

Tutorial 2 - VPCs, Persisting VM Images, Load Balancing

Disclaimer: Subject to updates as corrections are found

Version 0.10

The purpose of this tutorial is to build on tutorial #1 to introduce the use of the Virtual Private Clouds (VPCs) for creating groups of instances within subnetworks, and also to describe how to create and managing virtual machine images in Amazon to support use of EC2 for TCSS 562 projects.

Once creating and launching a virtual machine, the software stack can be customized, and an new image made to save the configuration for replication. A base image with a software stack can be replicated to support **horizontal scaling** of VM instances.

For this tutorial we follow “scenario 2”, which is a VPC with Public and Private Subnets (NAT)

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Scenario2.html

This is in contrast to “scenario 1”, which is a VPC with a Single Public Subnet.

The issue with scenario 1, is that ALL VMs must have public IPs to have internet access to make simple package updates and do things such as “ping google.com”.

Using scenario 2, ALL VMs will have internet access using NAT, which is what is typically used for Virtual Box VM’s.

A load balancer such as haproxy or nginx can be installed on a VM or Amazon’s Load Balancer can be used directly to route traffic across multiple VMs.

Complete the tutorial using any (inexpensive) availability zone within the US-EAST-1 Virginia Region. **Check spot market prices to ensure launching VMs is inexpensive.**

1. Create an Elastic IP Address for your NAT Gateway

Begin by creating a VPC.

Before creating a VPC, create an elastic IP address. This is a public IP address which will be used to route traffic to and from your instances, and the Internet. The elastic IP will be associated with a “NAT gateway”

From the EC2 Dashboard, select “Elastic IPs”:

- ▣ NETWORK & SECURITY
- Security Groups
- Elastic IPs
- Placement Groups
- Key Pairs
- Network Interfaces

And click the button to “Allocate new address”.
 Select “Allocate”

******* WARNING ABOUT ELASTIC IPs *******

Elastic IPs are public Internet IP addresses. They are used to provide Internet connectivity to your EC2 instances and to the NAT gateway.

IT IS ESSENTIAL TO DELETE ELASTIC IP ADDRESSES AFTER COMPLETING THE TUTORIAL.

ELASTIC IP Address costs for US-EAST-1:

.5 cents / hour
 \$.12/day
 \$.84/week
 \$3.60/month
 \$43.20/year

INSTRUCTIONS TO DELETE ELASTIC IP ADDRESS(ES):

Under the EC2 Dashboard:

On the left-hand sidebar, select “Elastic IPs”, under NETWORK & SECURITY.
 Select the ROW in the table for your ELASTIC IP.
 Click the “Actions” button, and first select “Disassociate address”...
 Next, click the “Actions” button again, and select “Release addresses”...

**FAILURE TO DELETE ELASTIC IP ADDRESSES
 WILL RAPIDLY DEplete YOUR AWS CREDITS !!!!**

2. Create your VPC using the VPC Wizard

From the list of services, select “VPC”. The “VPC Dashboard” should appear.

VPC Dashboard

Filter by VPC: None

Resources

[Start VPC Wizard](#) [Launch EC2 Instances](#)

Note: Your Instances will launch in the US East (N. Virginia) region.

You are using the following Amazon VPC resources in the US East (N. Virginia) region:

| | |
|---------------------------------|----------------------------|
| 4 VPCs | 4 Internet Gateways |
| 0 Egress-only Internet Gateways | 10 Subnets |
| 7 Route Tables | 4 Network ACLs |
| 3 Elastic IPs | 0 VPC Peering Connections |
| 0 Endpoints | 2 Nat Gateways |
| 4 Security Groups | 4 Running Instances |
| 0 VPN Connections | 0 Virtual Private Gateways |
| 0 Customer Gateways | |

VPN Connections

Amazon VPC enables you to use your own isolated resources within the AWS cloud, and then connect those resources directly to your own

Click “Start VPC Wizard”.
Then select the second tab, “VPC with Public and Private Subnets”

Step 1: Select a VPC Configuration

- VPC with a Single Public Subnet
- VPC with Public and Private Subnets**
- VPC with Public and Private Subnets and Hardware VPN Access
- VPC with a Private Subnet Only and Hardware VPN Access

In addition to containing a public subnet, this configuration adds a private subnet whose instances are not addressable from the Internet. Instances in the private subnet can establish outbound connections to the Internet via the public subnet using Network Address Translation (NAT).

