AWS SOLUTIONS ARCHITECT  
TOPIC BASED

PRACTICE QUESTIONS

Reference: [https://docs.aws.amazon.com/](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_ec2.html)

<https://tutorialsdojo.com/aws-cheat-sheets/>

# **EC2**

**1. QUESTION**

The company that you are working for has a highly available architecture consisting of an elastic load balancer and several EC2 instances configured with auto-scaling in three Availability Zones. You want to monitor your EC2 instances based on a particular metric, which is not readily available in CloudWatch.

Which of the following is a custom metric in CloudWatch which you have to manually set up?

* **A.** CPU Utilization of an EC2 instance
* **B.** