

GenC Use Cases



Table of Contents

1. Use Case Name: Create Your First Amazon EC2 Instance (Windows)	3
2. Use Case Name: Expanding EBS Volumes on Windows Instances	4
3. Use Case Name: Monitor Amazon CloudWatch Security Logs for failed SSH attempts.....	5
4. Use Case Name: First steps into the Linux console	7
5. Use Case Name: Serve your files using the CloudFront CDN	8
6. Use Case Name: Deploy WordPress using CloudFormation	9
7. Use Case Name: Create Your First Amazon S3 Bucket	10
8. Use Case Name: Introduction to AWS Lambda.....	11
9. Use Case Name: Expanding EBS Volumes on Linux Instances	12
10. Use Case Name: Create an EBS-Backed Linux AMI	13
11. Use Case Name: Introduction to the Elastic File System	14
12. Use Case Name: Configuring a Static Website with S3 and CloudFront	15
13. Use Case Name: Using S3 Bucket Policies and Conditions to Restrict Specific Permissions.....	17
14. Use Case Name: Introduction to CloudWatch	19
15. Use Case Name: Introduction to DynamoDB	20
16. Use Case Name: Creating your first Classic Load Balancer	23
17. Use Case Name: Creating your first Auto Scaling group.....	26
18. Use Case Name: Create Your First Amazon RDS Database	29
19. Use Case Name: Managing Instance Volumes Using EBS.....	32
20. Use Case Name: Working with Amazon EC2 Auto Scaling Groups and Classic ELB	36

1. Use Case Name: Create Your First Amazon EC2 Instance (Windows)

Brief Description	Amazon Elastic Compute Cloud (Amazon EC2) is one of the most used and basic web services that provides resizable compute services in the AWS Cloud. The servers are resizable so that the user can easily scale up or scale down the number of server instances to fulfill the computing demands. Amazon EC2 removes the up-front investment or rent for hardware and enables the user to build and run applications faster.
Objective	To create, configure and launch the first Amazon EC2 instance running Microsoft Windows Server
Service(s) Involved	AWS EC2
Prerequisites	N/A
Lab Steps	<ol style="list-style-type: none">1. Log into console2. Launch Windows Instance3. Connect to Windows Instance using RDP using Connect option in EC2 Console

2. Use Case Name: Expanding EBS Volumes on Windows Instances

Brief Description	<p>Amazon Elastic Block Store (EBS) provides persistent block storage volumes for use with Amazon EC2 instances. Users can set any size they want as long as it is bigger than the current size.</p> <p>Using EBS, a user can increase the storage volume of even a critical Windows Server running on AWS EC2 with zero downtime.</p> <p>Note: An EBS volume can only be mounted to a single EC2 instance at a time.</p>
Objective	To extend the EBS Volume attached to the Windows EC2 Instance on AWS
Services Involved	AWS EC2 and EBS
Prerequisites	Created Windows Instance
Lab Steps	<ol style="list-style-type: none">1. Create Windows Instance2. Create Snapshot of EBS Volume attached to the Instance3. Create new EBS Volume from the Snapshot taken <p>Note: Ensure the Availability Zone is same for the EBS volume as the instance to proceed with the creation.</p> <ol style="list-style-type: none">4. Expand Volume to a size greater than the previous one5. Detach Volume attached to the Windows Instance by stopping the instance.6. Attach newly created EBS Volume to the Windows Instance with device as /dev/sda17. Start the instance with newly attached volume.8. Delete old Volume

3. Use Case Name: Monitor Amazon CloudWatch Security Logs for failed SSH attempts

Brief Description	<p>Amazon CloudWatch is a monitoring service for the resources and applications run on the AWS Cloud. It is used to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in AWS resources.</p> <p>It monitors the occurrence of Secure Shell (SSH) attempts to the EC2 instance and create an alarm for frequent invalid authentication attempts.</p>
Objective	To monitor CloudWatch Security Logs for invalid (SSH) access attempts / connection requests to the application servers
Services Involved	EC2, SNS, CloudWatch (Alarm, Logs)
Prerequisites	Familiarity with CloudWatch service
Lab Steps	<ol style="list-style-type: none"> 1. Create Linux Instance and start the session: <ol style="list-style-type: none"> a. When the status of the instance is changed to running, choose Actions > Instance Settings > Attach/Replace IAM Role. From the IAM role dropdown menu, select GenC and click Apply. b. Select the instance, click Connect and select Session Manager and click on Connect. Note: Wait until connection to session manager opens. 2. SSH into the instance and install CloudWatch Agent <ol style="list-style-type: none"> a. Before installing CloudWatch agent, keep in mind to change the default Region from us-east-1 to us-east-2 (Ohio) in the /etc/awslogs/awscli.conf file and name the CloudWatch Log Group in /etc/awslogs/awslogs.conf file. Go to Step 2 in this link for installation. Once the awslogs service starts, user can find newly created log group and log stream in the CloudWatch console. b. Go to CloudWatch, locate and select Log group created by CloudWatch Agent. 3. Create an SNS Topic 4. Subscribe to the SNS Topic created with a protocol of email (confirm the subscription) 5. Create CloudWatch Alarm: <ol style="list-style-type: none"> a. Click on Select metric > Logs for Metrics > select the log group created and check the IncomingLogEvents metric. b. Click on Graphed metrics, select 1 Minute as Period and Sum as Statistic.

- c. Select **Static** and **Greater/Equal** radio buttons and **2** as threshold value under **Conditions**.
- d. On the next page, select **in Alarm** and select the SNS topic created earlier under **Notification** section and click on **Create Alarm**.

Note: The initial **State** is likely **INSUFFICIENT_DATA**.

Depending on timing, the **State** should transition to either **OK** or **ALARM**. Although the Alarm is created, user has not setup a Metric Filter yet. Set it up to match against a specific pattern within the logs sent from the EC2 instance to CloudWatch.

6. Create Metric Filter:

- a. Select the log group created by CloudWatch Agent and then click **Create Metric Filter**.
- b. **Filter Pattern:** Enter *[Mon, day, timestamp, ip, id, status = Invalid, ...]*
- c. If the log stream already has data from the instance, select it in the drop-down and then click Test Pattern (it may find one or two occurrences of "invalid" users if any) and click **Assign Metric**. Enter Metric name and click **Create Filter**.
- d. If there are any invalid user attempts running EC2 instance (beyond 2x), alarm triggers and sends notification to respective SNS topic.

4. Use Case Name: First steps into the Linux console

Brief Description	Linux provides a CLI (Command Line Interface) that interprets a user's commands / script files and communicates with the Amazon EC2 instance what to do with them. Often referred to as the shell, console, terminal, prompt etc.,		
Objective	To create an EC2 Linux instance, connect using SSH and practice basic Linux commands		
Services Involved	AWS EC2		
Prerequisites	N/A		
Lab Steps	<ol style="list-style-type: none"> 1. Log into console 2. Launch a basic EC2 Linux Instance 3. Connect to Linux Instance using Secure Shell (SSH) into the Console. 4. Get familiar with a few below basic Linux commands (all in lower case): 		
	S.No	Command	Usage
	1	cd	To change user's present working directory. Type <code>cd</code> followed by <code>/var/www</code> to navigate to that directory.
	2	ls	To list contents of a directory in output
	3	mkdir	To create a new directory
	4	pwd	To print the full filename of current working directory

```
$ cd [path-to-
directory]
cd /var/www
```

```
$ ls [files-or-
directories]
```

```
$ mkdir [dir-name]
```

```
$ pwd
/home/cloudlabs
```

5. Use Case Name: Serve your files using the CloudFront CDN

Brief Description	Amazon CloudFront is a CDN (Content Delivery Network) that speeds up the distribution of static and dynamic web content. It retrieves data from Amazon S3 bucket and distributes the content with the best possible performance through multiple data centers called edge locations. The nearest edge location is routed when the user requests for data, resulting in lowest latency (time delay), low network traffic, fast access to data and no minimum usage commitments. It enables user to serve large files such as videos, with a much better download rate and potentially a more solid connection.
Objective	To make the objects in S3 Bucket accessible worldwide without any latency.
Services Involved	S3, Amazon CloudFront
Prerequisites	Creating a Bucket in S3 and knowledge on CloudFront
Lab Steps	<ol style="list-style-type: none"> 1. Create an S3 Bucket 2. Upload a sample file (any data) into the bucket. Now, if the user tries to access the object using Object URL, it displays Permission Denied. 3. Hence, make the object public by: <ul style="list-style-type: none"> • Uncheck the 'Block all Public Access' option under Permissions tab, • Go to either Access Control List (ACL) or Bucket Policy (JSON script) under Permissions tab, • As part of this lab, use ACL by setting everyone option under public access. 4. After making the bucket public, the user can access the objects through: <a href="https://<bucket>.s3.amazonaws.com/<file>">https://<bucket>.s3.amazonaws.com/<file> 5. Navigate to CloudFront service and select Create Distribution as it should be the same content, but now it's being delivered from a local CDN server. 6. Choose Web Distribution and choose the S3 bucket created recently, under Original Domain Name. Note: If you are not able to list your bucket name, enter <bucketname.s3.amazonaws.com> 7. Leave the rest of the default options and click Create Distribution. Once the Distribution is created, the user can access the object through: <a href="https://<distribution domain>/<file>">https://<distribution domain>/<file> which avoids latency to the users across the world.

6. Use Case Name: Deploy WordPress using CloudFormation

Brief Description	<p>AWS CloudFormation enables users to create and manage a collection of the below listed AWS resources predictably and repeatedly without having to manually create and configure them:</p> <ul style="list-style-type: none"> • Amazon EC2 • Amazon Elastic Block Store • Amazon SNS • Elastic Load Balancing and • Auto Scaling <p>With CloudFormation, users can model complete complex infrastructure and applications in a text file, or template using easy-to-understand JSON or YAML format.</p>
Objective	To launch an EC2 (Linux) instance with a full WordPress installation.
Services Involved	AWS CloudFormation (internally creates EC2 instance)
Prerequisites	<p>Knowledge on launching an EC2 Instance</p> <p>Note: Before launching a CloudFormation Stack, go through the template file that consists code (attached here), to know what resources are being launched.</p>
Lab Steps	<ol style="list-style-type: none"> 1. Create a CloudFormation Stack <ol style="list-style-type: none"> a. Upload a Template file (attached above) b. Specify Stack Details and respective parameters 2. Check Stack Status 3. View Stack Resources

7. Use Case Name: Create Your First Amazon S3 Bucket

Brief Description	Amazon Simple Storage Service (Amazon S3) is web-based cloud storage service. With Amazon S3, a user can store and retrieve any amount of data at any time, from anywhere on the web, as it is designed for large-capacity, low-cost storage provision across multiple geographical regions. S3 has two primary entities called buckets and objects. Objects are stored inside buckets. Bucket names must be globally unique irrespective of which region they are created in.
Objective	To create an S3 bucket
Services Involved	Amazon S3
Prerequisites	N/A
Lab Steps	<ol style="list-style-type: none"> 1. Create a bucket 2. Create a folder 3. Upload files to the bucket. Now, if the user tries to access the object using Object URL, it displays Permission Denied. 4. Hence, we are making the object public by: <ul style="list-style-type: none"> • Under Permissions tab, uncheck the 'Block all Public Access' option • Under Permissions tab, go to either Access Control List (ACL) or Bucket Policy (JSON script) • As part of this lab, we use ACL by setting everyone option under public access. 5. After making the bucket public, the user is able to access the objects through: <a href="https://<bucket>.s3.amazonaws.com/<file>">https://<bucket>.s3.amazonaws.com/<file> 6. Set metadata for S3 bucket when uploading an object into S3 bucket. User can add metadata under Set Properties section – OR– 7. After uploading the object, under Properties section, the user can add/delete/edit the metadata 8. Delete the associated objects (folders and files) stored in the bucket and then delete the S3 bucket.

8. Use Case Name: Introduction to AWS Lambda

Brief Description	<p>AWS Lambda is Amazon's most reliable serverless AWS Compute service. Being serverless doesn't mean that servers are nowhere in the play. Instead, it means that users don't have to worry about provisioning or maintaining servers. AWS Lambda automatically provides us whatever resources required to run code to react to events. Users pay only for the compute time they consume and there is no charge when code is not running.</p> <p>AWS Lambda can be used for varied tasks including image processing, log processing, and security automation.</p>
Objective	<ul style="list-style-type: none"> • Understand and create Lambda functions using AWS Console • Test Lambda functions created
Services Involved	Amazon Lambda, S3
Prerequisites	<p>Basic programming knowledge is required to do this lab.</p> <p>Familiarity in coding in any one of the following: C#/PowerShell, Go Lang, Java, Node.js, Python, Ruby</p>
Lab Steps	<ol style="list-style-type: none"> 1. Create a Lambda Function 2. Choose the option: Use a Blueprint 3. From the list provided, select 'S3-Get-Object-Python' (Recommended to make use of other available Blueprint options to gain proficiency) 4. Click Configure 5. Name the function 6. Select 'Use an existing Role' (LambdaRole) under Execution Role 7. In parallel, create an S3 bucket. After creation, go back to Lambda Configuration Page. 8. Choose the bucket created from the list of buckets available and enable the trigger. 9. Let the Event type to be default – 'All object create events' 10. Now if you make any changes or upload any objects in your S3 bucket, your lambda function will be triggered.