Creates:
A /16 network with two /24 subnets. Public subnet instances use Elastic IPs to access the Internet. Private subnet instances access the Internet via Network Address Translation (NAT). (Hourly charges for NAT devices apply.)

[Select](#)

Internet, S3, DynamoDB, SNS, SQS, etc.

Amazon Virtual Private Cloud

Public Subnet NAT Private Subnet

Next configure your “VPC with Public and Private Subnets”

This VPC provides network isolation. An Internet Gateway provides internet access, and the NAT gateway supports translation of internal IP addresses to external IP addresses to route traffic over the internet.

Step 2: VPC with Public and Private Subnets

IPv4 CIDR block:* 10.0.0.0/16 (65531 IP addresses available)

IPv6 CIDR block: No IPv6 CIDR Block
 Amazon provided IPv6 CIDR block

VPC name: vpc_1

Public subnet's IPv4 CIDR:* 10.0.0.0/24 (251 IP addresses available)

Availability Zone:* us-east-1e

Public subnet name: e-subnet-pub

Private subnet's IPv4 CIDR:* 10.0.1.0/24 (251 IP addresses available)

Availability Zone:* us-east-1e

Private subnet name: e-subnet-priv

You can add more subnets after AWS creates the VPC.

Specify the details of your NAT gateway (NAT gateway rates apply). [Use a NAT instance instead](#)

Elastic IP Allocation ID:* eipalloc-45e7a474

Service endpoints

Enable DNS hostnames:* Yes No

Hardware tenancy:* Default

[Cancel and Exit](#)

Using the wizard, keep defaults and specify the following:

VPC name: vpc_1

Under Public subnet...

Availability Zone: us-east-1e

Public subnet name: e-subnet-pub

Under Private subnet...

Availability Zone: us-east-1e

Private subnet name: e-subnet-priv

For the NAT gateway

Elastic IP Allocation ID: select the elastic IP address you previously allocated

Then press the "Create VPC" button to create your VPC.

This wizard automatically creates:

Your VPC

Your Public and Private Subnets

An Internet Gateway

A DHCP Option Set, which provide the Domain Name Server functionality

A NAT Gateway

Once your VPC has been created, go to "VPC" under Services, and inspect all of the artifacts that have been created.

***** **WARNING ABOUT NAT GATEWAYS** *****

NAT Gateway's are special virtual appliances that provide gateway/router functionality between your subnetworks of the VPC and the internet.

IT IS ESSENTIAL TO DELETE NAT GATEWAYS AFTER COMPLETING THE TUTORIAL.

NAT Gateway costs for US-EAST-1:

4.5 cents / hour
\$1.08/day
\$7.56/week
\$32.40/month
\$388.80/year

INSTRUCTIONS TO DELETE NAT GATEWAY:

Under the VPC Dashboard:

On the left-hand sidebar, select "NAT Gateways".
Select the ROW in the table for your NAT Gateway.
Click the "Actions" button, then select "Delete NAT Gateway"

**FAILURE TO DELETE YOUR NAT GATEWAY
WILL RAPIDLY DEplete YOUR AWS CREDITS !!!!**

A NAT instance can be used in place of a NAT gateway. Selecting "NAT instance" will launch an EC2 instances to act as the Gateway/Router VM. NAT instances are billed at on-demand VM rates as shown at <https://www.ec2instances.info/>. Inexpensive "t2" instances can be used, but they provide poor network performance (Low to Moderate).

