## Scenario-1

* Purpose
  + To create the Microsoft Windows Server EC2 instance
* Steps
  + EC2 - Launch Instance Page
  + Choose Windows Server EC2
  + Choose the default settings for Storage and Network and click Finish
  + Download the Remote Desktop Profile (RDP) file
  + Credentials
    - Username - “Administrator”
    - Password
      * To get the upload, upload the PEM file
      * Choose Decrypt password to retrieve the password
  + Open the RDP file, type-in the Credentials to connect to Windows Server

## Scenario-2

* Purpose
  + To create a custom AMI
* Steps
  + EC2 - Launch Instance Page
  + Choose Ubuntu
  + Network Settings
    - Inbound Rules
      * HTTP / 80
      * SSH / 22
    - Outbound Rules
      * All traffic on all protocols
  + Choose the default settings for Storage and click Finish
  + SSH to Ubuntu and install the following
    - Install NGINX
      * Run the below commands
        + “sudo apt update”
        + “sudo apt install nginx”
      * Verify if NGINX is working fine
        + “curl <http://localhost:80>”
    - Install Docker
      * Run the below commands
        + “sudo apt update”
        + “sudo apt install docker.io”
        + To add the current ubuntu user to the docker group

“sudo usermod -aG docker ubuntu”

* + - * + Logout and login to the console

“Exit”

* + To create the AMI,
    - On the instance, Right click Instance Settings
    - Create Image from this
      * This process will initiate the creation of AMI
      * Go to AMI Page to see the AMI that is in the process of getting created
      * Wait for few minutes to see that the AMI is completed created
        + The status column can be checked for this
    - Once the AMI is created, the AMI will now contain the following
      * Base Image as Ubuntu
      * Docker and NGINX software installed
  + To launch the instance using this AMI
    - On the AMI, right click, create Instance
    - Proceed with the default settings
    - After connecting to the instance using SSH, verify the following
      * To verify Docker
        + Run docker commands such as “docker images” “docker ps --all”
      * To verify NGINX
        + “curl <http://localhost:80>”

## Scenario-3

* Purpose
  + To setup the EC2 resources to demonstrate Security Group related scenarios
* Steps
  + Ubuntu-1
    - EC2 - Launch Ubuntu
    - Network Settings
      * Ensure the following is in place
        + Inbound Rules

All traffic allowed on all ports numbers with “Any IP v4” selected for the source

* + - * + Outbound Rules

All traffic allowed on all ports numbers with “Any IP v4” selected for the source

* + - Install NGINX
  + Ubuntu-2
    - EC2 - Launch Ubuntu
    - Network Settings
      * Ensure the following is in place
        + Inbound Rules

All traffic allowed on all ports numbers with “Any IP v4” selected for the source

* + - * + Outbound Rules

All traffic allowed on all ports numbers with “Any IP v4” selected for the source

* + - Install NGINX

## Scenario-4

* Purpose
  + Inbound Rules Verification
* Steps
  + Attempt to reach Ubuntu-1 through the following ways
    - HTTP
      * From the Laptop/PC Browser
        + http:<public\_ip\_ubuntu\_1>:80
        + This should be accessible because the inbound rules don't have any restriction
      * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, issue the below command

“curl http://<public\_ip\_ubuntu\_1>:80

* + - * + This should be accessible because the inbound rules don't have any restriction
    - SSH
      * From the Laptop/PC
        + Launch Command Prompt / Terminal

Do a SSH to Ubuntu-1

This should be accessible because the inbound rules don't have any restriction

* + - * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, attempt doing a SSH to Ubuntu-1
        + This should be accessible because the inbound rules don't have any restriction

## Scenario-5

* Purpose
  + Outbound Rules Verification
* Steps
  + From AWS CloudShell, connect to Ubuntu-1 with SSH
  + Attempt to reach to external systems from Ubuntu-1 through the following ways
    - Curl
      * Issue the below command
        + “ping google.com”
      * This should be accessible because the outbound rules don't have any restriction
    - HTTP
      * From the terminal, issue the below command
        + “curl http://<public\_ip\_ubuntu\_2>:80
      * This should be accessible because the outbound rules don't have any restriction

## Scenario-6

* Purpose
  + Reconfigure Inbound Rules to dis-allow traffic
* Steps
  + For Ubuntu-1, reconfigure the Security Group Settings
    - Inbound Rules
      * Remove the rule that “allows traffic from elsewhere”
      * Click save
    - Outbound Rules
      * Ensure that for Outbound rules “all traffic allowed on all protocols” is defined”
  + Attempt to reach Ubuntu-1 through the following ways
    - HTTP
      * From the Laptop/PC Browser
        + http:<public\_ip\_ubuntu\_1>:80
        + This should be NOT accessible because they are no inbound rules configured
      * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, issue the below command

“curl http://<public\_ip\_ubuntu\_1>:80

* + - * + This should be NOT accessible because they are no inbound rules configured
    - SSH
      * From the Laptop/PC
        + Launch Command Prompt / Terminal

Do a SSH to Ubuntu-1

This should be NOT accessible because they are no inbound rules configured

* + - * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, attempt doing a SSH to Ubuntu-1
        + This should be NOT accessible because they are no inbound rules configured

## Scenario-7

* Purpose
  + Reconfigure Outbound Rules to dis-allow traffic
* Steps
  + For Ubuntu-1, reconfigure the Security Group Settings
    - Inbound Rules
      * All traffic allowed on all ports numbers with “IP v4” selected for the source
    - Outbound Rules
      * Remove the rule that “allows traffic from elsewhere”
      * Click save
  + From AWS CloudShell, connect to Ubuntu-1 with SSH
  + Attempt to reach to external systems from Ubuntu-1 through the following ways
    - Curl
      * Issue the below command
      * “ping google.com”
      * This should be NOT accessible because they are no outbound rules configured
    - HTTP
      * From the terminal, issue the below command
        + “curl http://<public\_ip\_ubuntu\_2>:80
      * This should be NOT accessible because they are no outbound rules configured

## Scenario-8

* Purpose
  + Reconfigure Inbound Rules to allow traffic only from certain sources
* Steps
  + For Ubuntu-1, reconfigure the Security Group Settings
    - Inbound Rules
      * Remove the rule that “allows traffic from elsewhere”
      * Add a rule with the below settings
        + Type - All traffic
        + Source

Choose Custom

Type in the ‘public IP of Ubuntu-2’ and Save

* + - * This rule says that “only traffic from Ubuntu-2 is allowed onto Ubuntu-1”
    - Outbound Rules
      * Ensure that for Outbound rules “all traffic allowed on all protocols” is defined”
  + Attempt to reach Ubuntu-1 through the following ways
    - HTTP
      * From the Laptop/PC Browser
        + http:<public\_ip\_ubuntu\_1>:80
        + This should be NOT accessible because the inbound rules are configured in a manner that allows access only from Ubuntu-2
      * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, issue the below command

“curl http://<public\_ip\_ubuntu\_1>:80

* + - * + This should be accessible because the inbound rules are configured in a manner that allows access from Ubuntu-2
    - SSH
      * From the Laptop/PC
        + Launch Command Prompt / Terminal

Do a SSH to Ubuntu-1

This should be NOT accessible because the inbound rules are configured in a manner that allows access only from Ubuntu-2

* + - * From the Ubuntu-2
        + Using Cloud Shell terminal, SSH to Ubuntu-2
        + From the terminal, attempt doing a SSH to Ubuntu-1
        + This should be accessible because the inbound rules are configured in a manner that allows access from Ubuntu-2