	t1.micro	us-east-1a	running	Initializing	None	ec2-54-88-200-218.co...	54.88.200.
	i-c136f37f	t1.micro	us-east-1a	running	Initializing	None	ec2-52-91-166-68.com...	52.91.166.
	i-58f295ef	t1.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-84-237-130.co...	54.84.237.
First Instance	i-3aa5c28d	t1.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-175-18-89.com...	54.175.18.

Instance: i-c236f37c Public DNS: ec2-54-88-200-218.compute-1.amazonaws.com

Create a load balancer



Step 1: Define Load Balancer

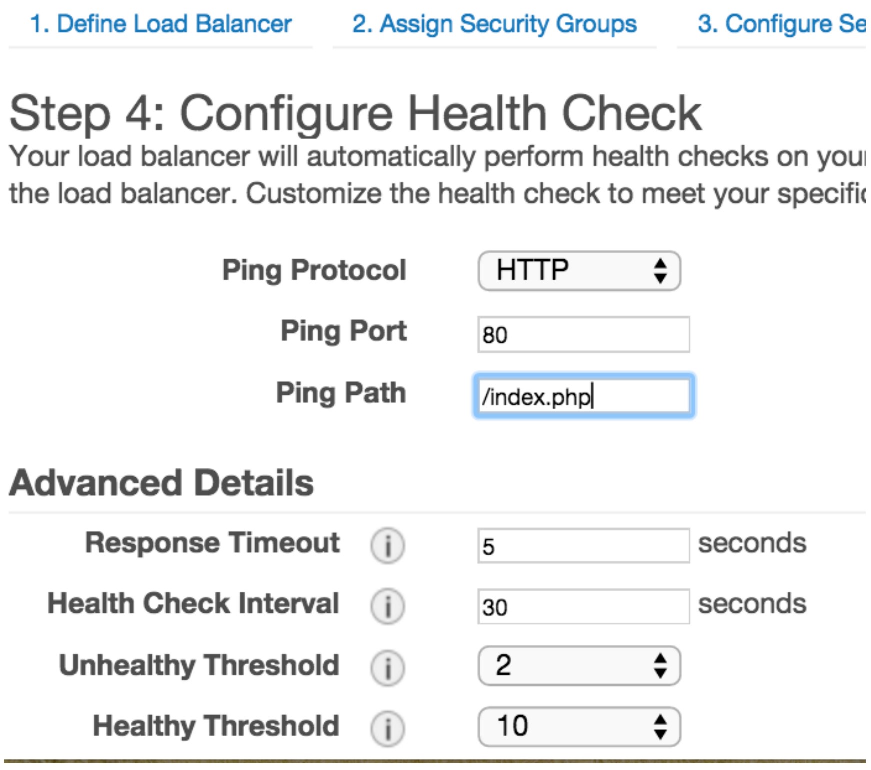
Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name so that you can identify it from other load balancers you n
configure ports and protocols for your load balancer. Traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've confi
standard web server on port 80.

Load Balancer name:	HW9_LB
Create LB Inside:	My Default VPC (172.31.0.0/16)
Create an internal load balancer:	☐ (what's this?)
Enable advanced VPC configuration:	☐
Listener Configuration:	

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port
HTTP	80	HTTP	80

Load Balancer health check URL /index.php



Step 4: Configure Health Check

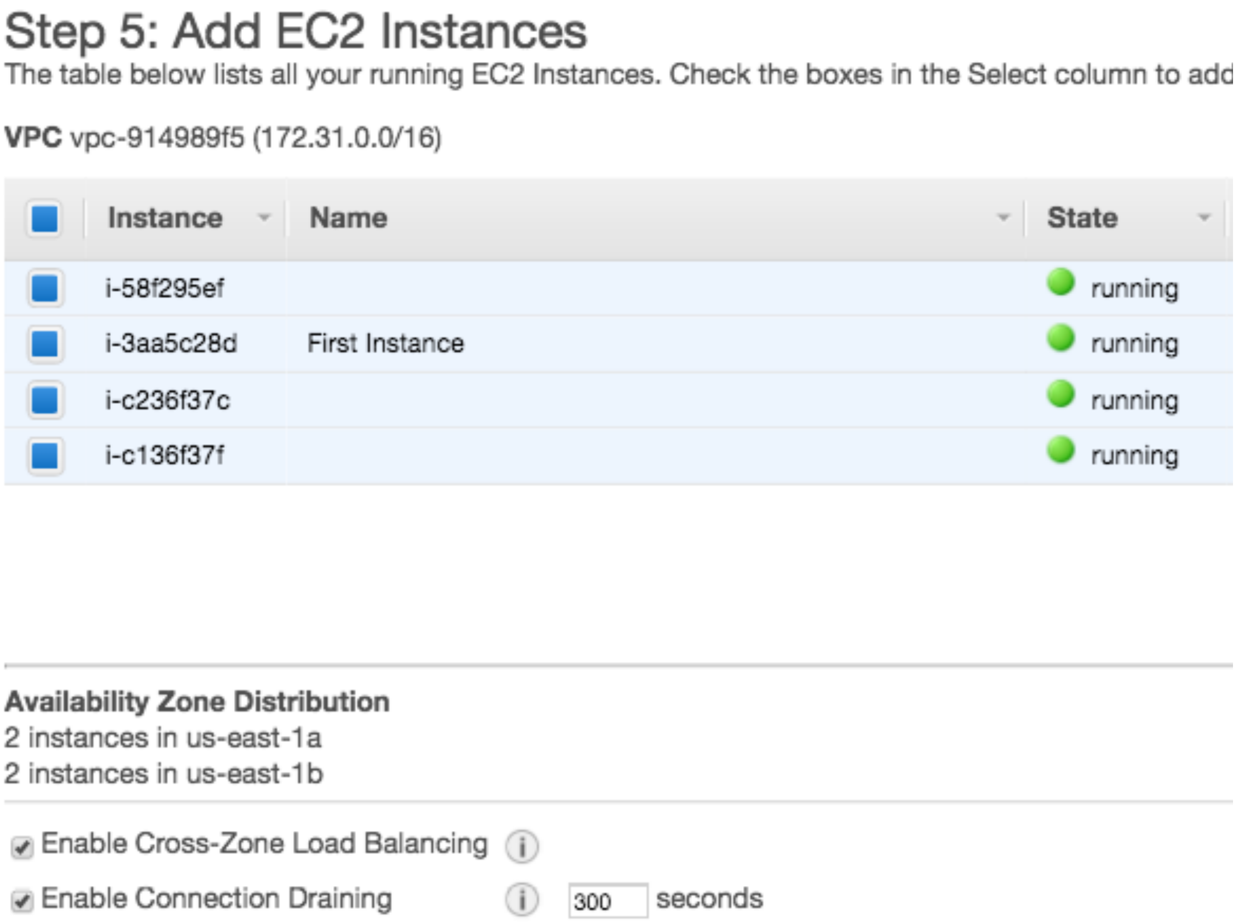
Your load balancer will automatically perform health checks on you
the load balancer. Customize the health check to meet your specifi

Ping Protocol	HTTP
Ping Port	80
Ping Path	/index.php

Advanced Details

Response Timeout	5 seconds
Health Check Interval	30 seconds
Unhealthy Threshold	2
Healthy Threshold	10

Associate all four instances with that load balancer



Step 5: Add EC2 Instances

The table below lists all your running EC2 Instances. Check the boxes in the Select column to add

VPC vpc-914989f5 (172.31.0.0/16)

	Instance	Name	State
✓	i-58f295ef		● running
✓	i-3aa5c28d	First Instance	● running
✓	i-c236f37c		● running
✓	i-c136f37f		● running

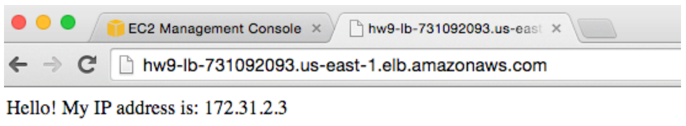
Availability Zone Distribution
2 instances in us-east-1a
2 instances in us-east-1b