9. Use Case Name: Expanding EBS Volumes on Linux Instances

Brief Description	<p>Amazon Elastic Block Store (EBS) provides persistent block storage volumes for use with Amazon EC2 instances. Users can set any size they want as long as it is bigger than the current size.</p> <p>Using EBS, a user can increase the storage volume of even a critical Linux Server running on AWS EC2 with zero downtime.</p> <p>Note: An EBS volume can only be mounted to a single EC2 instance at a time.</p>
Objective	To extend the EBS Volume attached to the Linux EC2 Instance on AWS
Services Involved	AWS EC2 and EBS
Prerequisites	Created Linux Instance
Lab Steps	<ol style="list-style-type: none"> 1. Create Linux Instance 2. Create Snapshot of EBS Volume attached to the Instance 3. Create new EBS Volume from the Snapshot taken <p>Note: Ensure the Availability Zone is same for the EBS volume as the instance to proceed with the creation.</p> 4. Expand Volume to a size greater than the previous one 5. Detach Volume attached to the Linux Instance by stopping the instance. 6. Attach newly created EBS Volume to the Linux Instance 7. Start the instance with the newly attached volume. 8. Delete old Volume

10. Use Case Name: Create an EBS-Backed Linux AMI

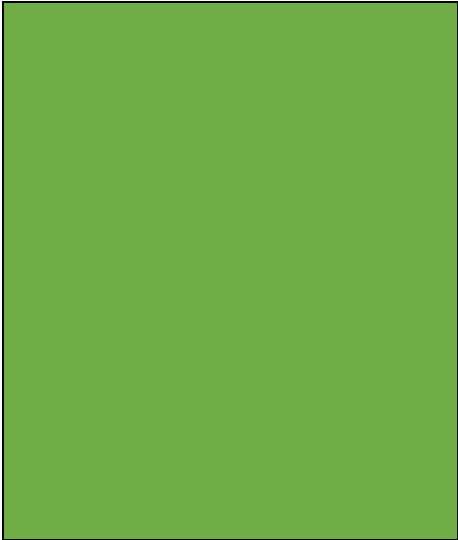
Brief Description	An Amazon Machine Image (AMI) provides the information required to launch an EC2 instance. An 'EBS-backed' instance is an EC2 instance which uses an EBS volume as its root device. User can customize an EC2 instance and then save the configuration as a custom AMI for private or public use. Every EC2 instance launched selecting the customized AMI will contain any software or file that the user has previously added.
Objective	To setup a webserver EC2 instance starting from a Linux AMI, and then generate a new AMI.
Services Involved	EC2
Prerequisites	Knowledge to create an EC2 instance and connect to it with SSH
Lab Steps	<ol style="list-style-type: none"> 1. Create a basic EC2 instance for serving static files. <ol style="list-style-type: none"> a. After selecting an AMI, log in to SSH using keypair and install nginx webserver. 2. Create an AMI starting from an EBS-backed instance: <ol style="list-style-type: none"> a. Go to Instances section of EC2 Console b. Find the previously created instance, select it and then right-click it. c. Select Image submenu and click on Create Image. Check the status of AMI creation in the Snapshot and AMI sections 3. Make an AMI public or private: <ol style="list-style-type: none"> a. Select created AMI, click on Permissions Tab > Edit button

11. Use Case Name: Introduction to the Elastic File System

Brief Description	Elastic File System (EFS) is a file storage service for Amazon EC2 Instances. As the name EFS implies, it is elastic. The storage capacity automatically increases or shrinks depending on the files. The pricing model for EFS is to pay only for the storage used by the file system. It has simple web services interfaces, which allow the users to create and configure the respective file systems easily and quickly. Using EFS, users can avoid the complexity of patching, deploying and maintaining the complex file systems, as it can simply manage the entire files storage infrastructure for them.
Objective	To create and mount EFS to a Linux instance
Services Involved	AWS EC2, AWS EFS
Prerequisites	Knowledge in EFS and basics of Shell Scripting
Lab Steps	<ol style="list-style-type: none">1. Create a Linux Instance2. Go to Elastic File System Service Page.3. Create a new File System and leave all the settings to default.4. Select the Amazon EC2 Mount Instructions (from local VPC), once the EFS is created.5. SSH into the EC2 Linux Instance created6. Execute the Steps shown on clicking the Mount Instructions


12. Use Case Name: Configuring a Static Website with S3 and CloudFront

Brief Description	Using AWS, users can deploy their static sites into an Amazon S3 bucket and host them for almost no cost. Besides, users can make the websites incredibly fast by using CloudFront to proxy and cache web traffic.
Objective	To host static website using the Amazon Simple Storage Service (S3)
Services Involved	Amazon S3 and CloudFront
Prerequisites	Basic knowledge on S3 buckets and CloudFront distributions
Lab Steps	<ol style="list-style-type: none"> 1. Create an S3 bucket and a Host Website: <ol style="list-style-type: none"> a. Log in to console. b. Open S3 and create bucket. c. Open bucket and select Permissions tab > Edit button. d. Uncheck all the options to make bucket publicly accessible. e. Choose Static Web-Site Hosting Tile from Properties Tab. f. Click Use this bucket to host a website and upload the required html files and save. g. Edit Bucket Policy in Permissions Tab and add the below policy: <pre> { "Version": "2012-10-17", "Statement": [{ "Sid": "AddPerm", "Effect": "Allow", "Principal": "*", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::YOUR_BUCKET_NAME/*" }] }</pre> <p>Note: Replace 'YOUR_BUCKET_NAME' with the created bucket name.</p> h. Find buckets endpoint from static website hosting option under Properties Tab. i. Copy and paste the endpoint in web browser, to see Static website hosted.

- 
2. Create a CloudFront Distribution for website:
 - a. Open CloudFront and create distribution.
Note: If you are not able to list your bucket name, enter **<bucketname.s3.amazonaws.com>**
 - b. Provide Origin Domain Name - Endpoint of Website (S3 bucket endpoint).
 - c. Leave other settings to default and create Distribution. (Takes about 15 minutes).
 - d. Click the distribution created under distributions and copy the domain name and paste it in browser with **/index.html** at the end of the domain name.
 - e. Try viewing the same website from different web browsers.
 - f. Find Viewers Tab in the left pane of distribution and use it to view graphically where are users of the website coming from, what OS, what devices they are using.

13. Use Case Name: Using S3 Bucket Policies and Conditions to Restrict Specific Permissions

Brief Description	Anonymous users and accounts are restricted by default from accessing S3 unless they are given access through Simple Storage Service's (S3) bucket policy. Bucket policies are based around the use of JSON (JavaScript Object Notation) files. AWS makes it relatively easy to set basic permissions on its S3 buckets through the AWS Policy Generator that presents with a Policy JSON Document that user can copy and paste to achieve this lab exercise.
Objective	To create a bucket policy to restrict specific IP address and to deny any uploaded object that does not have server-side encryption
Services Involved	Amazon S3, AWS Policy Generator (If the user is unfamiliar with JSON-based policy language.)
Prerequisites	Knowledge on creating a Bucket in S3
Lab Steps	<ol style="list-style-type: none"> 1. Navigate to S3 service and create a S3 bucket and inside a bucket, choose bucket policy under Permissions section. 2. Attach a policy directly or generate it by using a Policy Generator. For this lab exercise, use Policy Generator. 3. Select policy type as S3 bucket policy. Select Effect as Allow to reject permissions and Principal as * (The Principal dictates the user, account or service that policy applies to. An asterisk is a wildcard to match all.) 4. Service by default stays Amazon S3. Scroll through the actions available and choose s3:GetObject and mention the bucket arn. 5. Click Add Statement and then Generate Policy when ready to proceed. It generates a bucket policy which will be used for the bucket created. 6. Copy and paste the bucket policy in respective bucket created. Note: Error message displays if user tries to access objects in the bucket as the Policy denies get objects permission for the bucket. 7. Now use another policy which allows to upload objects only if they are encrypted. 8. Create a new policy, follow the same steps (from steps 5 to 10) with a few changes, select actions as PutObject and under conditions choose Condition as StringNotEquals and Key as s3:x-amz-server-side-encryption and under Value enter AES256. 9. Error message displays if the user tries to upload an object inside the bucket. 10. Try uploading the file again with encryption enabled. Click Next and navigate to Set properties screen of the wizard.