3. Launch an EBS-backed Amazon EC2 Ubuntu 16.04 instance in your public subnet

Next we will launch a VM into the public subnet of your newly created VPC.

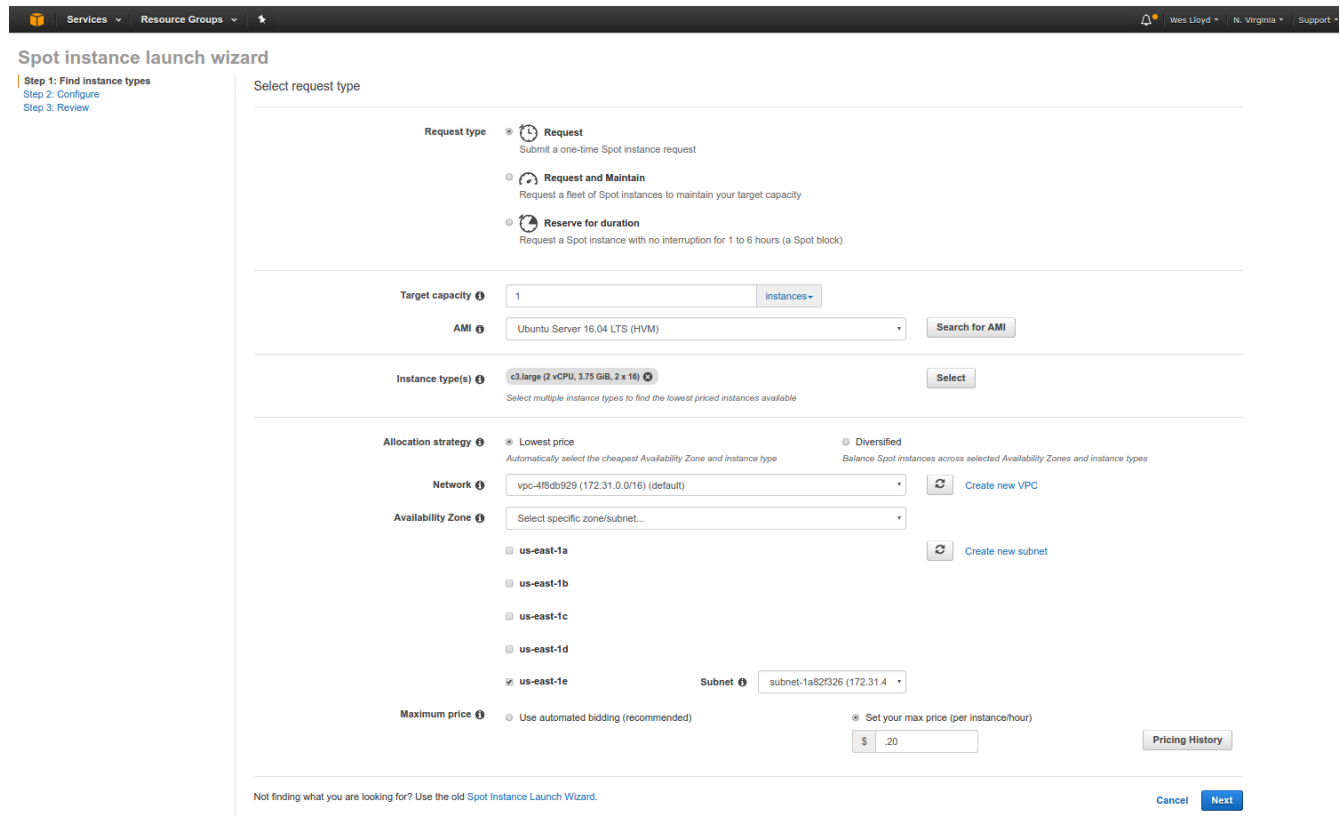
From the EC2 Dashboard, click on the "Spot Requests" option on the left-hand menu:

- ☐ INSTANCES
- Instances
- | **Spot Requests**
- Reserved Instances
- Scheduled Instances
- Dedicated Hosts

Next, Select the “Request Spot Instances” button:



As in the screenshot, configure the spot request as follows.



