ELB AND AUTOSCALING

Attempt	1
Marks Obtained	6 / 10
Your score is	60%
Completed on	Sunday , 13 January 2019 , 12:51 PM
Time Taken	00 H 06 M 00 S
Result	Fail

Domains / Topics wise Quiz Performance Report

S.No.	Topic	Total Questions	Correct	Incorrect	Unattempted
1	Other	10	6	4	0

10 Questions	**6** Correct	**4** Incorrect	**0** Unattempted

Show Answers

All

QUESTION 1 CORRECT

Which of the following are components required to effectively setup AutoScaling on your EC2 instances for a web-based application? Choose 3 correct Options:

- [x] **A.** Launch Configuration ✔
- [x] **B.** Elastic Load Balancer ✔
- [ ] **C.** Lambda
- [x] **D.** AutoScaling Group ✔
- [ ] **E.** Elastic IP

Explanation :

Answer: A,B, D

Option A is correct.

A launch configuration specifies the type of EC2 instance that Amazon EC2 Auto Scaling creates for you. You create the launch configuration by including information such as the ID of the Amazon Machine Image (AMI) to use, the instance type, the key pair, security groups, and block device mapping.

Launch Configurations

A *launch configuration* is a template that an Auto Scaling group uses to launch EC2 instances. When you create a launch configuration, you specify information for the instances such as the ID of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping. If you've launched an EC2 instance before, you specified the same information in order to launch the instance.

You can specify your launch configuration with multiple Auto Scaling groups. However, you can only specify one launch configuration for an Auto Scaling group at a time, and you can't modify a launch configuration after you've created it. Therefore, if you want to change the launch configuration for an Auto Scaling group, you must create a launch configuration and then update your Auto Scaling group with the new launch configuration.

Keep in mind that whenever you create an Auto Scaling group, you must specify a launch configuration, a launch template, or an EC2 instance. When you create an Auto Scaling group using an EC2 instance, Amazon EC2 Auto Scaling automatically creates a launch configuration for you and associates it with the Auto Scaling group. For more information, see Creating an Auto Scaling Group Using an EC2 Instance. Alternatively, if you create a launch template, you can use your launch template to create an Auto Scaling group instead of creating a launch configuration. For more information, see Creating an Auto Scaling Group Using a Launch Template.

- https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchConfiguration.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchConfiguration.html)

Option B is correct.
You can attach a load balancer to your Auto Scaling group. The load balancer automatically distributes incoming traffic across the instances in the group.

Using a Load Balancer With an Auto Scaling Group

You can automatically increase the size of your Auto Scaling group when demand goes up and decrease it when demand goes down. As the Auto Scaling group adds and removes EC2 instances, you must ensure that the traffic for your application is distributed across all of your EC2 instances. The Elastic Load Balancing service automatically routes incoming web traffic across such a dynamically changing number of EC2 instances. Your load balancer acts as a single point of contact for all incoming traffic to the instances in your Auto Scaling group. For more information, see the Elastic Load Balancing User Guide.

To use a load balancer with your Auto Scaling group, create the load balancer and then attach it to the group.

Contents

- Attaching a Load Balancer to Your Auto Scaling Group
- Using Elastic Load Balancing Health Checks with Auto Scaling
- Expanding Your Scaled and Load-Balanced Application to an Additional Availability Zone

https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/autoscaling-load-balancer.html)

Option C is not correct.
Lambda functions are not required to setup auto scaling for EC2 instances.

Option D is correct.

An Auto Scaling group is a collection of EC2 instances, and the core of Amazon EC2 Auto Scaling. When you create an Auto Scaling group, you include information such as the subnets for the instances and the number of instances the group must maintain at all times.

Auto Scaling Groups

An *Auto Scaling group* contains a collection of EC2 instances that share similar characteristics and are treated as a logical grouping for the purposes of instance scaling and management. For example, if a single application operates across multiple instances, you might want to increase the number of instances in that group to improve the performance of the application, or decrease the number of instances to reduce costs when demand is low. You can use the Auto Scaling group to scale the number of instances automatically based on criteria that you specify, or maintain a fixed number of instances even if an instance becomes unhealthy. This automatic scaling and maintaining the number of instances in an Auto Scaling group is the core functionality of the Amazon EC2 Auto Scaling service.

An Auto Scaling group starts by launching enough EC2 instances to meet its desired capacity. The Auto Scaling group maintains this number of instances by performing periodic health checks on the instances in the group. If an instance becomes unhealthy, the group terminates the unhealthy instance and launches another instance to replace it. For more information about health check replacements, see Maintaining the Number of Instances in Your Auto Scaling Group.