- 
11. In the **Encryption** section, click **Amazon S3 master-key** radio button to enable encryption and under review page encryption is enabled.
 12. Click **Upload**. This time the file is uploaded to S3 bucket, because the bucket policy was not violated.

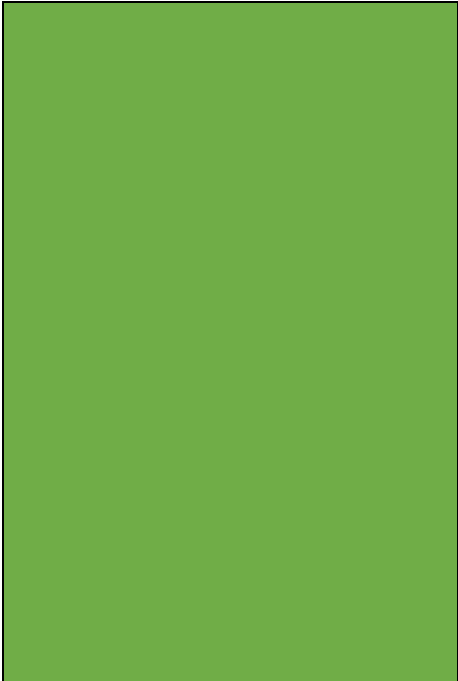
14. Use Case Name: Introduction to CloudWatch

Brief Description	<p>Amazon CloudWatch is a monitoring service for the resources and applications run on the AWS Cloud.</p> <p>It is used to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in AWS resources.</p>
Objective	To provide basics to get started in Amazon CloudWatch Service
Services Involved	AWS EC2, Amazon CloudWatch (Alarms)
Prerequisites	Familiarity with CloudWatch Service (creating alarms etc.,)
Lab Steps	<ol style="list-style-type: none"> 1. Create a Linux Instance (Enable Detailed Monitoring Option) 2. Find basic metrics for the EC2 instance under Monitoring Tab 3. Create an SNS Topic 4. Subscribe to the SNS Topic created with a protocol of email (confirm the subscription) 5. Follow the detailed steps in this Doc for installation of CloudWatch Monitoring Metric Scripts: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html 6. Navigate to CloudWatch Metrics Page. Select the name space called Linux System. 7. Play with the metric options. 8. Navigate to Alarms Section and create a New Alarm 9. Select HighCPUInstance metric in the Select metric page, under Per-instance metrics. 10. Configure Thresholds required for an alarm to trigger. 11. Select the created SNS Topic to get the mail once the alarm is triggered.

15. Use Case Name: Introduction to DynamoDB

Brief Description	<p>Amazon DynamoDB is a fully managed non-relational (NoSQL) database solution that provides fast performance at any scale.</p> <p>With Amazon DynamoDB, developers scaling cloud-based applications can start small with just the capacity they need and then increase the request capacity of a given table as their app grows in popularity. It stores data on Solid State Drives (SSDs) and replicates it synchronously across multiple AWS Availability Zones in an AWS Region to provide built-in high availability and data durability.</p>
Objective	<ul style="list-style-type: none"> • Create and configure DynamoDB tables • Add and edit rows in DynamoDB tables • Query information in DynamoDB
Services Involved	DynamoDB
Prerequisites	Basic Database Concepts
Lab Steps	<ol style="list-style-type: none"> 1. Create DynamoDB with Partition Key: <ol style="list-style-type: none"> a. Create DynamoDB Table with the below values: <ol style="list-style-type: none"> i. Table Name: Forum ii. Primary Key: Name (select String type from dropdown menu) iii. Uncheck 'Use default settings' b. Click Create button and the DynamoDB Table is created. 2. Create DynamoDB Table with Local and Global Secondary Indexes: <ol style="list-style-type: none"> a. Click Create table on the DynamoDB dashboard b. Fill out the form with the below values: <ol style="list-style-type: none"> i. Table Name: Thread ii. Primary Key: ForumName (select String type from dropdown menu) iii. Check 'Add sort key' iv. Sort Key: Subject (select String type from dropdown menu) c. Click +Add Index in the Secondary indexes to create a secondary index for the Thread table. d. Fill out the form with the below values and click Add index > Create to create the table: <ol style="list-style-type: none"> i. Primary Key: ForumName (select String type from dropdown menu) ii. Sort Key: CreationDate (select String type from dropdown menu)

- iii. Index Name: **Creationdate-index**
 - iv. Projected Attributes: **All Attributes**
 - e. Click **Create table** once more and fill out the form with the below values:
 - i. Table Name: **Reply**
 - ii. Primary Key: **ID** (select **String** type from dropdown menu)
 - iii. Sort Key: **CreationDate** (select **String** type from dropdown menu)
 - iv. Uncheck 'Use default settings'
 - f. Click **+Add Index** in the Secondary indexes section and fill out the form with the below values:
 - i. Primary Key: **ID** (select **String** type from dropdown menu)
 - ii. Sort Key: **Sticky** (select **String** type from dropdown menu)
 - iii. Index Name: **Sticky-index**
 - iv. Projected Attributes: **All Attributes**
 - v. Check 'Create as Local Secondary Index'
 - g. Click **+Add Index** once more and fill out the form with the below values:
 - i. Primary Key: **AuthorId** (select **String** type from dropdown menu)
 - ii. Sort Key: **CreationDate** (select **String** type from dropdown menu)
 - iii. Index Name: **AuthorId-CreationDate-index**
 - iv. Projected Attributes: **All Attributes**
 - v. Uncheck 'Create as Local Secondary Index'
 - h. Click **Create**.
- 3. Insert Items (demo data) into DynamoDB Table:
 - a. Select **Forum** from the created tables and click **Create Item** under **Items** Tab.
 - b. Add any forum **Name**, append **Description** and click **Save**.
 - c. Repeat steps a and b to insert 3 more different items (entries) in **Forum** table.
 - d. Select **Thread** from the created tables and click **Create Item** under **Items** Tab.
 - e. Provide any values for **CreationDate**, **ForumName**, **Subject** and click **Save**.
Note: **ForumName** value must match the name of any one of the forums.
 - f. Repeat steps d and e to insert 3 more different items (entries) in **Thread** table.

- 
4. Edit DynamoDB Table Items:
 - a. Select **Forum** table, and choose any Item created and click on its **Name** to get **Edit item** modal.
 - b. Click inside any value to make an update to its content and click **Save**.
 5. Query DynamoDB Table:

DynamoDB provides following two commands for searching data on the table: Scan (default) and Query.

 - a. Select **Thread** table and change **Scan** to **Query** under **Items** Tab.
 - b. Enter the name of a forum that has thread for in the **Partition Key** field and click **Start search** and find the results displayed.
 - c. Enter the value of an existing Subject one of the threads has in the **Sort key** field and click **Start search** to find a fewer result.

Note: The user has an option to choose different conditions for **Sort key** and can sort results.

16. Use Case Name: Creating your first Classic Load Balancer

Brief Description	Classic load balancer distributes incoming application traffic, based on either application or network level information, across multiple EC2 instances in multiple Availability Zones. Health checks can be configured in the load balancer so that the user can monitor the health of the attached EC2 instances and route the traffic only to healthy instances. This increases the fault tolerance of the applications and web traffic.
Objective	To create and use the Elastic Load Balancing (ELB) instance to balance the HTTP traffic between two EC2 instances. Also, to understand the Classic Load Balancer's behavior during an instance outage.
Services Involved	Amazon EC2
Prerequisites	Knowledge on launching EC2 instances
Lab Steps	<ol style="list-style-type: none"> 1. Create 2 Amazon EC2 Instances (Amazon Linux 2) 2. Install Apache 2 on each of them and edit the index.html file to display server1 in first instance and server2 in the second instance. 3. Create a Classic Load Balancer and register EC2 instances: <ol style="list-style-type: none"> a. Go to EC2 Console and click on Load Balancers from the left pane. b. Click on Create Load Balancer, select and create Classic Load Balancer from three available flavors of ELBs. A 7-step wizard opens to configure the specifics of Load Balancer (LB). c. In Step 1 of the multi-step wizard, specify a name for LB and choose a default VPC where the LB will live. Note: This VPC should be the same VPC where EC2 instances are running. d. Leave the checkbox Create an internal load balancer unchecked to make load balancer publicly accessible. e. Select Enable advanced VPC configuration to choose subnets. f. Click on the plus button (+) for all the subnets available in VPC to provide higher availability for load balancer. g. In Step 2, assign a new Security Group (SG) and provide a name and description for this SG. h. Allow connections coming from Anywhere (0.0.0.0/0) to the Port Range 80. i. Skip to Step 4 to configure a health check. Set HTTP as Ping Protocol, 80 as Port and (/index.html) as root path. Leave other advanced details to default settings.

- j. In **Step 5**, select instances called web-nodes to add EC2 instances to LB.
- k. Skip **Step 6** and click **Review and Create** button. Load balancer **classic-elb** is created.

4. Configure security groups for ELB:

- a. Go to EC2 Console and click on **Load Balancers** from the left pane and select the LB (classic-elb) created.
Note: Ensure both the instances are correctly attached to the LB created with the status **InService**. If yes, proceed with next step.
- b. Copy the Load Balancer DNS Name and paste into a new tab in browser. The page is generated and displays the EC2 instance ID.
Note: While copying DNS Name, do not include the "A Record" information in parentheses.
- c. Refresh the page to see this value changing to confirm both instances are in service.
- d. Go back to EC2 Console and click on **Load Balancers**.
- e. Select the LB created, go to **Description** tab.
- f. Scroll down to find **Security** and copy the **Source Security** – unique identifier of LB – to use in the **step i** below.
- g. Click on Security Groups from left pane and select the SG which has the Group Name - **cis-learners**
- h. Click on **Inbound** tab > **Edit** to change the current rules
- i. Select **Custom** in Source column and replace the source in HTTP rule with the already created unique identifier in **step f** above and click **Save**.
Tip: Type 'sg-' to select the SG identifier from the list
- j. Repeat the **step b** above to test this rule.
Note: Connection will timeout if user tries to copy and paste this public DNS address into another browser.

5. Check load balancer's behavior during instance failures:

- a. Go to EC2 Console and click on Load Balancers and select the LB (classic-elb) created.
Note: Ensure both the instances are currently on service.
- b. Click **Instances** from the left pane and choose any of the running instances and click on **Actions > Instance > State > Stop** to stimulate an instance failure.
Note: Ensure Instance State is changed to **stopped**. If yes, proceed with the next step.
- c. Click on **Load Balancers** to select the LB created and then go to **Instances** tab to check if the status of an instance changed to **OutOfService**.

	<ul style="list-style-type: none">d. Go to Descriptions tab and try to access the DNS Name of the LB to test the behaviour. Note: Hit the refresh button multiple times and it will always be forwarded to the same instance.e. Click Instances from the left pane and start the already stopped instance by clicking on Actions > Instance > State > Start. Note: Ensure Instance State is changed to running. If yes, proceed with the next step.f. Click on Load Balancers to select the LB created and then go to Instances tab to confirm if all the instances have an InService status.g. Test the LB's behaviour again by accessing its DNS Name in browser. <p>6. Monitor Classic Load Balancer:</p> <ul style="list-style-type: none">a. Go to EC2 Console and click on Load Balancers from the left pane and select the LB (classic-elb) created.b. Click on Monitoring tab to find the metrics of the LB selected. Note: The ELB Service, reports metrics to CloudWatch only when requests are flowing through the LB.
--	---

17. Use Case Name: Creating your first Auto Scaling group

Brief Description	A limited number of servers to cope with the application load can result in various failures and latency issues when the number of requests rises. The AWS Auto Scaling service automatically increases or decreases the Amazon EC2 compute capacity allocated for the cloud application, in response to changes in demand. Thus, it makes resources highly available which in turn eliminates various failure issues related to the increased load on limited servers.
Objective	To create an Auto Scaling Group and to ensure the availability of optimum number of functional instances.
Services Involved	AWS Auto Scaling, EC2
Prerequisites	Knowledge on launching EC2 instances
Lab Steps	<ol style="list-style-type: none">1. Create Launch Configuration:<ol style="list-style-type: none">a. Go to EC2 console and select Launch Configuration under AUTO SCALING in the left pane.b. Click on Create launch configuration.c. Launch an Amazon Linux 2 image for this lab session.d. Select As text radio button while giving User data input under Advanced Details.

- e. Use the below script as User Data to setup launch configuration:

```
#!/bin/bash

#Update packages
yum update -y
#Install Git and Apache
yum install -y git httpd

#Configure Apache to run by default on startup
chkconfig --levels 235 httpd on
#Cleaning the Welcome page config flie (Disable Apache default
welcome page on root)
echo "# Nothing in here" > /etc/httpd/conf.d/welcome.conf
#Clone the lab-util repo on the machine
git clone https://github.com/cloudacademy/lab-utils.git
#Enter in the lab-utils folder
cd lab-utils/
#Get the Instance ID from metadata
INSTANCEID=$(curl http://169.254.169.254/2016-06-30/meta-
data/instance-id)
#Get public DNS from metadata
PUBLICIP=$(curl http://169.254.169.254/2016-06-30/meta-
data/public-ipv4)
#Replace the Instance ID in the Html file
sed -i "s/instanceID/$INSTANCEID/g" html/legacy.html
#Replace the public DNS in the Html file
sed -i "s/publicIP/$PUBLICIP/g" html/legacy.html
#Copy the html file to the public html folder at the root level
cp -f html/legacy.html /var/www/html/index.html
#Restart apache
service httpd restart
```

- f. Under **IP Address Type**, select '**Assign a public IP Address to every instance**'
 - g. Assign a new Security Group (SG) and provide a name and description for this SG.
 - h. Select **HTTP** as the type of rule in the port **80** and select **My IP** from the Source field drop down menu.
 - i. Click **Create launch Configuration**.
 - j. Proceed without a key pair when prompted to select one and click on **Create Auto Scaling group using this launch configuration**.
2. Create an Auto Scaling Group:
 - a. Go to EC2 console and click on **Auto Scaling Groups** under **AUTO SCALING** in the left navigation pane.
 - b. Click on **Create Auto Scaling Group**
 - c. Select **Create an Auto Scaling group from an existing launch-configuration** and choose the launch configuration that is created.
 - d. Provide name to the Auto Scaling Group, group size (keep the size as 1), and subnets to use inside the default VPC.
 - e. Skip configuring the Advanced Details.