If the parameter is not specified, keep the default value.

Request type: * Request (one-time)

Target capacity: 1

AMI: (select from the dropdown) Ubuntu Server 16.04 LTS (HVM)

Instance type: c3.large

Network: For the network, select **“vpc_1”**

This is the VPC we created with the wizard.

Next for the Availability Zone, choose “Select specific zone/subnet...”

Select, "us-east-1e"

For the subnet, select your "e_subnet_pub".

Check the pricing history (click the button)

Set your max price (per instance/hour)

Select a value at your discretion. (check the pricing graph)
At least 10 cents is recommended.

Click on the <Next> button

Now, the next screen is "Step 2 - Configure".

EBS volumes:

Keep the capacity at 8GB.

Key pair name:

If this is the first time you've launched a VM, you'll need to create a new keypair. When generating a keypair, you'll download the key file to your machine and use in place of a plaintext password to log into the cloud VM.

If using the Windows Putty client to access the VM, follow these instructions to convert the keypair for use:

<http://www.cnx-software.com/2012/07/20/how-use-putty-with-an-ssh-private-key-generated-by-openssh/>

If using Ubuntu/Linux, no conversion is required.

Security groups: default

Auto-assign IPv4 Public IP: Select "Enable" from the dropdown.

** We want to assign this VM a public IP. We will use it to access VMs in our VPC that only have private, internal Ips **

Next, click on the review button.

Check over everything, and then submit your spot request.

http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Scenario1.html

4. Log into your Amazon EC2 Spot Instance

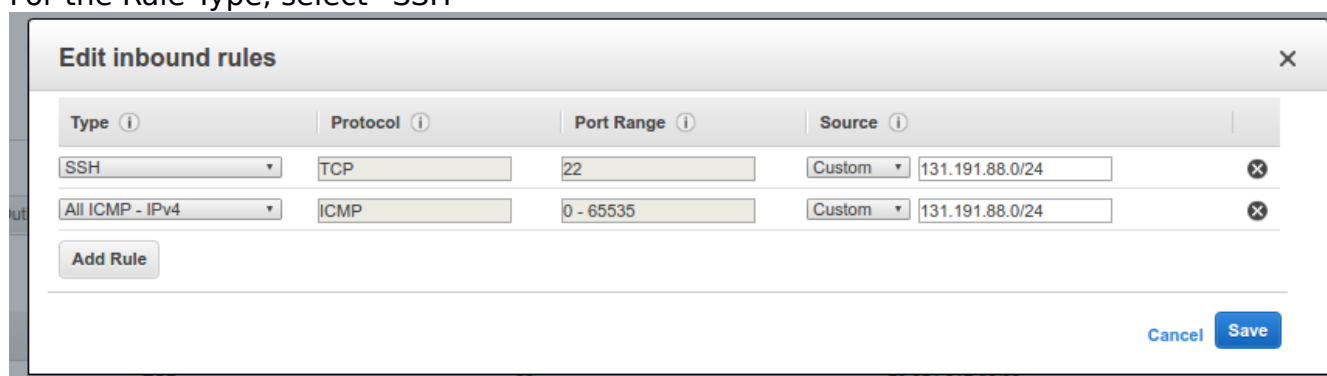
Once the spot request has been submitted, let's check what is your IP address. In your web browser, open Google search, and type in "What is my IP?".

Your IP address should appear.
Note the first 3 numbers.
Let's add SSH permission for your CIDR network block.

If your IP is for example 120.118.53.108, then your 24-bit CIDR block which would include all 255 addresses on the local subnet, and will be "120.118.53.0/24".