[✓] Enable Cross-Zone Load Balancing

[✓] Enable Connection Draining 300 seconds

Test the load balancer URL in the browser about a dozen times

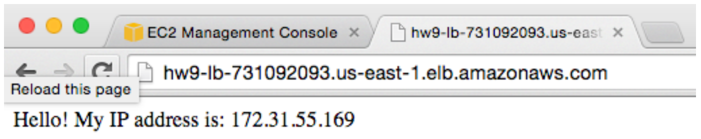
1st Server IP



hw9-lb-731092093.us-east-1.elb.amazonaws.com

Hello! My IP address is: 172.31.2.3

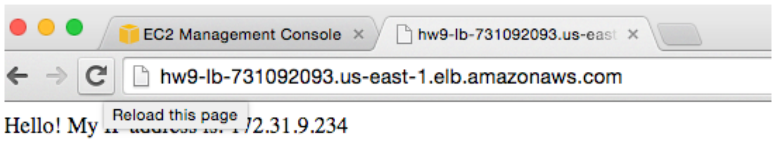
2nd Server IP



hw9-lb-731092093.us-east-1.elb.amazonaws.com

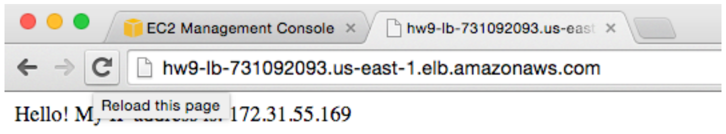
Hello! My IP address is: 172.31.55.169

3rd Server IP



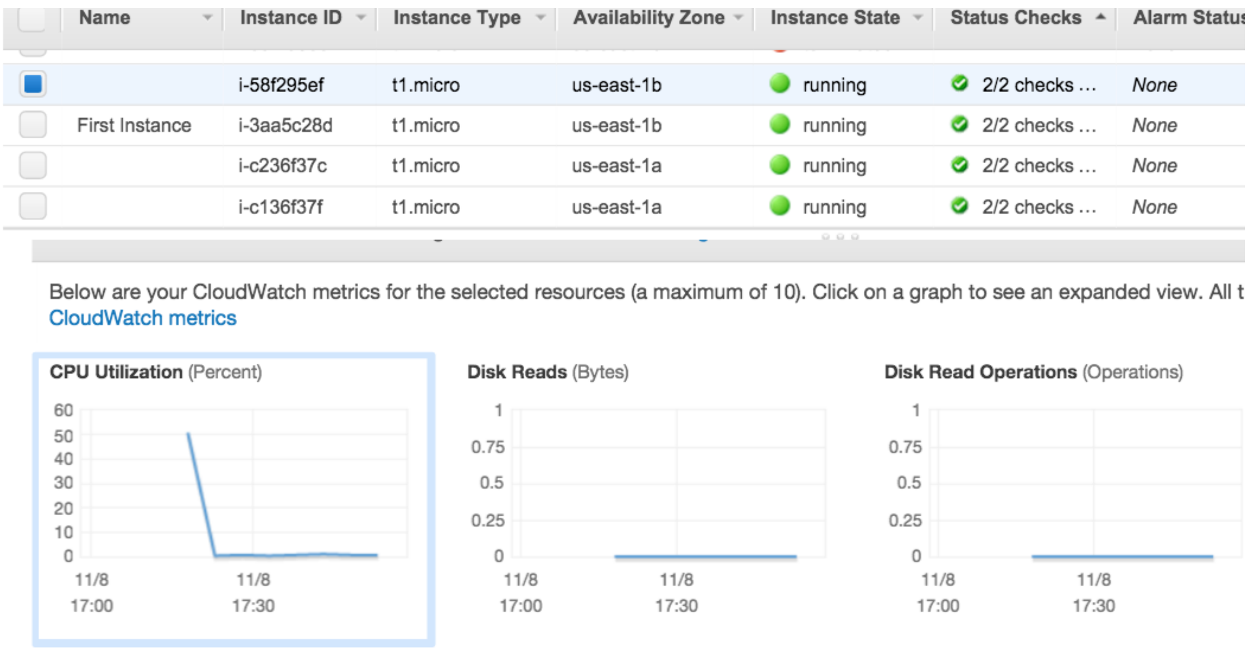
Hello! My IP address is: 172.31.9.234

4th Server IP



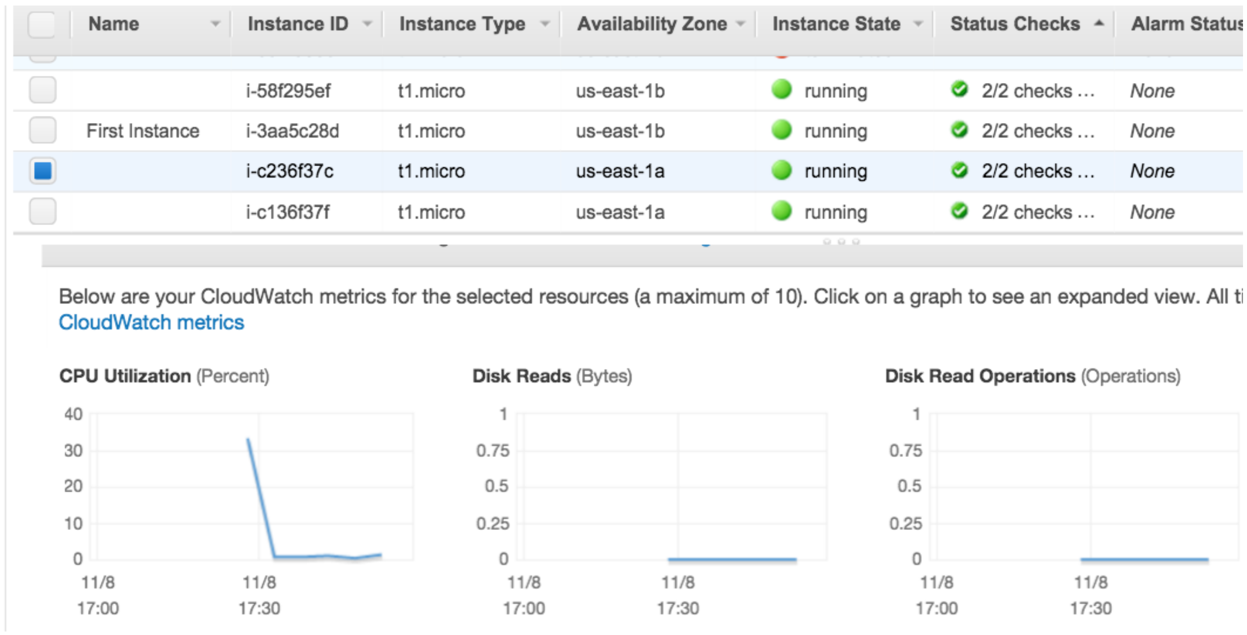
Hello! My IP address is: 172.31.55.169

View Cloud Watch Monitor to view the distribution of requests between instances



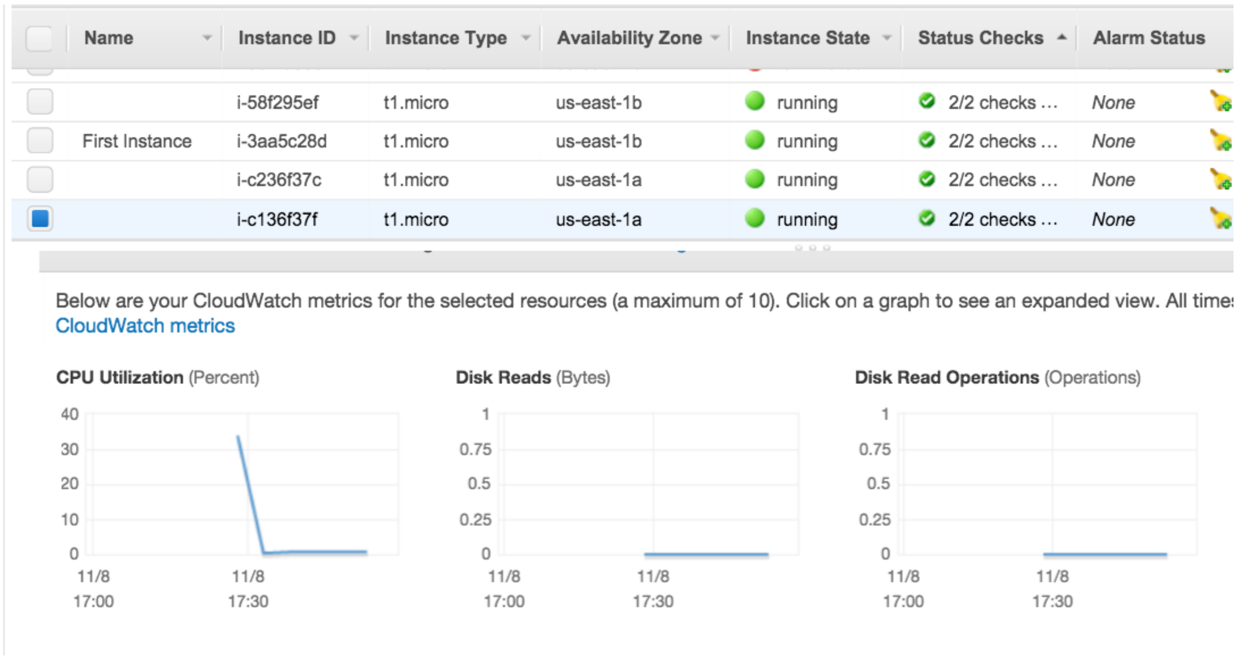
	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
■		i-58f295ef	t1.micro	us-east-1b	● running	✓ 2/2 checks ...	None
	First Instance	i-3aa5c28d	t1.micro	us-east-1b	● running	✓ 2/2 checks ...	None
		i-c236f37c	t1.micro	us-east-1a	● running	✓ 2/2 checks ...	None
		i-c136f37f	t1.micro	us-east-1a	● running	✓ 2/2 checks ...	None

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All t
CloudWatch metrics



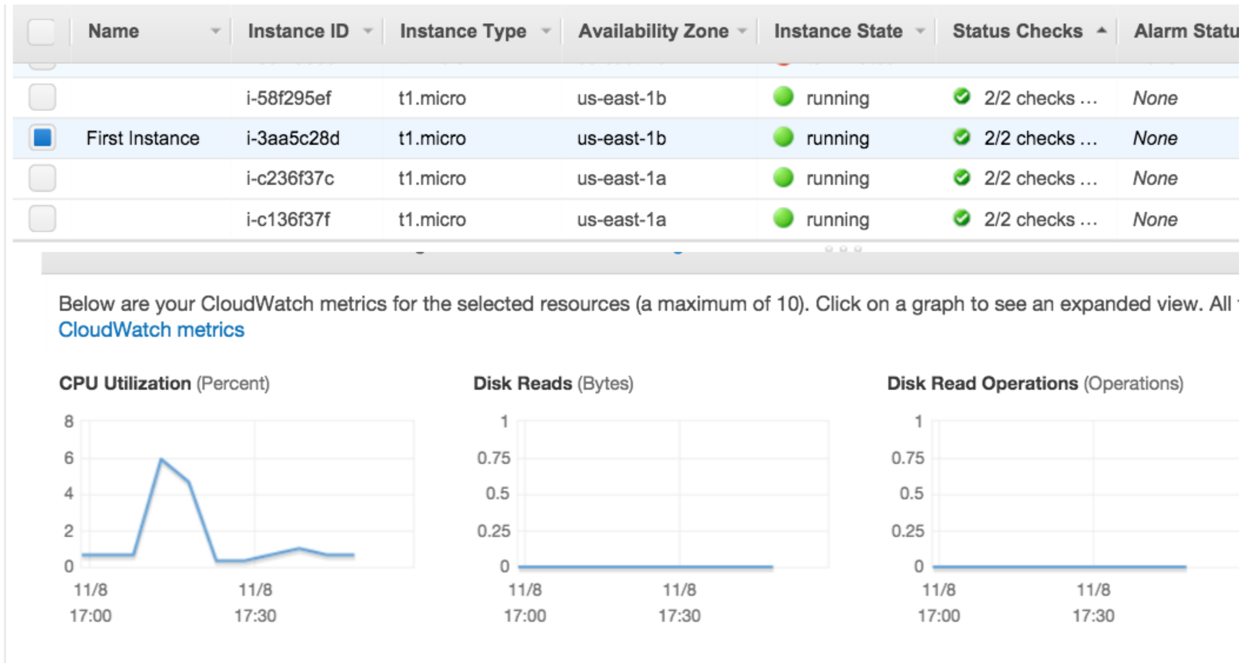
	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
		i-58f295ef	t1.micro	us-east-1b	running	2/2 checks ...	None
	First Instance	i-3aa5c28d	t1.micro	us-east-1b	running	2/2 checks ...	None
✓		i-c236f37c	t1.micro	us-east-1a	running	2/2 checks ...	None
		i-c136f37f	t1.micro	us-east-1a	running	2/2 checks ...	None

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All ti
CloudWatch metrics



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
		i-58f295ef	t1.micro	us-east-1b	running	2/2 checks ...	None
	First Instance	i-3aa5c28d	t1.micro	us-east-1b	running	2/2 checks ...	None
		i-c236f37c	t1.micro	us-east-1a	running	2/2 checks ...	None
✓		i-c136f37f	t1.micro	us-east-1a	running	2/2 checks ...	None

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All times
CloudWatch metrics



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Statu
		i-58f295ef	t1.micro	us-east-1b	● running	✓ 2/2 checks ...	None