	<ul style="list-style-type: none">f. Select Keep this group at its initial size under Configure scaling policies.g. Select the type of notifications by creating an SNS Topic under Add notification.h. Skip configuring tags and click Create Auto Scaling Group button. Auto Scaling Group is created and wait for the status to be up and running. <p>3. Test ASG behaviour during failures:</p> <ul style="list-style-type: none">a. Open Auto Scaling Group just created.b. Select the running instance and terminate it to see that the Auto Scaling Group is up and running. Auto Scaling Group creates a new instance with same configuration on terminating the existing one. <p>Note: If personal email address is configured in the SNS topic, it receives an email notification to inform the termination of instance.</p>
--	--

18. Use Case Name: Create Your First Amazon RDS Database

Brief Description	Amazon Relational Database Service (RDS) forms a central part to set up a relational database in the cloud. The service provides very easy to use and manage common database administration tasks. Amazon RDS is highly cost-efficient, resizable with industry standard relational database. Any application in the Cloud that uses an SQL database such as MySQL, SQL Server, Oracle or PostgreSQL can use Amazon RDS as a scalable and reliable database.								
Objective	To launch a highly scalable relational database service in AWS Cloud								
Services Involved	Amazon RDS								
Prerequisites	Knowledge on basic Linux Commands								
Lab Steps	<ol style="list-style-type: none"> Create a Database Cluster Using RDS: <ol style="list-style-type: none"> Click Databases on the left pane followed by Create database. The wizard appears. Choose Standard Create method and select MySQL database engine. Choose a Free tier Template. Enter a Master username (login ID) and password under Credential Settings. Specify a unique Database Identifier name and click Create database. Under Connectivity Tab, click on Additional connectivity Configuration. Under VPC security group, select the Create new VPC Security Group and give a unique name. Under Additional Configuration, specify an Initial database name and click Create Database. Note: Creation of RDS Database Cluster takes up to 10 minutes for completion. Wait until its status becomes available. Launch an EC2 Linux instance, with new Security Group using the below rule and click Save. <table border="1" data-bbox="690 1501 1083 1650"> <tr> <td>Type</td><td>MySQL/Aurora</td></tr> <tr> <td>Protocol</td><td>TCP</td></tr> <tr> <td>Port Range</td><td>3306</td></tr> <tr> <td>Source</td><td>Anywhere</td></tr> </table> <ol style="list-style-type: none"> When the status of the instance is changed to running, choose Actions > Instance Settings > Attach/Replace IAM Role. From the IAM role dropdown menu, select GenC and click Apply. 	Type	MySQL/Aurora	Protocol	TCP	Port Range	3306	Source	Anywhere
Type	MySQL/Aurora								
Protocol	TCP								
Port Range	3306								
Source	Anywhere								

- b. Select the instance, click **Connect** and select **Session Manager** and click on **Connect**.
Note: Wait until connection to session manager opens.
3. Set up Security Group Rules for Connecting to the RDS Instance:
 - a. When the status of the database is changed to **Available**, click on the database created and under **Connectivity and security** section, select the Security Group which is created while launching the DB instance.
 - b. Click on **Inbound Rules** tab > **Edit rules**
 - c. Change the current rules using the below values and click **Save**.

Type	MySQL/Aurora
Protocol	TCP
Port Range	3306
Source	Custom (Security Group of the EC2 instance used to connect to the DB)

4. Connect to RDS and Create Database Table:
 - a. Key in the below command to change to the default Amazon Linux user (ec2-user) running in a bash shell, in the Session Manager shell session:
 - b. Install the MySQL client with the below command and this installs the necessary tools to interact with RDS instance:
 - c. Navigate to the **RDS > Databases** view and click **DB identifier** of the RDS instance created.
 - d. Copy the Endpoint displayed under **Endpoint & port** in the **Connectivity & security** section.
 - e. Run the below command replacing **your.endpoint.aws.com** with the endpoint copied in **step d** to connect to the database:

```
sudo -i -u ec2-user
```

```
sudo yum -y install mysql
```


```
mysql -h <your.endpoint.aws.com> -u <master username of the db given in step 1.e> -p
```

- f. Insert the Master RDS Password when prompted.
 - g. Open the Database already created, to start working:

```
USE <initial Database name(given in step 1.h)>
```

- h. Create a new table by executing the below command:

```
CREATE TABLE laboratory ( id INT,
name VARCHAR(100) );
```

- 
- i. Execute the below command to verify the creation of table and execute the second command to close database connection.

```
DESC laboratory;
```

```
quit;
```

19. Use Case Name: Managing Instance Volumes Using EBS

Brief Description	AWS EBS is largely defined as a raw block level storage service, designed to be used with Amazon EC2 instances. Each block acts like a hard drive, where any type of file can be stored. Each EBS volume attached with EC2 instance is automatically replicated within its availability zone to prevent data loss and component failure. With EBS, one can easily scale their usage up or down in a matter of few minutes.
Objective	<ul style="list-style-type: none"> • To create an EC2 instance with an additional EBS volume • To attach and detach an EBS to/from an EC2 instance and • To take an EBS Snapshot – an incremental copy of data
Services Involved	Amazon EBS, EC2
Prerequisites	Knowledge on launching EC2 instances and running Linux commands
Lab Steps	<ol style="list-style-type: none"> 1. Create an EC2 instance with multiple storage volumes: <ol style="list-style-type: none"> a. Go to EC2 console and click Launch Instance b. Select 64-bit Amazon Linux 2 AMI from the list of basic configurations displayed in Step 1: Choose an Amazon Machine Image (AMI) and skip to Step 4: Add Storage. c. Click on Add New Volume to add a second EBS to your EC2 instance. Note: Before creating a new volume, ensure Availability Zone is same as the one used by Operating System. d. Set the Volume 'Size' to 16 GiB, 'Volume Type' to General Purpose SSD (selected by default) with Delete on Termination unchecked and then click the Review and Launch button. Note: Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. e. Click Launch and create a new key pair. f. Click the Download Key Pair button to download the .pem file (privacy enhanced mail). g. Click Launch Instances. A confirmation page appears. h. Click View Instances and wait until Instance State is changed to running. i. Click on the Volumes menu item under ELASTIC BLOCK STORE in the left pane, to find the EBS volumes used by the newly created EC2 instance. j. Hover cursor near the Name cell of the desired volume and a small pencil icon will appear, click on it to give a name to the disk. 2. Create a new EBS volume:

- a. Ensure the **Availability Zone** is same for both the volumes in the **Volumes** list page and click **Create Volume**.
 - b. In the **Volumes > Create Volume** page, set Size to **10 GiB** from the default 100 and select the same **Availability Zone*** as the other volumes and click **Create Volume**.
Note: Remember to decrease the size of volume to 10 GiB otherwise it will exceed the storage limit imposed on this lab.
 - c. Click Close to the success notification and wait until volume status is available.
Tip: Refresh the page often view the change in volume status.
3. Attach and Detach an EBS volume
 - a. To attach an EBS volume:
 - i. Right-click on the available volume and select **Attach Volume**.
 - ii. Select **EC2 Instance** and assign a **Device** for the drive and click **Attach** in the **Attach Volume** page.
 - b. To detach an EBS volume:
 - i. Select the volume, then click **Detach Volume** and click **Yes, Detach** in the confirmation dialog box.
Note: If the attached instance is running, unmount the volume from the instance before detaching it. If an Amazon EBS volume is the root device of an instance, stop the instance before detaching volume. Detach an Amazon EBS volume by terminating the instance.
 4. Connect to a Remote Shell Using an SSH Connection:
 - a. Convert private key (.pem file) - downloaded in Step 1 of this document - into this format (.ppk file) using PuTTYgen and save it as private key.
 - b. Copy the Public IP address of the EC2 instance created.
 - c. Start PuTTY and paste the IP address in **Host Name** field in the following format: **ec2-user@ipv6 address**.
 - d. Select **Connection > SSH > Auth** and then browse for the downloaded .ppk Keypair, then click **Open**. An authentication form opens.
 - e. Log in as **ec2-user** and the EC2 server welcome banner is displayed.
 5. Create a filesystem on an EBS Volume:
 - a. Use the below Linux command to list all disks:

```
lsblk
```

Note: Take note of the name of the device/disk to mount.

- b. To create an ext4 filesystem on the new volume, issue the below command:

```
sudo mkfs -t ext4  
/dev/NAME OF YOUR DEVICE
```

- c. Create the directory to mount the new volume using the below command:

```
sudo mkdir /mnt/ebs-store
```

- d. Mount the new volume using the below command:

```
sudo mount /dev/NAME_OF_YOUR_DEVICE  
/mnt/ebs-store
```

- e. To configure the Linux instance to mount this volume on boot, open `/etc/fstab` file in an editor with the below command:

```
sudo nano /etc/fstab
```

- f. Append the below command line to the end of `/etc/fstab`:

```
/dev/NAME_OF_YOUR_DEVICE /mnt/ebs-store  
ext4 defaults,noatime 1 2
```

- g. In the text editor, hit `Ctrl+O` and then enter to save, then `Ctrl+X` to exit the editor.

- h. Type again the below command to find the device mounted to `/mnt/ebs-store`:

```
lsblk
```

6. Create an EBS snapshot:

- a. Unmount the volume in Linux using the below command:

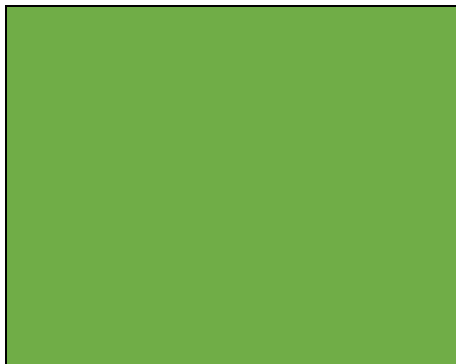
```
sudo umount -d /mnt/ebs-store/
```

- b. Click **Snapshots** in the left navigation pane and click **Create Snapshot**.

- c. Select the desired volume and click **Create Snapshot** in the **Create Snapshot** page.

- d. Click **Close** to the success notification.

- e. Locate the desired snapshot from Snapshots Console, right-click and select **Create Volume** to restore data from the snapshot.



- f. Select the desired **Volume Type, Size, Availability Zone** and click **Create Volume**.
- g. Click **Close** to the success notification.
Note: Once data is restored to a new volume, user can attach it to an instance and mount the storage as per the previous steps. Since the new volume has previous data, do not create a new file system (mkfs in Linux). Simply mount the volume (Linux) and start using existing file system and data.

20. Use Case Name: Working with Amazon EC2 Auto Scaling Groups and Classic ELB

Brief Description	<p>A load balancer distributes incoming application traffic across multiple EC2 instances in varied Availability Zones. This makes resources highly available which in turn eliminates various failure issues related to the increased load on limited servers.</p> <p>Elastic Load Balancing detects unhealthy instances and routes traffic only to healthy instances. The AWS Auto Scaling service automatically increases or decreases the Amazon EC2 compute capacity allocated for the cloud application, in response to changes in demand.</p>
Objective	To create and configure Auto-Scaling groups, Classic ELB and triggering notifications when scaling events occur based on CPU utilization
Services Involved	Auto Scaling, CloudWatch, EC2, ELB, SNS
Prerequisites	<ul style="list-style-type: none"> • Knowledge on launching EC2 instances and connecting to them using SSH • Understanding of CloudWatch, SNS and EC2 Security Group concepts
Lab Steps	<ol style="list-style-type: none"> 1. Create a Load Balancer using ELB: <ol style="list-style-type: none"> a. Go to EC2 Console and click on Load Balancers from the left pane under LOAD BALANCING section. b. Click on Create Load Balancer, select and create Classic Load Balancer from three available flavors of ELBs. A 7-step wizard opens to configure the specifics of Load Balancer (LB). c. In Step 1 of the multi-step wizard, specify a name (Web) for LB and choose the default VPC where the LB will live. d. Leave the checkbox Create an internal load balancer unchecked to make load balancer publicly accessible. e. Select the checkbox Enable advanced VPC configuration to choose subnets. f. Leave the default value (HTTP/80/HTTP/80) for Listener Configuration. g. Click on the plus button (+) under Select Subnets > Available subnets, for all the subnets available in VPC to provide higher availability for load balancer. Note: Select us-east-2 (Ohio) and its availability zones. h. In Step 2, while assigning a new Security Group (SG), enter elb-webserver for Security group name and ELB for a webserver cluster for Description. i. Allow connections coming from Anywhere (0.0.0.0/0) to the Port Range 80. j. Skip to Step 4 to configure a health check. Set TCP as Ping Protocol and 80 as Ping Port. Leave other advanced details to default settings.

Note: In **Step 5: Add EC2 Instances**, the message '*No instances available*' appears as no Auto Scaling Group is created and launched yet.

- k. Skip **Step 6** and click **Review and Create** button. Load balancer is created.

2. Create a Launch Configuration:

- a. Go to EC2 console and select **Launch Configuration** under **AUTO SCALING** in the left pane.

Note: User can attach only one Launch Configuration to an Auto Scaling group at a time and it cannot be modified.

- b. Click on **Create launch configuration**.
- c. Launch an **Amazon Linux 2** image for this lab session.
- d. On the **Configuration details** page:
 - i. Enter **webserver-cluster** for the **Name** and
 - ii. Check **Enable CloudWatch detailed monitoring**
- e. Expand **Advanced Details**, select **As text** radio button and paste the below script as **User data** to setup launch configuration:

```
#!/bin/bash
#Enable the epel-release
amazon-linux-extras install epel
#Install and start Apache web
server
yum install -y httpd php
service httpd start
#Install CPU stress test tool
sudo yum install -y stress
```

- f. Under **IP Address Type**, select '**Assign a public IP Address to every instance**'
 - g. Leave the default settings on the **Add Storage** page.
 - h. On the **Configure Security Group** page, select the radio button **Create a new security group** and enter **Webserver-cluster** for Security group name and enter description.
 - i. Add a second rule in addition to the one (SSH Type) configured by default. Set **HTTP** as Type, **TCP** as Protocol, **80** as Port Range and **Anywhere (0.0.0.0/0)** as Source.
 - j. Click **Review > Create launch configuration** and create a new key pair when prompted.
 - k. Click the **Download Key Pair** button to download the .pem file (privacy enhanced mail).
3. Create an Auto Scaling Group:
- a. Go to EC2 console and click on **Auto Scaling Groups** under **AUTO SCALING** in the left pane.
 - b. Click on **Create Auto Scaling Group** and select **Use an existing launch configuration** radio button and choose the launch

configuration (webserver-cluster) that is created. A 5-step wizard opens.