Go to EC2 - Instances, and select your instance once it appears.
In the instance description pane, find "Security Groups".
Click on "default"...

This takes you into the Security Groups editor.
Click on the "Inbound" tab.
Click on "Edit".
Click "Add Rule"
For the Rule Type, select "SSH"



For the Source, add your CIDR block, e.g. 120.118.53.0/24
Optionally, you can add the "All ICMP - IPv4" permission. This enables you to "ping" VM instances.

ADDITIONALLY, please two more rules, one for "HTTP" which is port 80, and one "Custom TCP Rule", select port 8080.

First Rule:

Select Type: HTTP
The port will default to 80.
And specify the Custom source, add your CIDR block for your inbound network, e.g. 120.118.53.0/24.

Second Rule:

Select Type: HTTP
Specify a custom port of 8080.
And specify the VPC's CIDR address block, e.g. 10.0.0.0/16...

These security group assignments will now apply to all VMs we create within this VPC!

Now, navigate back to "Instances", find your instance, and select it.
Note your instance's "IPv4 Public IP".

If you've configured the "All ICMP - IPv4" permission, try pinging your instance **from your laptop**... Use the windows, MAC, or Linux command line on your laptop...

```
$ ping 54.165.102.178
PING 54.165.102.178 (54.165.102.178) 56(84) bytes of data.
64 bytes from 54.165.102.178: icmp_seq=1 ttl=41 time=79.2 ms
64 bytes from 54.165.102.178: icmp_seq=2 ttl=41 time=71.7 ms
```

CTRL-C will exit.

Next, using your putty or ssh terminal client, and your keypair, SSH into the instance as follows. The first time you're connecting to a new IP, you may need to acknowledge the host's authenticity.

```
ssh -i <your_key_file_name> ubuntu@<the IPv4 Public IP>
```

Example Output:

```
$ssh -i <your_key_file_name> ubuntu@54.165.102.178
The authenticity of host '54.165.102.178 (54.165.102.178)' can't be established.
ECDSA key fingerprint is SHA256:35893chQunQ2eVt908X8jxyvJpYJb0NdxOQjmi6U3OQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.165.102.178' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-64-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

```
ubuntu@ip-172-31-56-31:~$
```

5. Configure your instance

Let's first download the SSH keypair to your instance from your local machine.

If your using MAC or Linux, from the directory where the keypair is located, simply use secure copy:

```
scp -i <your key file> <your key file> ubuntu@<public IP of VM>:.
```

An example of this command looks like this:

```
scp -i mykey.pem mykey.pem ubuntu@34.206.59.188:.
```

Next, let's install some software:

```
sudo apt-get install haproxy  
sudo apt-get install tomcat7
```

Next let's configure haproxy. We will move the original config file aside, and create a new one.

```
cd /etc/haproxy  
sudo mv haproxy.cfg haproxy.cfg.bak  
sudo nano haproxy.cfg
```

Copy the following contents:

```
global  
  log 127.0.0.1 local0 notice  
  maxconn 2000  
  user haproxy  
  group haproxy
```

```
defaults
  log global
  mode http
  option httplog
  option dontlognull
  retries 3
  option redispatch
  timeout connect 5000
  timeout client 10000
  timeout server 10000

frontend myserver
  bind proxy:80
  option http-server-close
  option forwardfor
  default_backend mytomcat

backend mytomcat
  mode http
  balance roundrobin
  server ubuntu1 localhost:8080 cookie A check inter 5000 rise 2 fall 5
```

Verify that the config file is correct:

```
haproxy -c -f /etc/haproxy/haproxy.cfg
```

If so, then restart (or start for the first time) haproxy:

```
sudo /etc/init.d/haproxy restart
```

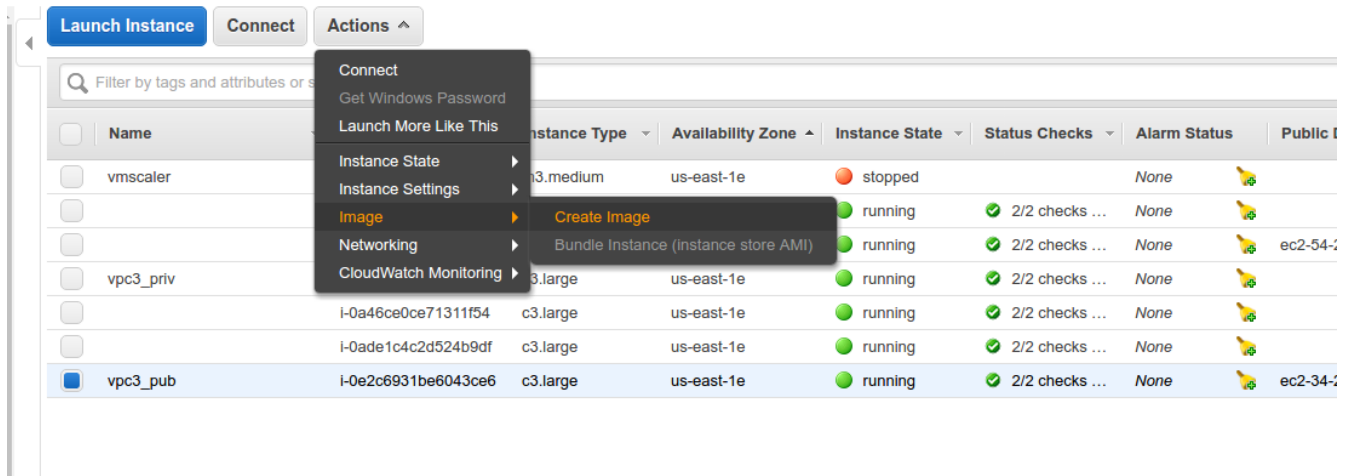
In your hosts file, add an entry which defines “proxy” to be the private IP of your VM:

```
sudo nano /etc/hosts
```

Add a line after localhost:
Replace 10.0.0.248 with your VM’s private IP
10.0.0.248 proxy

6. Create an Amazon Machine Image (AMI) of your instance

Now let’s create an image of your VM with the software installed:



In the “Create Image” dialog, provide:

Image name: worker_vm
 Leave all other options as-is.

Click “Create Image”...

An image of your instance with your key, tomcat7, and haproxy is created after a few minutes.

Once the image has been created, check under “AMIs” for your “worker_vm”
 The worker VM will have an “AMI ID”. This number, will be something like “ami-0a6ec428”. This is the unique identifier for your instance.

Note this ID.

7. Launch spot instance worker VMs using your AMI on the private subnet within your VPC

Now, navigate back to the Spot instance launch wizard.

We will now launch 2 x c3.large instances of this newly create image.

From the spot instance launch wizard, in “Step 1: Find instance types”

Target capacity: 2

AMI: Under “Custom”, select “Search for AMI” and select your newly created AMI.

Instance type: c3.large

Network: vpc_1

Availability Zone: Select specific zone/subnet...

Select (X) us-east-1e

Subnet: select the “e-subnet-priv”

This will launch these new spot instances within your private subnet of your VPC.

Set price as before.

Click “Next” and go to “Step 2: Configure”

Be sure that your key pair is specified.
Otherwise, no other changes are required.

Click “Review” and go to “Step 3: Review”, and then launch the new instances.

8. Configure the new instances

After sometime, these two new instances will appear.

Using your keypair let’s SSH into the first instance:

```
ssh -i <your_key_pair> ubuntu@10.0.1.xxx
```

While on the instance, note that we are able to ping www.google.com:

```
ping www.google.com
```

Does it work?

Network address translation has allowed our VM with a private IP access the internet.

On the instance, we will update the tomcat7 ROOT application index.html file