■	First Instance	i-3aa5c28d	t1.micro	us-east-1b	● running	✓ 2/2 checks ...	None
		i-c236f37c	t1.micro	us-east-1a	● running	✓ 2/2 checks ...	None
		i-c136f37f	t1.micro	us-east-1a	● running	✓ 2/2 checks ...	None

Below are your CloudWatch metrics for the selected resources (a maximum of 10). Click on a graph to see an expanded view. All
CloudWatch metrics

Problem 2:
Use techniques we relied in the RESTful Web Service client class CustomerResourceClient.java to build an automated testing tool for verifying performance of the Load Balancer. That testing tool should send Http GET requests to the Load Balancer and keep track of the responses. Hardcode the number of pings you are sending (100-1000). In the first iteration print responses coming from the Load Balancer (your application producing IP addresses). In the second iteration count how many times you received each IP address. Work with 3 instances behind the load balancer.
Points: [30]

Edit Customer ResourceClient.java to connect to Load Balancer

```
Map<String, Integer> hashMap = new HashMap<String, Integer>();
int pings = 1000;
URL url = new URL("http://HW9-LB-731092093.us-
east-1.elb.amazonaws.com");
String ip = null;
Integer count = null;
HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
```

Send GET requests to Load Balancer and track the responses (The Private IP address of the server)
Iterate through responses and display IP addresses from responses

```
for (int i = 0; i < pings; i++) {
     connection = (HttpURLConnection) url.openConnection();
```

```
	connection.setRequestMethod("GET");
	BufferedReader reader = new BufferedReader(
	new InputStreamReader(connection.getInputStream()));
	ip = "";
	for (String line = reader.readLine(); line != null; line =
reader.readLine())
	ip += line;
	count = hashMap.get(ip);
	hashMap.put(ip, ((count == null) ? 0 : count.intValue()) +
1);
	System.out.println (ip);
}
```

Iterate through responses and count how many times IP address was received and display counts

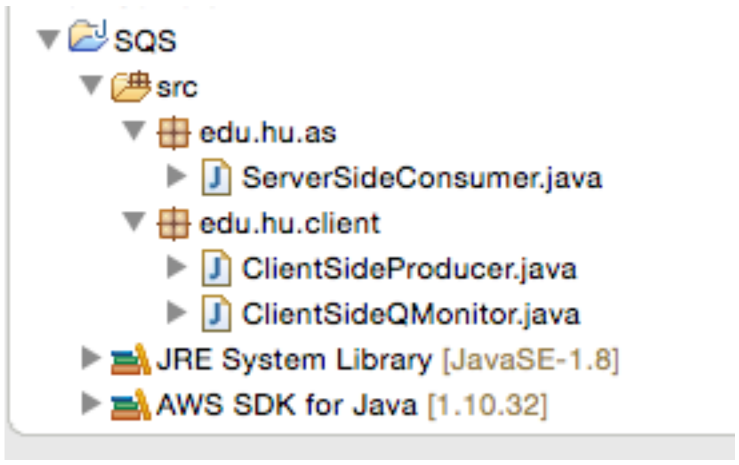
```
for (Map.Entry<String, Integer> entry : hashMap.entrySet()) {
	System.out.println(entry.getKey() + ": " +
entry.getValue());
}
```