You can use scaling policies to increase or decrease the number of running EC2 instances in your group dynamically to meet changing conditions. When the scaling policy is in effect, the Auto Scaling group adjusts the desired capacity of the group and launches or terminates the instances as needed. If you manually scale or scale on a schedule, you must adjust the desired capacity of the group in order for the changes to take effect. For more information, see Scaling the Size of Your Auto Scaling Group.

Before you get started, take the time to review your application thoroughly as it runs in the AWS Cloud. Take note of the following:

- How long it takes to launch and configure a server.
- What metrics have the most relevance to your application's performance.
- How many Availability Zones you want the Auto Scaling group to span.
- Do you want to scale to increase or decrease capacity? Do you just want to ensure that a specific number of servers are always running? (Keep in mind that Amazon EC2 Auto Scaling can do both simultaneously.)
- What existing resources (such as EC2 instances or AMIs) you can use.

The better you understand your application, the more effective you can make your Auto Scaling architecture.

https://docs.aws.amazon.com/autoscaling/ec2/userguide/AutoScalingGroup.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/AutoScalingGroup.html)

Option E is not correct.
Elastic IP is not required to setup auto scaling for EC2 instances.

Ask our Experts

QUESTION 2 CORRECT

Your organization had setup Auto Scaling for an EC2 instance. They intend to launch one additional new instance with same configuration automatically when the workload increases and shut it down automatically when the workload is back to normal. However, for security reasons, they have applied operating system patches to the main instance and would like this to be reflected when Auto Scaling group launches new EC2 instance. What would happen in this scenario?

- **A.** Auto Scaling group will launch new EC2 instance from the main instance latest snapshot. New instance will have updated patches.
- **B.** Create an image out of main EC2 instance and update Auto Scaling group configuration with new image AMI ID.
- **C.** Create an image out of main EC2 instance and update Launch Configuration with new image AMI ID.
- **D.** Create an image out of main EC2 instance, create a new Launch Configuration with new image AMI ID, update Auto Scaling group with new Launch Configuration ID ✔

Explanation :

Answer: D

Option A is not correct.

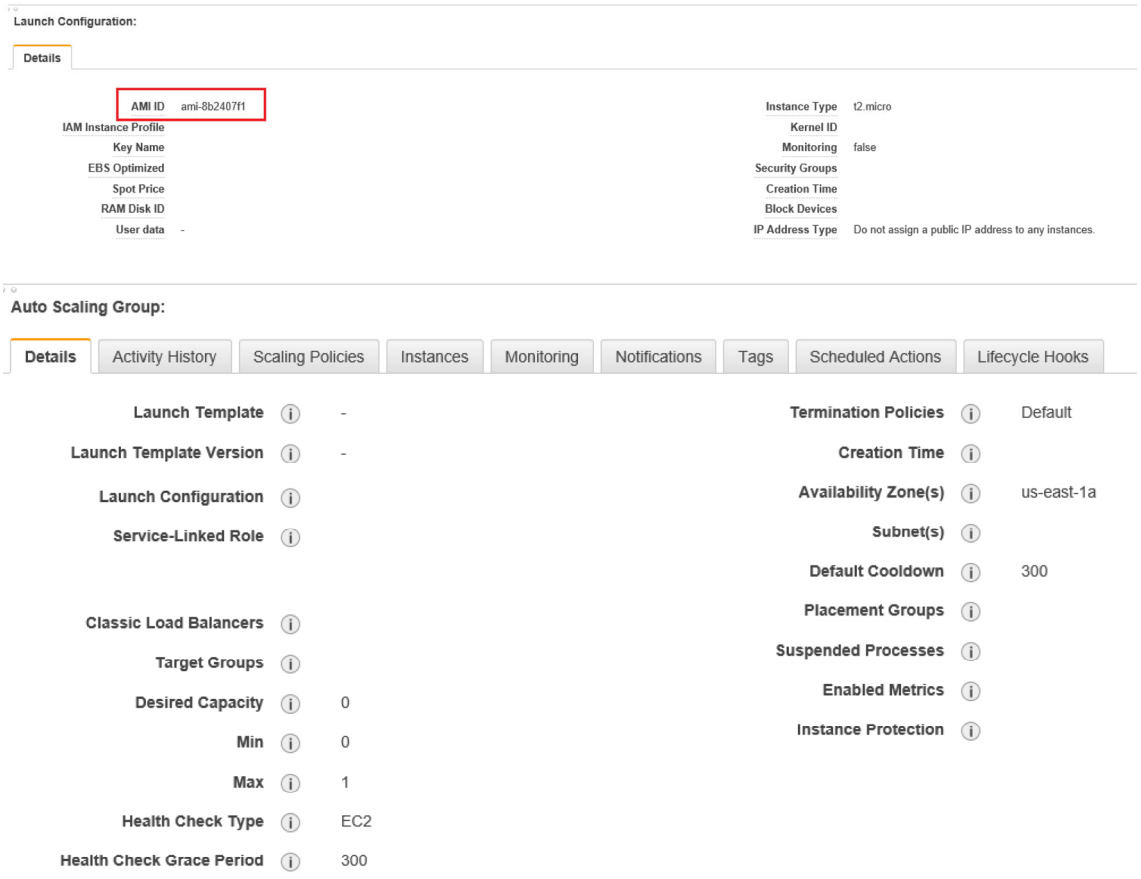
Auto Scaling group launches new instances based on the configuration defined in Launch Configuration. AMI ID is one of the configuration parameter which defines the type of instance to be launched when auto scaling logic is executed.