```
cd /var/lib/tomcat7/webapps/ROOT
```

```
sudo mv index.html index.old
```

Using nano, let’s create a new index.html

```
sudo nano index.html
```

Paste the contents:

```
<html>
<body>
Hello from VM #1
</body>
</html>
```

Now, repeat these steps for the second instance, but this time, let’s say “VM #2”

```
<html>
<body>
Hello from VM #2
</body>
</html>
```

9. Configure haproxy on your public instance

Now, let's log out of the second instance and update the `/etc/haproxy/haproxy.cfg` file.

At the bottom of the file, under backend mytomcat, add the following two lines at the end. Revise the IP addresses to reflect the IP addresses your private IPs:

```
server ubuntu1 10.0.1.69:8080 cookie A check inter 5000 rise 2 fall 5
server ubuntu1 10.0.1.196:8080 cookie A check inter 5000 rise 2 fall 5
```

10. Test your worker pool

Now using a web browser, enter the public IP address of your first virtual machine.

If everything works correctly, you should see the tomcat welcome page:

It works !

If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

This is the default Tomcat home page. It can be found on the local filesystem at: `/var/lib/tomcat7/webapps/ROOT/index.html`

Tomcat7 veterans might be pleased to learn that this system instance of Tomcat is installed with `CATALINA_HOME` in `/usr/share/tomcat7` and `CATALINA_BASE` in `/var/lib/tomcat7`, following the rules from `/usr/share/doc/tomcat7-common/RUNNING.txt.gz`.

You might consider installing the following packages, if you haven't already done so:

tomcat7-docs: This package installs a web application that allows to browse the Tomcat 7 documentation locally. Once installed, you can access it by clicking [here](#).

tomcat7-examples: This package installs a web application that allows to access the Tomcat 7 Servlet and JSP examples. Once installed, you can access it by clicking [here](#).

tomcat7-admin: This package installs two web applications that can help managing this Tomcat instance. Once installed, you can access the [manager webapp](#) and the [host-manager webapp](#).

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager-gui". The host-manager webapp is restricted to users with role "admin-gui". Users are defined in `/etc/tomcat7/tomcat-users.xml`.

When you refresh your browser, the content should alternate between the splash page and our two hellow world pages:

Hello from VM #1

Hello from VM #2

11. Submit for Grading

Using Google Docs or MS Word, create a PDF document that captures the following content for submission to CANVAS:

1. Content of haproxy.cfg file. Copy the entire file as plain text to the Document.

Label the start of the file:

haproxy.cfg File Contents:

<place file data here>

Add a delimiter line at the bottom:

2. Next, SSH to your first VM. This is the VM on the public network with a public IP address.

3. Capture and label the output to your PDF document for the following commands and include this on your submission:

```
time wget http://localhost
```

```
time wget http://localhost
```

```
time wget http://localhost
```

This task will create a local file called "index.html", and then "index.html.1" and "index.html.2"...

4. Capture and label the content to your PDF document for each of the following.

```
wc -l index.html*
```

```
ls -alt index.html*
```

```
md5sum index.html*
```

5. Include and label the output of each file in the PDF:

```
cat index.html
```

```
cat index.html.1
```

```
cat index.html.2
```

Once capturing the relevant data to the document, submit the document to Canvas for grading.

Teams of Two (Pair Programming) is permitted for Tutorial #2...

Names of both team members must appear at the top of the PDF file.

Only one member should submit the PDF.

12. Cleanup

At the end of the tutorial, be sure to **TERMINATE** all EC2 instances, and **DELETE** all EBS volumes. Also, be sure to **DELETE your NAT Gateway**, and **DELETE any Elastic IP addresses** that were created.

Failing to delete all cloud artifacts will result in loss of AWS credits leading to potential charges to your credit card.

As a reminder (again), when concluding, use MUST:

- **DELETE your NAT gateway**
- **DELETE any Elastic IP addresses**

Failing to do so will result in charges.