Output in Console

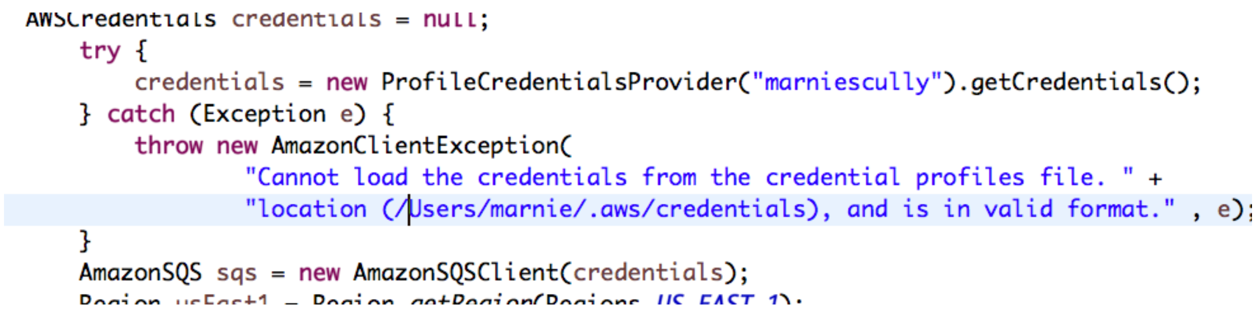
```
Hello! My IP address is: 172.31.55.168
Hello! My IP address is: 172.31.9.234
Hello! My IP address is: 172.31.55.169
Hello! My IP address is: 172.31.55.168
Hello! My IP address is: 172.31.9.234
Hello! My IP address is: 172.31.55.169
Hello! My IP address is: 172.31.55.168
Hello! My IP address is: 172.31.9.234
Hello! My IP address is: 172.31.55.169
Hello! My IP address is: 172.31.55.168
Hello! My IP address is: 172.31.9.234
Hello! My IP address is: 172.31.55.169
Hello! My IP address is: 172.31.55.168
Hello! My IP address is: 172.31.9.234
Hello! My IP address is: 172.31.55.169

Hello! My IP address is: 172.31.55.169: 334
Hello! My IP address is: 172.31.9.234: 333
Hello! My IP address is: 172.31.55.168: 333
```

Problem 3:
In the attached: hwk9_`edu.zip` archive you will find classes `ServerSideConsumer.java`, `ClientSideProducer.java` and `ClientSideQMonitor.java` used in class. Create an AWS Java project based on `SQSSample` application. Remove the class that comes with that project. Copy provided classes into the new project. Make sure that you configure your project with your AWS credentials. Make sure that you change compliance level of your Eclipse project to Java 1.5.
Create an SQS Queue and enter its URL in the appropriate places in those classes. Test your classes in Eclipse and make sure they can communicate with your SQS Queue. You will next create three executable jars, each with one of the above classes as the Main class. Follow the procedure used in class. Test those jars again on the operating system prompt. You noticed that my classes have hard coded wait times which determine the frequency with which producer sends messages to the queue and the frequency with which the consumer retrieves messages from the queue.
If you know how, modify those classes so that you can pass those wait times to the main classes at the run time as command line parameters. If you do not know how to do that, do not bother. Secure copy (`scp`) the executable jar with `ServerSideConsumer` class to the running instance in the Cloud. Start client side producer and make sure that the server side consumer truly consumes messages from the queue. Open AWS Console for SQS service and verify that messages are placed in queue and subsequently consumed. Modify your Cloud instance so that that the server side jar is started on every reboot or startup of that instance. Test your arrangement by stopping or rebooting the instance. Once the instance is restarted, right clink on the instance in EC2 Console (My Instances page) and select `Get System Log`. If your jar works, at the bottom of the log, you will see its output confirming that messages are indeed read.
Points: [35]

Create an AWS Java project based on `SQSSample` application.
Remove the class that comes with that project. Copy provided classes into the new project.

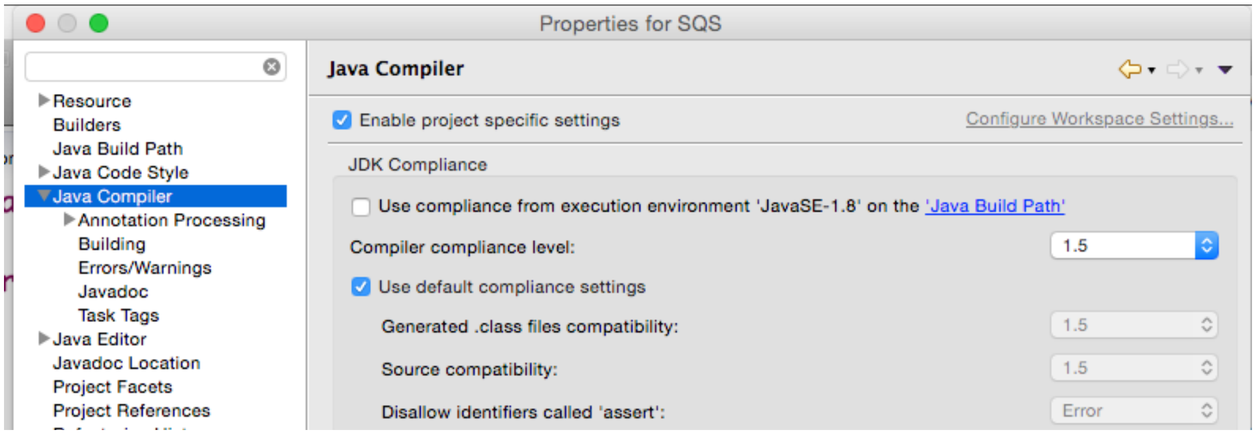

```
▼ SQS
  ▼ src
    ▼ edu.hu.as
      ▶ ServerSideConsumer.java
    ▼ edu.hu.client
      ▶ ClientSideProducer.java
      ▶ ClientSideQMonitor.java
  ▶ JRE System Library [JavaSE-1.8]
  ▶ AWS SDK for Java [1.10.32]
```

Configure your project with your AWS credentials



```
AWSCredentials credentials = null;
    try {
        credentials = new ProfileCredentialsProvider("marniescully").getCredentials();
    } catch (Exception e) {
        throw new AmazonClientException(
                "Cannot load the credentials from the credential profiles file. " +
                "location (/Users/marnie/.aws/credentials), and is in valid format." , e);
    }
    AmazonSQS sqs = new AmazonSQSClient(credentials);
    Region usEast1 = Region.getRegion(Regions.US_EAST_1);
```

Change compliance level of your Eclipse project to Java 1.5



Java Compiler

☑ Enable project specific settings

JDK Compliance

☐ Use compliance from execution environment 'JavaSE-1.8' on the 'Java Build Path'