AMI ID is set during the creation of launch configuration and cannot be modified.

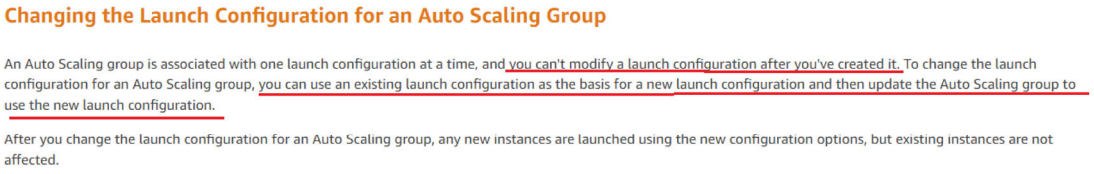
So, auto scaling group will not launch new instance based on the latest image of main instance.

Option B is not correct.

AMI ID is a configuration on Launch Configuration, not Auto Scaling Group.



Option C is not correct and Option D is correct.



https://docs.aws.amazon.com/autoscaling/ec2/userguide/change-launch-config.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/change-launch-config.html)

Ask our Experts

QUESTION 3 CORRECT

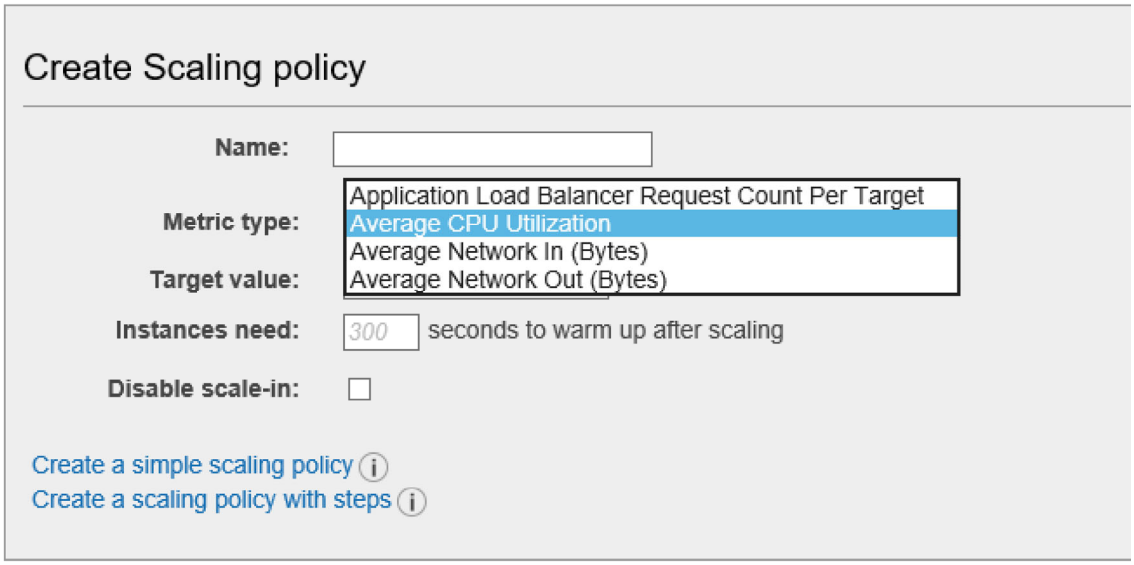
Which of the following is not a default metric type for Auto Scaling Group policy?