- c. In **Step 1: Configure Auto Scaling group details**, provide the below details:
 - i. Group name: **webserver-cluster**
 - ii. Group size: Start with **1** instance
 - iii. Network: Select the **vpc-<Alphanumeric>** from the dropdown menu
 - iv. Subnet: Select two subnets – one in **us-east-2a** and the other in **us-east-2b**.
 - v. Expand **Advanced Details** and provide the below details:
 - a) Load Balancing: Check **Receive traffic from one or more load balancers**
 - b) Select **Web** (the ELB created earlier) for the Classic Load Balancer.
 - c) Health Check Type: **ELB**
 - d) Monitoring: Check **Enable CloudWatch detailed monitoring**
 - vi. Leave the rest of the default options and click **Configuring scaling policies**.
- d. In **Step 2**, select **Use scaling policies to adjust the capacity of this group** radio button.
- e. Set it to scale between **1** and **5** instances, then select **Scale the Auto Scaling group using step or simple scaling policies**.
- f. Click **Add new alarm** under **Increase Group Size** section and provide the below details in the **Create Alarm** dialogue.

Note: Set up a Simple Notification Service (SNS) topic to receive a notification when the alarm is triggered.

 - i. Check the **Send a notification** checkbox
 - ii. Click **create** topic. Enter **autoscaling-alarm-up** for the SNS topic name and then enter the valid email address in **With these recipients** field.
 - iii. Set **Whenever** to **Average** and **CPU Utilization** as the type of metric
 - iv. Set the relationship to **>=** and enter **80** as the **Percent**
 - v. Set **For at least** to **1** consecutive period of **5 Minutes** and click **Create Alarm**.
- g. Click **Add new alarm** under **Decrease Group Size** section, the **Create Alarm** dialogue is displayed. Follow the same steps above-mentioned, except for the below two:
 - i. Enter **autoscaling-alarm-down** for the SNS topic name
 - ii. Set the relationship to **<=** and enter **10** as the **Percent** and click **Create Alarm**.

- h. In **Step 3: Configure Notifications**, click **Add notification** to get a notice whenever an Auto Scaling Group instance is launched or terminated, with or without success.
 - i. Select **launch** and **fail to launch** checkboxes for the topic name **autoscaling-alarm-up**.
 - ii. Click **Add notification** to add a second notification.
 - iii. Select **terminate** and **fail to terminate** checkboxes for the topic name **autoscaling-alarm-down**.
 - i. Skip **Step 4: Configure Tags** and click **Review > Create Auto Scaling group**.
 - j. Go to **EC2 Dashboard > Auto Scaling Groups**. The cluster gets deployed and EC2 instance is ready to use.
 - k. Open **Load Balancers** section, select the previously created ELB (Web), and then open **Instances** tab. The new Auto Scaling instance is automatically added to the ELB configuration.
4. Test EC2 Auto Scaling Groups from End to End:
- Confirm an instance is running and settles at the desired number of one. User can do this from checking either the **EC2 Dashboard > Running Instances** or **Auto Scaling Group > Instances** tab.
 - Check the following email received in the mailing address entered for SNS notifications earlier: **AWS Notification – Subscription Confirmation** email from **AWS Notifications** for both **autoscaling-alarm-up** and **autoscaling-alarm-down** SNS topics. Click the **confirm subscription** link in the email.
 - Enter either the **Public DNS** or **Public IP** address of the running instance in a new browser window. The **Amazon Linux AMI Test Page** displays from the Apache web server installed and started via **User data** in the Launch Configuration.
 - Go to **EC2 > Load Balancers**, then select **Instances** tab for Load Balancer. Confirm the running instance is registered with the Load Balancer and the **Status** is **InService**.
 - Go to **EC2 > Load Balancers** page, copy the **DNS name** and paste it into a new browser window. The default **Amazon Linux AMI Test Page** displays.
Note: This request goes through the ELB to the instance.
 - Go to **EC2 > Running Instances**, select **Actions > Instance State > Terminate** on the running instance. It relaunches automatically. Let the new instance launch and settle into a running state.
 - SSH into the running instance and stress test it:
 - a) Connecting from a local Windows host to a running Linux host requires a PPK file. Use PuTTYgen to convert a PEM file to a PPK and then connect via PuTTY.
 - b) Install stress by using the yum package manager given below:

```

$ sudo yum install stress
Loaded plugins: priorities, update-motd, upgrade-helper
amzn-main | 2.1
kB      00:00
amzn-updates | 2.3
kB      00:00
Resolving Dependencies
--> Running transaction check
---> Package stress.x86_64 0:1.0.4-4.2.amzn1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
=====
Package           Arch           Version           Repository
Size
=====
=====
Installing:
  stress           x86_64         1.0.4-4.2.amzn1   amzn-
main              38 k

Transaction Summary
=====
=====
Install 1 Package

Total download size: 38 k
Installed size: 89 k
Is this ok [y/d/N]: y
Downloading packages:
stress-1.0.4-4.2.amzn1.x86_64.rpm | 38
kB      00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : stress-1.0.4-
4.2.amzn1.x86_64                                1/1
  Verifying  : stress-1.0.4-
4.2.amzn1.x86_64                                1/1

Installed:
  stress.x86_64 0:1.0.4-4.2.amzn1

```


- c) Enter the below command to confirm the stress executable is in the path. (/usr/bin/stress)

```
which stress
```

- d) Run stress to eat up CPU cycles for five minutes (-t 5m option):

```
$ stress -c 2 -i 1 -m 1 --vm-bytes 128M -t 5m
stress: info: [23710] dispatching hogs:
2 cpu, 1 io, 1 vm, 0 hdd
stress: info: [23710] successful run
```

- Go to **EC2 Dashboard > Running Instances > Monitoring** tab. Look at the CPU Utilization.
Tip: Click the refresh icon every 30 seconds. The horizontal line in the graph quickly ramps up from nearly 0% to 100% within a few minutes.
- Check for a new instance that automatically starts when the CPU climbs on the running instance over the next 5+ minutes.
- Further, it registers with ELB and pass health checks with **InService** status.
- Wait for another 5 minutes. As the stress test runs for only 5 minutes, CPU utilization spreads across both instances in the load balancer pool and quickly goes back down to near zero. Since the scale-down policy relationship is ≤ 10 , one of the two **Instance State** transitions from shutting-down to terminated automatically.
- The email address configured in the SNS topic receives an email from **AWS Notifications captioned: ALARM: "awsec2-webserver-cluster-Low-CPU-Utilization" in US East – Ohio**, as a CloudWatch Alarm is raised.
Note: Eventually, user can find only one running instance, and two terminated instances (one terminated manually, and another terminated based on the alarm triggered when the CPU dropped below 10% for 5 minutes)
- Go to **Load Balancers > Monitoring** tab and ensure there is no Unhealthy Hosts, and find the Healthy Hosts scale up from 1 to 2 and back down to 1.