Compiler compliance level: 1.5

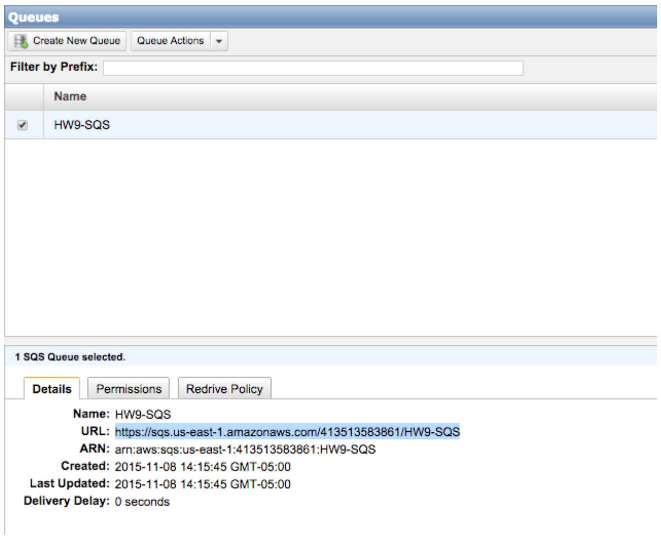
☑ Use default compliance settings

Generated .class files compatibility: 1.5

Source compatibility: 1.5

Disallow identifiers called 'assert': Error

Create an SQS Queue



	Name
✓	HW9-SQS

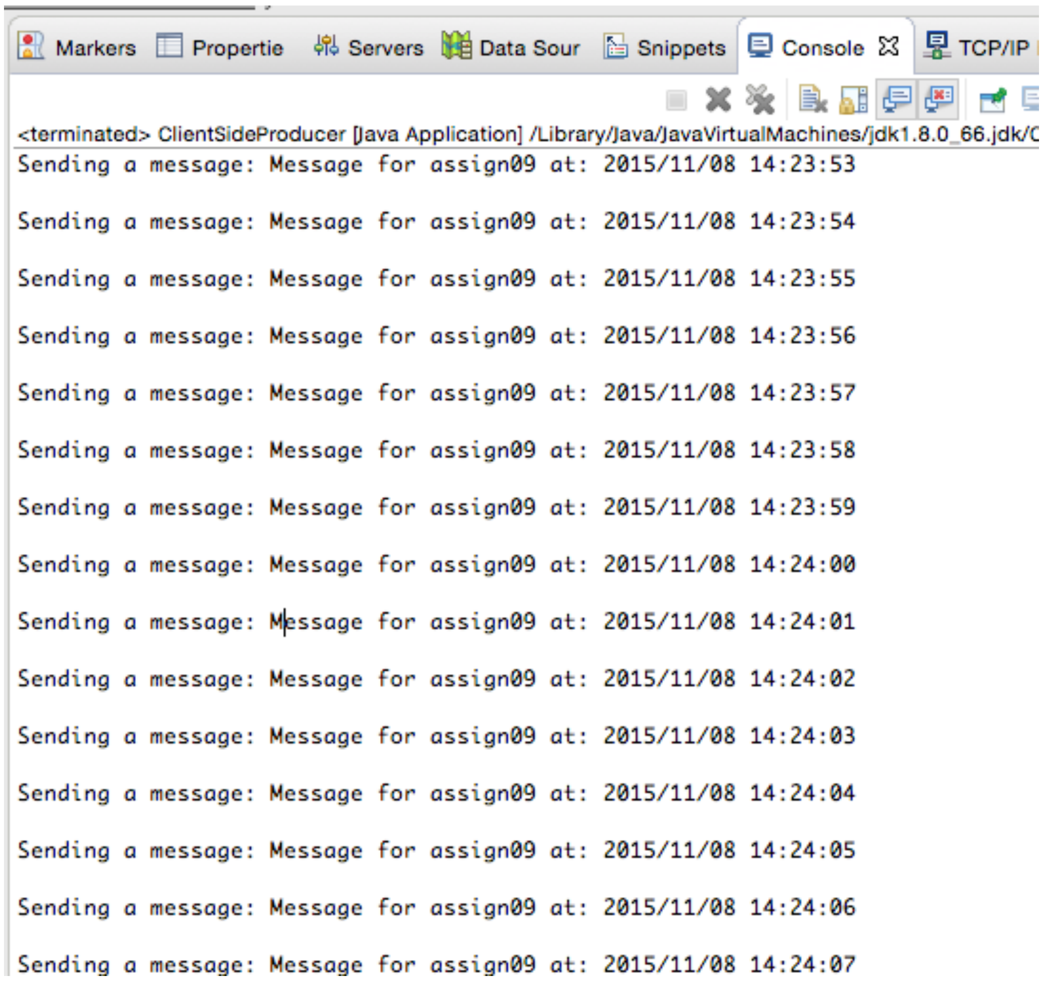
1 SQS Queue selected.

Name: HW9-SQS
URL: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
ARN: arn:aws:sqs:us-east-1:413513583861:HW9-SQS
Created: 2015-11-08 14:15:45 GMT-05:00
Last Updated: 2015-11-08 14:15:45 GMT-05:00
Delivery Delay: 0 seconds

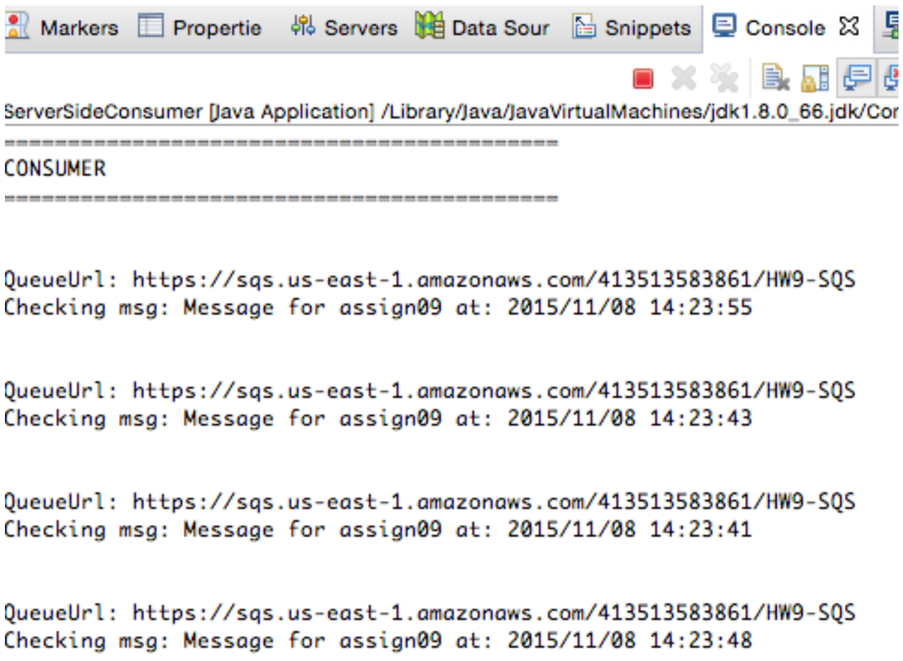
and enter its URL in the appropriate places in those classes.

```
/* replace with your sqs queue URL */
     sqs.sendMessage( new SendMessageRequest( "https://sqs.us-
east-1.amazonaws.com/413513583861/HW9-SQS", msg ) );
```

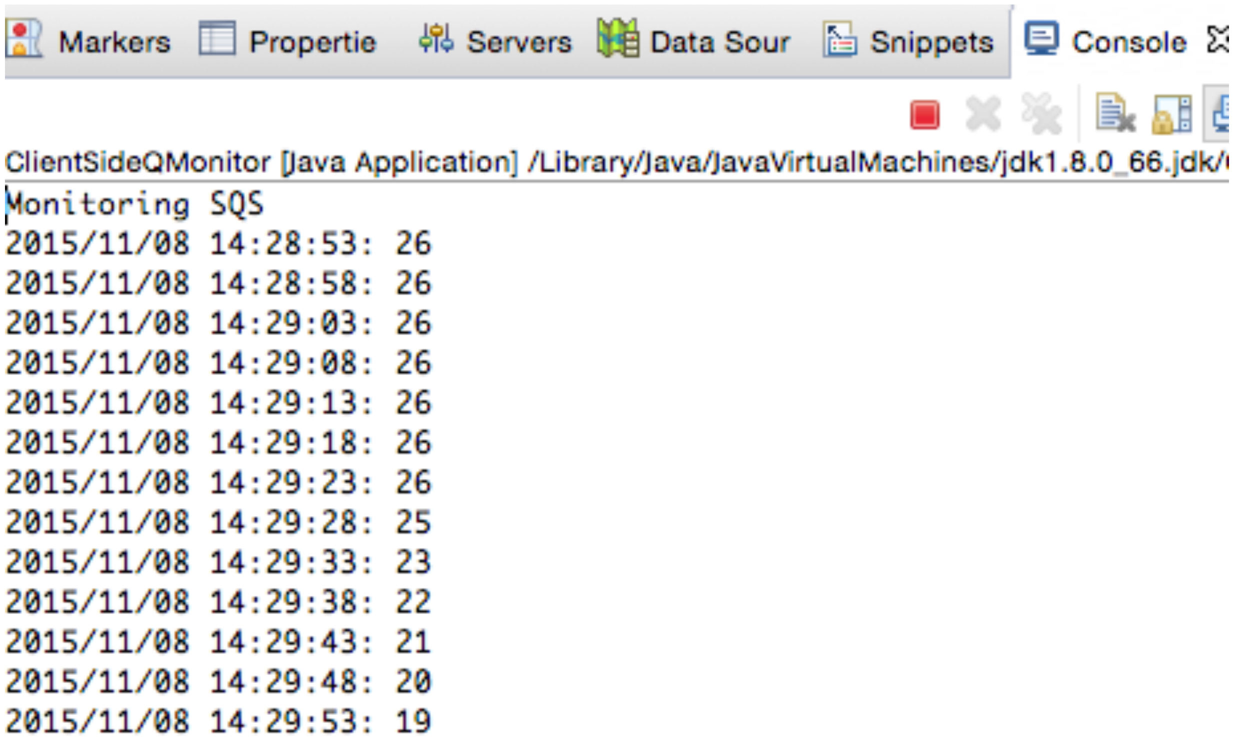
Test your classes in Eclipse and make sure they can communicate with your SQS Queue.



```
<terminated> ClientSideProducer [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_66.jdk/C
Sending a message: Message for assign09 at: 2015/11/08 14:23:53

Sending a message: Message for assign09 at: 2015/11/08 14:23:54

Sending a message: Message for assign09 at: 2015/11/08 14:23:55

Sending a message: Message for assign09 at: 2015/11/08 14:23:56

Sending a message: Message for assign09 at: 2015/11/08 14:23:57

Sending a message: Message for assign09 at: 2015/11/08 14:23:58

Sending a message: Message for assign09 at: 2015/11/08 14:23:59

Sending a message: Message for assign09 at: 2015/11/08 14:24:00

Sending a message: Message for assign09 at: 2015/11/08 14:24:01

Sending a message: Message for assign09 at: 2015/11/08 14:24:02

Sending a message: Message for assign09 at: 2015/11/08 14:24:03

Sending a message: Message for assign09 at: 2015/11/08 14:24:04

Sending a message: Message for assign09 at: 2015/11/08 14:24:05

Sending a message: Message for assign09 at: 2015/11/08 14:24:06

Sending a message: Message for assign09 at: 2015/11/08 14:24:07
```