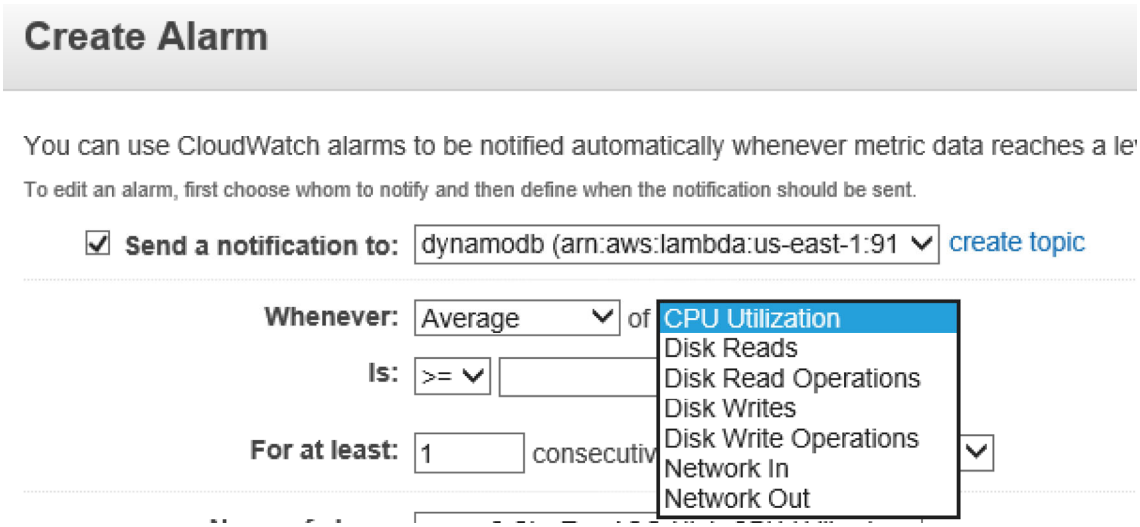
- A. Average CPU Utilization
- B. Memory Utilization ✔
- C. Network In
- D. Network Out

Explanation :

Answer: B

Following are the default metric types available for Simple Policy and Step Policy





- https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-simple-step.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-simple-step.html)

Ask our Experts

QUESTION 4 INCORRECT

An application team in your organization set up an autoscaling group configuration (with simple scaling policy) to launch a new instance when CPU utilization reaches 85%. However, at times, when EC2 instance comes into in-service, it reports Unhealthy status immediately. However, as the replacement of the unhealthy instance by another instance (launched by Autoscaling group) takes more than 15 minutes, the unhealthy instance has to be removed manually. What do you think is the reason behind this?

- A. Auto scaling policy alarm incorrectly configured.
- B. Health Check Grace Period set to 20 minutes. ✔

C. Termination policy set to Do Not Terminate instances.

D. Launch Configuration is not configured to report Unhealthy status ✖

Explanation :

Answer: B

Option A is not correct.

Instance health status is not determined by the cloudwatch alarms.

Instance Health Status

Amazon EC2 Auto Scaling determines the health status of an instance using one or more of the following:

- Status checks provided by Amazon EC2 (systems status checks and instance status checks. For more information, see Status Checks for Your Instances in the *Amazon EC2 User Guide for Linux Instances*.
- Health checks provided by Elastic Load Balancing. For more information, see Health Checks for Your Target Groups in the *User Guide for Application Load Balancers* or Configure Health Checks for Your Classic Load Balancer in the *User Guide for Classic Load Balancers*.
- Custom health checks.

By default, Amazon EC2 Auto Scaling health checks use the results of the EC2 status checks to determine the health status of an instance. Amazon EC2 Auto Scaling marks an instance as unhealthy if its instance fails one or more of the status checks.

Option B is correct.

Health Check Grace Period

Frequently, an Auto Scaling instance that has just come into service needs to warm up before it can pass the health check. Amazon EC2 Auto Scaling waits until the health check grace period ends before checking the health status of the instance. While the EC2 status checks and ELB health checks can complete before the health check grace period expires, Amazon EC2 Auto Scaling does not act on them until the health check grace period expires. To provide ample warm-up time for your instances, ensure that the health check grace period covers the expected startup time for your application. Note that if you add a lifecycle hook to perform actions as your instances launch, the health check grace period does not start until the lifecycle hook is completed and the instance enters the `InService` state.

Option C is not correct.

Termination policy does not have “Do Not Terminate” option.

Amazon EC2 Auto Scaling supports the following custom termination policies:

- `OldestInstance`. Terminate the oldest instance in the group. This option is useful when you're upgrading the instances in the Auto Scaling group to a new EC2 instance type. You can gradually replace instances of the old type with instances of the new type.
- `NewestInstance`. Terminate the newest instance in the group. This policy is useful when you're testing a new launch configuration but don't want to keep it in production.
- `OldestLaunchConfiguration`. Terminate instances that have the oldest launch configuration. This policy is useful when you're updating a group and phasing out the instances from a previous configuration.
- `ClosestToNextInstanceHour`. Terminate instances that are closest to the next billing hour. This policy helps you maximize the use of your instances and manage your Amazon EC2 usage costs.
- `Default`. Terminate instances according to the default termination policy. This policy is useful when you have more than one scaling policy for the group.

Option D is not a correct statement.

Ask our Experts

QUESTION 5 INCORRECT

In an auto scaling group setup, with default termination policy, which statement is correct?