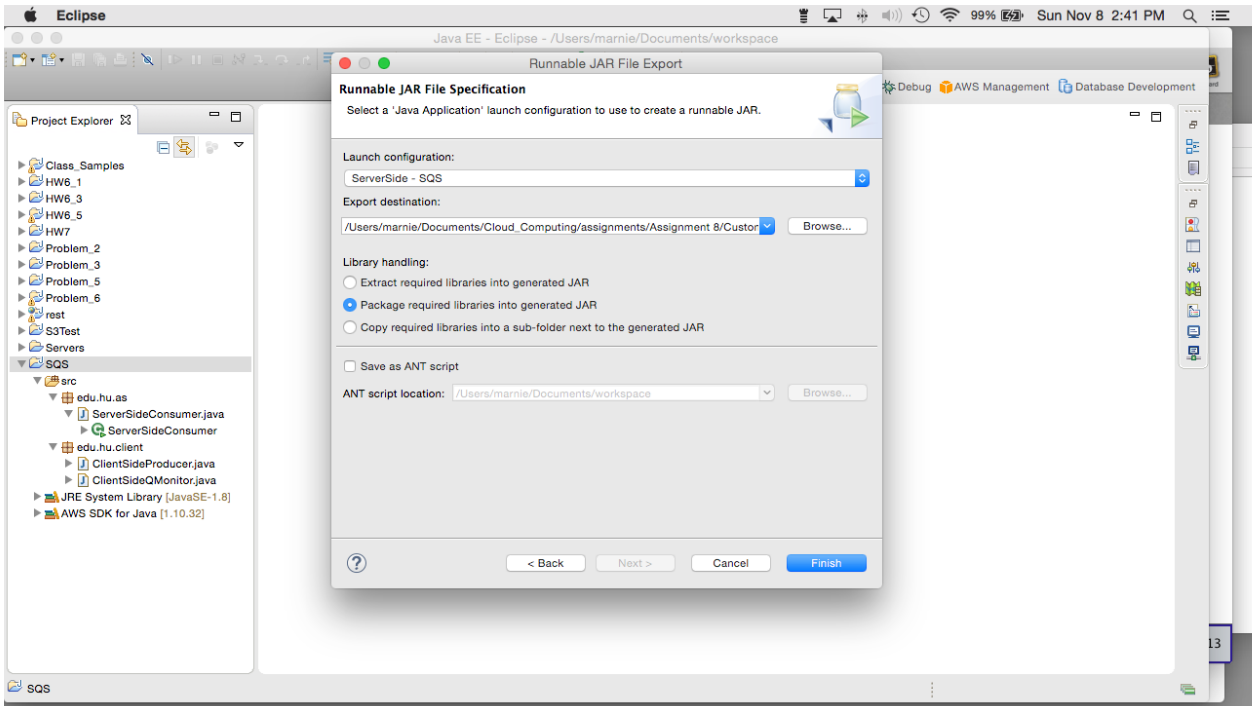


```
ServerSideConsumer [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_66.jdk/Cor
============================================
CONSUMER
============================================


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:55


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:43


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:41


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:48
```



```
ClientSideQMonitor [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_66.jdk/
Monitoring SQS
2015/11/08 14:28:53: 26
2015/11/08 14:28:58: 26
2015/11/08 14:29:03: 26
2015/11/08 14:29:08: 26
2015/11/08 14:29:13: 26
2015/11/08 14:29:18: 26
2015/11/08 14:29:23: 26
2015/11/08 14:29:28: 25
2015/11/08 14:29:33: 23
2015/11/08 14:29:38: 22
2015/11/08 14:29:43: 21
2015/11/08 14:29:48: 20
2015/11/08 14:29:53: 19
```

Create an executable jars for ServerSideConsumer, to run on the Cloud server



Runnable JAR File Export

Runnable JAR File Specification

Select a 'Java Application' launch configuration to use to create a runnable JAR.

Launch configuration: ServerSide - SQS

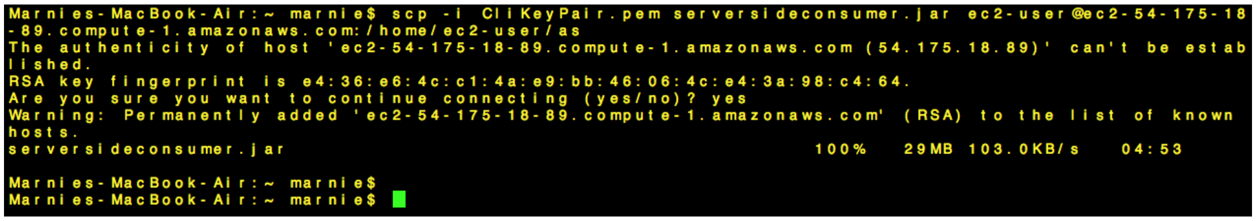
Export destination: /Users/marnie/Documents/Cloud_Computing/assignments/Assignment 8/Custor

Library handling:
- ○ Extract required libraries into generated JAR
- ● Package required libraries into generated JAR
- ○ Copy required libraries into a sub-folder next to the generated JAR

☐ Save as ANT script

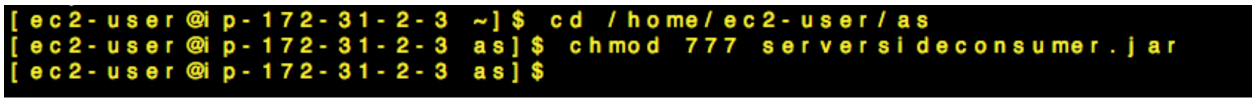
ANT script location: /Users/marnie/Documents/workspace

Secure copy (`scp`) the executable jar with `ServerSideConsumer` class to the running instance



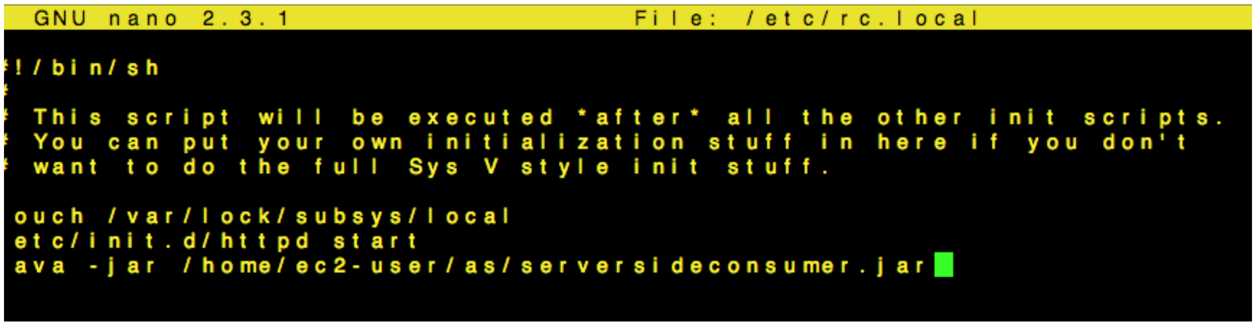
```
Marnies-MacBook-Air:~ marnie$ scp -i CliKeyPair.pem serversideconsumer.jar ec2-user@ec2-54-175-18
-89.compute-1.amazonaws.com:/home/ec2-user/as
The authenticity of host 'ec2-54-175-18-89.compute-1.amazonaws.com (54.175.18.89)' can't be estab
lished.
RSA key fingerprint is e4:36:e6:4c:c1:4a:e9:bb:46:06:4c:e4:3a:98:c4:64.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-175-18-89.compute-1.amazonaws.com' (RSA) to the list of known
hosts.
serversideconsumer.jar                                    100%   29MB 103.0KB/s   04:53

Marnies-MacBook-Air:~ marnie$
Marnies-MacBook-Air:~ marnie$
```

Make the jar executable on the server



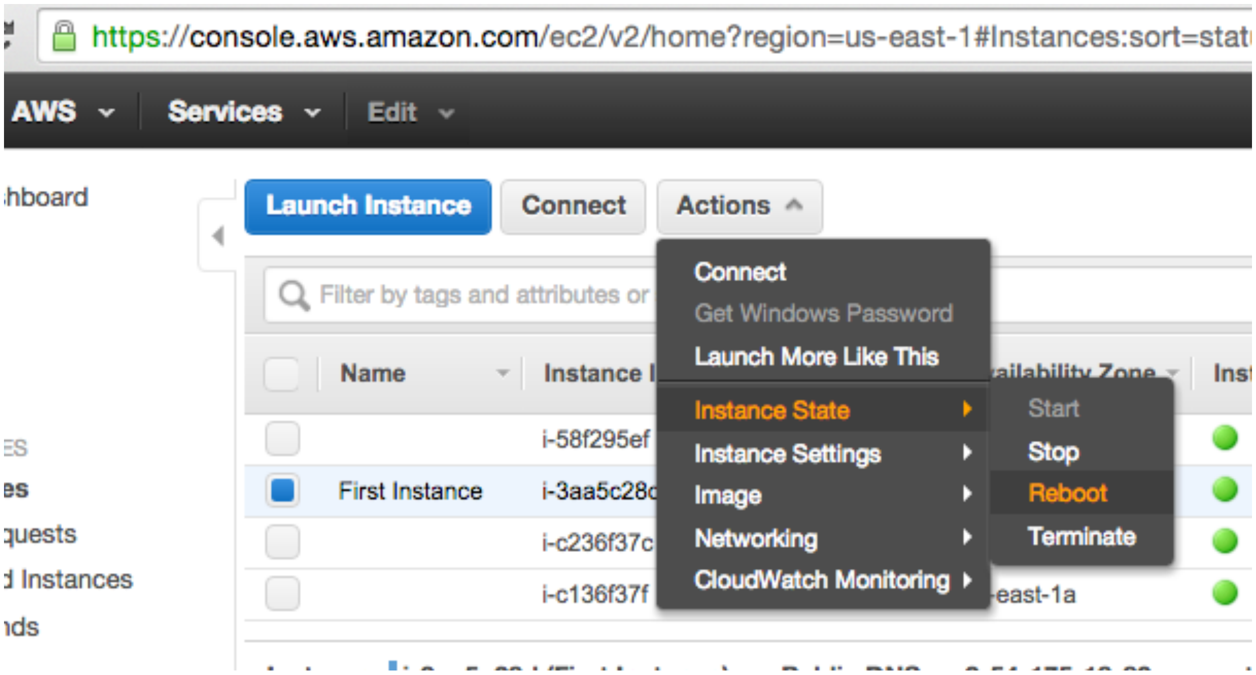
```
[ec2-user@ip-172-31-2-3 ~]$ cd /home/ec2-user/as
[ec2-user@ip-172-31-2-3 as]$ chmod 777 serversideconsumer.jar
[ec2-user@ip-172-31-2-3 as]$
```

Have the jar run on boot on server



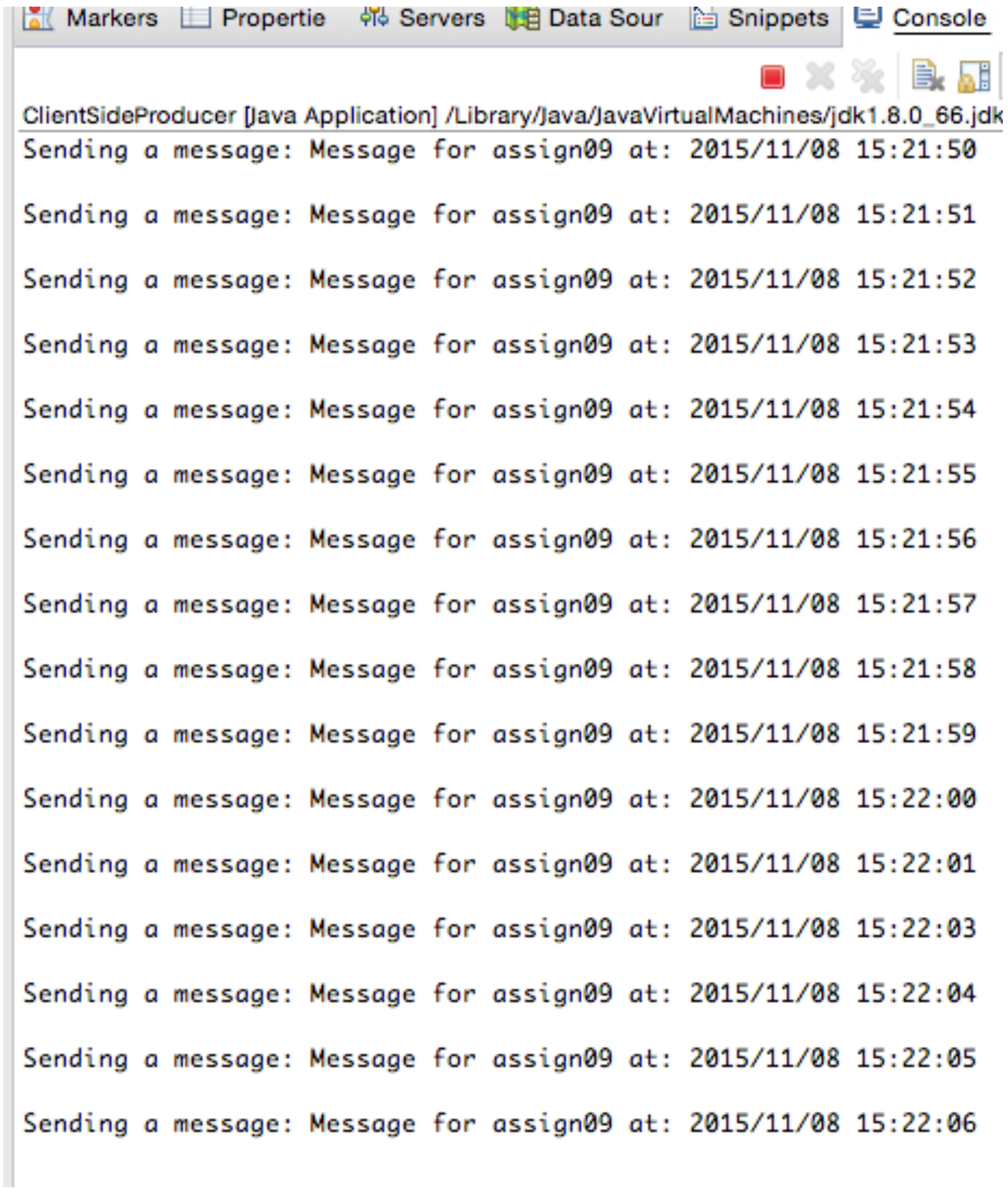
```
  GNU nano 2.3.1                    File: /etc/rc.local

!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here if you don't
# want to do the full Sys V style init stuff.

ouch /var/lock/subsys/local
etc/init.d/httpd start
ava -jar /home/ec2-user/as/serversideconsumer.jar
```