- **A.** Select the Availability Zone with the most instances and at least one instance that is not protected from scale in. If there is more than one Availability Zone with this number of instances, select the Availability Zone with the instances that use the oldest launch configuration ✔
- **B.** Determine which unprotected instances in the selected Availability Zone use the newest launch configuration. If there is one such instance, terminate it.

- **C.** If there are multiple instances that use the oldest launch configuration, determine which unprotected instances are closest to the previous billing hour and terminate it. ✖
- **D.** If there is more than one unprotected instance closest to the previous billing hour, select one of these instances at random.

Explanation :

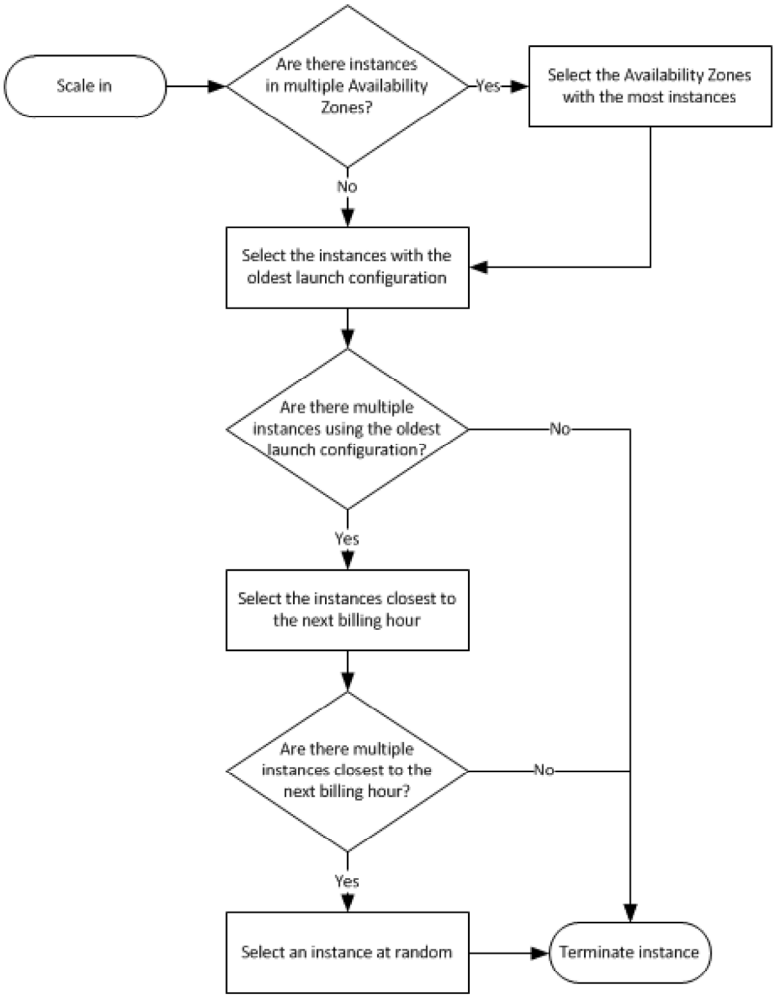
Answer: A

Default Termination Policy

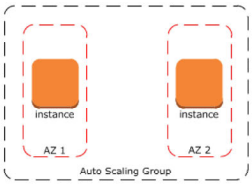
The default termination policy is designed to help ensure that your network architecture spans Availability Zones evenly. With the default termination policy, the behavior of the Auto Scaling group is as follows:

1. If there are instances in multiple Availability Zones, select the Availability Zone with the most instances and at least one instance that is not protected from scale in. If there is more than one Availability Zone with this number of instances, select the Availability Zone with the instances that use the oldest launch configuration.
2. Determine which unprotected instances in the selected Availability Zone use the oldest launch configuration. If there is one such instance, terminate it.
3. If there are multiple instances that use the oldest launch configuration, determine which unprotected instances are closest to the next billing hour. (This helps you maximize the use of your EC2 instances and manage your Amazon EC2 usage costs.) If there is one such instance, terminate it.
4. If there is more than one unprotected instance closest to the next billing hour, select one of these instances at random.

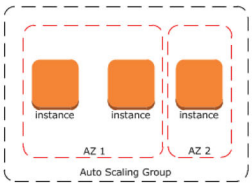
The following flow diagram illustrates how the default termination policy works.



Consider an Auto Scaling group that has two Availability Zones, a desired capacity of two instances, and scaling policies that increase and decrease the number of instances by 1 when certain thresholds are met. The two instances in this group are distributed as follows.



When the threshold for the scale-out policy is met, the policy takes effect and the Auto Scaling group launches a new instance. The Auto Scaling group now has three instances, distributed as follows.



When the threshold for the scale-in policy is met, the policy takes effect and the Auto Scaling group terminates one of the instances. If you did not assign a specific termination policy to the group, it uses the default termination policy. It selects the Availability Zone with two instances, and terminates the instance launched from the oldest launch configuration. If the instances were launched from the same launch configuration, then the Auto Scaling group selects the instance that is closest to the next billing hour and terminates it.

- https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html (https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html)

Ask our Experts

QUESTION 6 CORRECT

You had setup an internal HTTP(S) Elastic Load Balancer to route requests to two EC2 instances inside a private VPC. However, one of the target EC2 instance is showing Unhealthy status. Which of the following options could not be a reason this?

- **A.** Port 80/443 is not allowed on EC2 instance's Security Group from load balancer.
- **B.** EC2 instance is in different availability zones than load balancer. ✔
- **C.** The ping path does not exist on the EC2 instance.
- **D.** The target did not return a successful response code

Explanation :

Answer: B
If a target is taking longer than expected to enter the InService state, it might be failing health checks. Your target is not in service until it passes one health check

Target Health Status

Before the load balancer sends a health check request to a target, you must register it with a target group, specify its target group in a listener rule, and ensure that the Availability Zone of the target is enabled for the load balancer. Before a target can receive requests from the load balancer, it must pass the initial health checks. After a target passes the initial health checks, its status is `Healthy`.

The following table describes the possible values for the health status of a registered target.

Value	Description
`initial`	The load balancer is in the process of registering the target or performing the initial health checks on the target.
`healthy`	The target is healthy.
`unhealthy`	The target did not respond to a health check or failed the health check.
`unused`	The target is not registered with a target group, the target group is not used in a listener rule for the load balancer, or the target is in an Availability Zone that is not enabled for the load balancer.
`draining`	The target is deregistering and connection draining is in process.

A security group does not allow traffic

The security group associated with an instance must allow traffic from the load balancer using the health check port and health check protocol. You can add a rule to the instance security group to allow all traffic from the load balancer security group. Also, the security group for your load balancer must allow traffic to the instances.

A network access control list (ACL) does not allow traffic

The network ACL associated with the subnets for your instances must allow inbound traffic on the health check port and outbound traffic on the ephemeral ports (1024-65535). The network ACL associated with the subnets for your load balancer nodes must allow inbound traffic on the ephemeral ports and outbound traffic on the health check and ephemeral ports.

The ping path does not exist

Create a target page for the health check and specify its path as the ping path.

The connection times out

First, verify that you can connect to the target directly from within the network using the private IP address of the target and the health check protocol. If you can't connect, check whether the instance is over-utilized, and add more targets to your target group if it is too busy to respond. If you can connect, it is possible that the target page is not responding before the health check timeout period. Choose a simpler target page for the health check or adjust the health check settings.

The target did not return a successful response code

By default, the success code is 200, but you can optionally specify additional success codes when you configure health checks. Confirm the success codes that the load balancer is expecting and that your application is configured to return these codes on success.

- https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-troubleshooting.html#target-not-inservice (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-troubleshooting.html#target-not-inservice)
- https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html)

Ask our Experts

QUESTION 7 INCORRECT

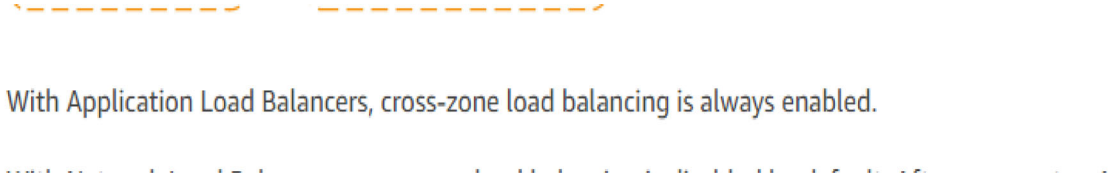
In an AWS Setup of a company, a web-based application has a fleet of 10 EC2 instances. 7 EC2 instances are present in Availability Zone A whereas 3 EC2 instances in Availability Zone B. There is also a Classic Load Balancer on HTTPS to distribute traffic equally across all the 10 EC2 instances. But the percentage (%) of requests received in Availability Zone B is greater than the percentage (%) of requests in Availability Zone A. Which of the following statements are correct as per given scenario? (choose multiple)

- [ ] **A.** Use Application Load Balancer to achieve this ability. ✔
- [x] **B.** Enable "split traffic equally" checkbox under load balancer configuration. ✖
- [x] **C.** Enable Cross-Zone Load Balancing ✔
- [ ] **D.** Use Network Load Balancer to achieve his ability.

Explanation :

Answer: A, C

Option A is correct.