Reboot the instance from AWS Console



https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=stat

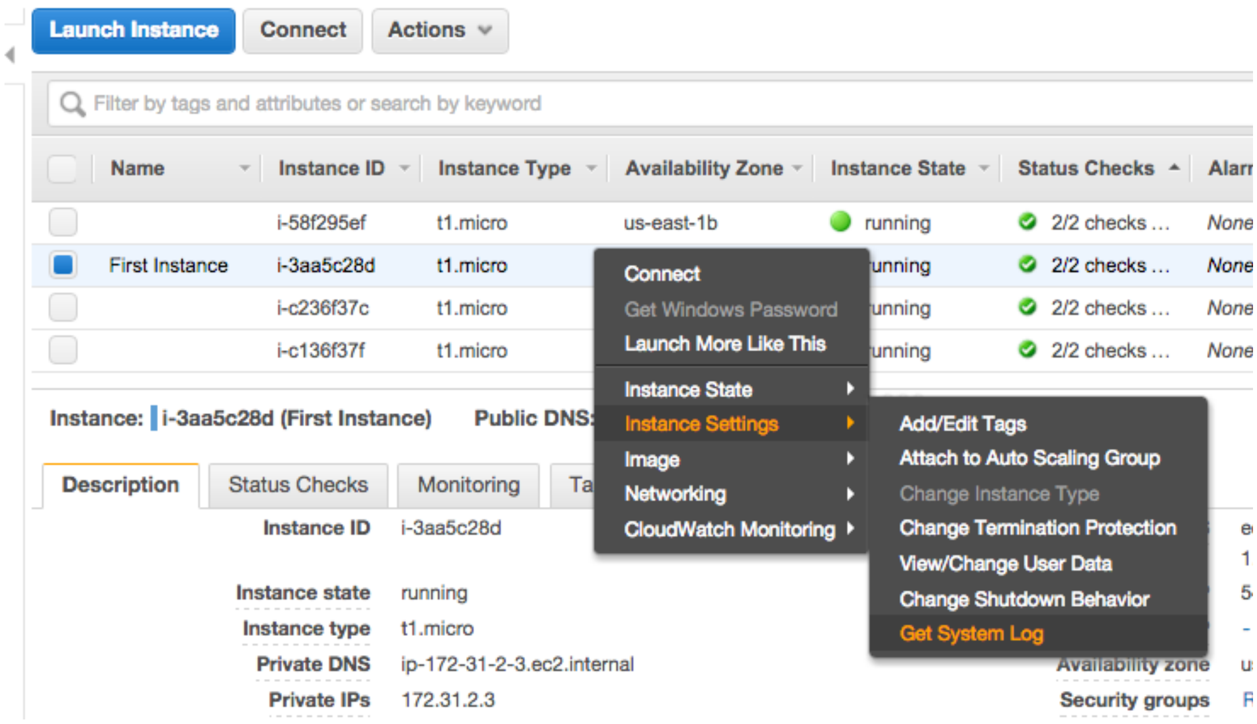
Start client side producer



```
ClientSideProducer [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_66.jdk
Sending a message: Message for assign09 at: 2015/11/08 15:21:50

Sending a message: Message for assign09 at: 2015/11/08 15:21:51

Sending a message: Message for assign09 at: 2015/11/08 15:21:52

Sending a message: Message for assign09 at: 2015/11/08 15:21:53

Sending a message: Message for assign09 at: 2015/11/08 15:21:54

Sending a message: Message for assign09 at: 2015/11/08 15:21:55

Sending a message: Message for assign09 at: 2015/11/08 15:21:56

Sending a message: Message for assign09 at: 2015/11/08 15:21:57

Sending a message: Message for assign09 at: 2015/11/08 15:21:58

Sending a message: Message for assign09 at: 2015/11/08 15:21:59

Sending a message: Message for assign09 at: 2015/11/08 15:22:00

Sending a message: Message for assign09 at: 2015/11/08 15:22:01

Sending a message: Message for assign09 at: 2015/11/08 15:22:03

Sending a message: Message for assign09 at: 2015/11/08 15:22:04

Sending a message: Message for assign09 at: 2015/11/08 15:22:05

Sending a message: Message for assign09 at: 2015/11/08 15:22:06
```

Open AWS Console for SQS service and verify that messages are placed in queue and subsequently consumed

Right-Click on the instance in EC2 Console (My Instances page) and select `Get System Log`.

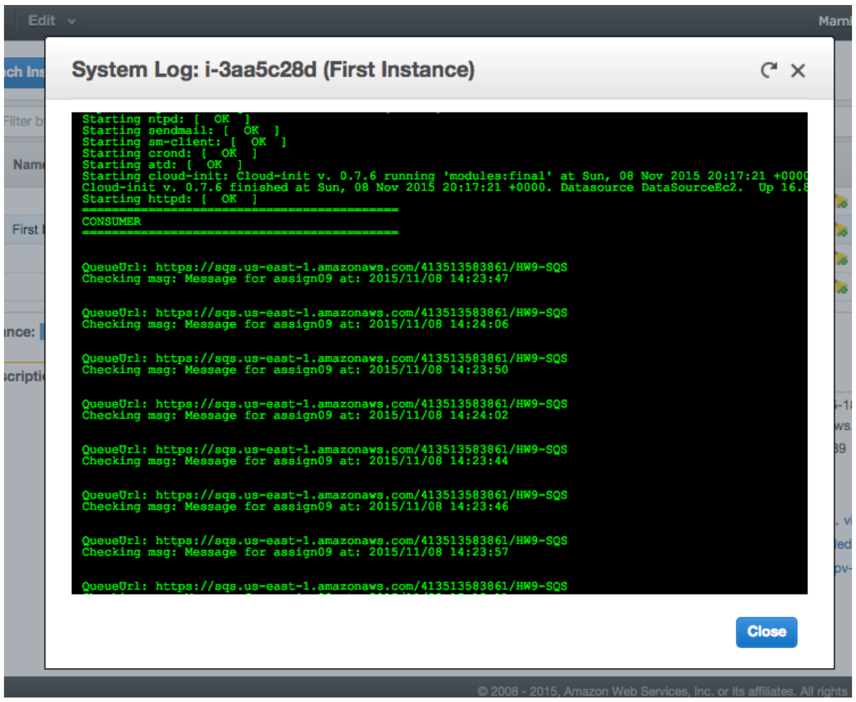


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm
	i-58f295ef	t1.micro	us-east-1b	running	2/2 checks ...	None
First Instance	i-3aa5c28d	t1.micro		unning	2/2 checks ...	None
	i-c236f37c	t1.micro		unning	2/2 checks ...	None
	i-c136f37f	t1.micro		unning	2/2 checks ...	None

Instance: i-3aa5c28d (First Instance) Public DNS:

Instance ID	i-3aa5c28d
Instance state	running
Instance type	t1.micro
Private DNS	ip-172-31-2-3.ec2.internal
Private IPs	172.31.2.3

View output confirming that messages are indeed read


Edit
Close
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights

System Log: i-3aa5c28d (First Instance)

```
Starting ntpd: [  OK  ]
Starting sendmail: [  OK  ]
Starting sm-client: [  OK  ]
Starting crond: [  OK  ]
Starting atd: [  OK  ]
Starting cloud-init: Cloud-init v. 0.7.6 running 'modules:final' at Sun, 08 Nov 2015 20:17:21 +0000
Cloud-init v. 0.7.6 finished at Sun, 08 Nov 2015 20:17:21 +0000. Datasource DataSourceEc2.  Up 16.8
Starting httpd: [  OK  ]
============================================
CONSUMER
============================================


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:47


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:24:06


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:50


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:24:02


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:44


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:46


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
Checking msg: Message for assign09 at: 2015/11/08 14:23:57


QueueUrl: https://sqs.us-east-1.amazonaws.com/413513583861/HW9-SQS
```