- https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#availability-zones (https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#availability-zones)

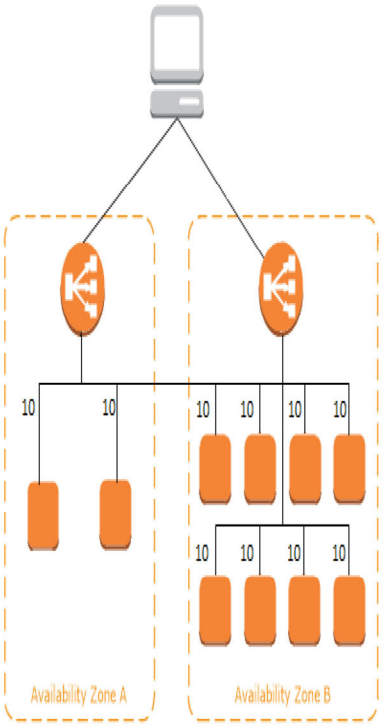
- Option B is not correct. There is no such option.
- Option C is correct.

Cross-Zone Load Balancing

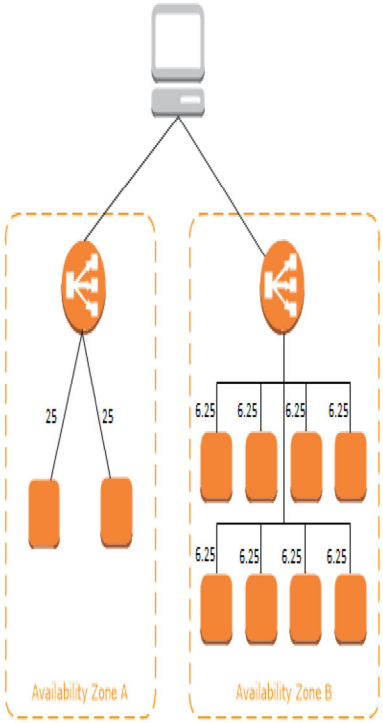
The nodes for your load balancer distribute requests from clients to registered targets. When cross-zone load balancing is enabled, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones. When cross-zone load balancing is disabled, each load balancer node distributes traffic across the registered targets in its Availability Zone only.

The following diagrams demonstrate the effect of cross-zone load balancing. There are two enabled Availability Zones, with 2 targets in Availability Zone A and 8 targets in Availability Zone B. Clients send requests, and Amazon Route 53 responds to each request with the IP address of one of the load balancer nodes. This distributes traffic such that each load balancer node receives 50% of the traffic from the clients. Each load balancer node distributes its share of the traffic across the registered targets in its scope.

If cross-zone load balancing is enabled, each of the 10 targets receives 10% of the traffic. This is because each load balancer node can route its 50% of the client traffic to all 10 targets.



If cross-zone load balancing is disabled, each of the 2 targets in Availability Zone A receives 25% of the traffic and each of the 8 targets in Availability Zone B receives 6.25% of the traffic. This is because each load balancer node can route its 50% of the client traffic only to targets in its Availability Zone.



- https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#availability-zones (https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html#availability-zones)
- Option D is not correct. With network load balancer, if we enable the cross-zone load balancing also, it supports only TCP protocol, not HTTPS protocol.

Ask our Experts

QUESTION 8 INCORRECT

Which are the following are features for monitoring application load balancer? Choose the 3 correct options.

- [x] **A.** CloudWatch metrics ✔
- [ ] **B.** Request tracing ✔
- [x] **C.** VPC Flow Logs ✖
- [x] **D.** CloudTrail logs ✔
- [ ] **E.** EC2 Flow Logs

Explanation :

Answer: A, B, D

Monitor Your Application Load Balancers

You can use the following features to monitor your load balancers, analyze traffic patterns, and troubleshoot issues with your load balancers and targets.

CloudWatch metrics

You can use Amazon CloudWatch to retrieve statistics about data points for your load balancers and targets as an ordered set of time-series data, known as *metrics*. You can use these metrics to verify that your system is performing as expected. For more information, see CloudWatch Metrics for Your Application Load Balancer.

Access logs

You can use access logs to capture detailed information about the requests made to your load balancer and store them as log files in Amazon S3. You can use these access logs to analyze traffic patterns and to troubleshoot issues with your targets. For more information, see Access Logs for Your Application Load Balancer.

Request tracing

You can use request tracing to track HTTP requests. The load balancer adds a header with a trace identifier to each request it receives. For more information, see Request Tracing for Your Application Load Balancer.

CloudTrail logs

You can use AWS CloudTrail to capture detailed information about the calls made to the Elastic Load Balancing API and store them as log files in Amazon S3. You can use these CloudTrail logs to determine which calls were made, the source IP address where the call came from, who made the call, when the call was made, and so on. For more information, see Logging API Calls for Your Application Load Balancer Using AWS CloudTrail.

- https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-monitoring.html (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-monitoring.html)

Ask our Experts

QUESTION 9 CORRECT

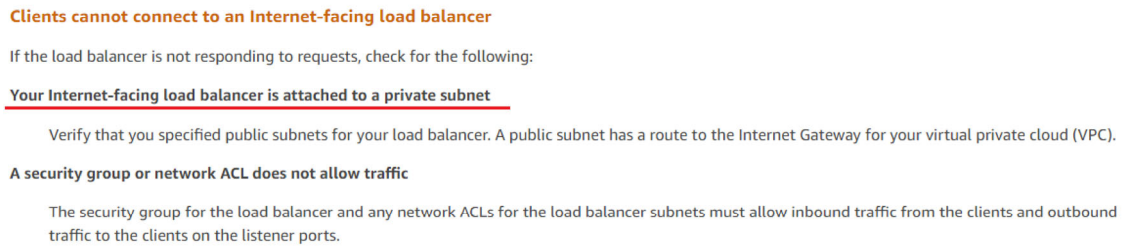
You have created an application load balancer and selected two EC2 instances as targets. However, when you are trying to make a request to load balancer from internet, the requests are getting failed. What could be the reason?

- A. The subnets specified for load balancer does not have internet gateway attached to their route tables. ✔
- B. Target EC2 instances are in private subnets without any internet gateway attached.
- C. There is no elastic IP address attached to the load balancer.
- D. Cross-zone load balancing is not enabled.

Explanation :

Answer: A

Option A is correct.



Option B is not correct.

You have can target EC2 instance in private subnet and can attach to load balancer which will be on public subnet.

- https://aws.amazon.com/premiumsupport/knowledge-center/public-load-balancer-private-ec2/ (https://aws.amazon.com/premiumsupport/knowledge-center/public-load-balancer-private-ec2/)

Option C is not correct.

Elastic load balancers are AWS highly available and scalable components managed by AWS. Once you create a load balancer, you will be given a internet-accessible(if the load balancer is internet facing) hostname to connect. AWS manages the underlying IP addresses and you do not need to attach an elastic IP address in order to connect from internet.

Option D is not correct.

With Application Load Balancers, cross-zone load balancing is always enabled.

Ask our Experts

QUESTION 10 CORRECT

Which of the following is a correct way to register a target in Elastic Load Balancer target group?

- A. EC2 instance names
- B. IP addresses ✔
- C. EC2 imageid
- D. EC2 Primary Network Inteface ID

Explanation :

Answer: B

Target Type

When you create a target group, you specify its target type, which determines how you specify its targets. After you create a target group, you cannot change its target type.

The following are the possible target types:

`instance`

The targets are specified by instance ID.

`ip`

The targets are specified by IP address.

When the target type is `ip`, you can specify IP addresses from one of the following CIDR blocks:

- The subnets of the VPC for the target group
- 10.0.0.0/8 (RFC 1918)
- 100.64.0.0/10 (RFC 6598)
- 172.16.0.0/12 (RFC 1918)
- 192.168.0.0/16 (RFC 1918)

These supported CIDR blocks enable you to register the following with a target group: ClassicLink instances, instances in a peered VPC, AWS resources that are addressable by IP address and port (for example, databases), and on-premises resources linked to AWS through AWS Direct Connect or a VPN connection.

Important

You can't specify publicly routable IP addresses.

If you specify targets using an instance ID, traffic is routed to instances using the primary private IP address specified in the primary network interface for the instance. If you specify targets using IP addresses, you can route traffic to an instance using any private IP address from one or more network interfaces. This enables multiple applications on an instance to use the same port. Each network interface can have its own security group.

- https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-target-groups.html#target-type (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-target-groups.html#target-type)

Ask our Experts

Finish Review (https://www.whizlabs.com/learn/course/aws-csaa-practice-tests/quiz/14827)

Certification

- Cloud Certification (https://www.whizlabs.com/cloud-certification-training-courses/)

Company

- Support (https://help.whizlabs.com/hc/en-us)
- Discussions (http://ask.whizlabs.com/)
- Blog (https://www.whizlabs.com/blog/)

- Java Certification (https://www.whizlabs.com/oracle-java-certifications/)
- PM Certification (https://www.whizlabs.com/project-management-certifications/)
- Big Data Certification (https://www.whizlabs.com/big-data-certifications/)

Mobile App

- Android Coming Soon
- iOS Coming Soon

Follow us

(https://www.facebook.com/whizlabs.software/)

(https://in.linkedin.com/company/whizlabs-software)

(https://twitter.com/whizlabs?lang=en)

(https://plus.google.com/+WhizlabsSoftware)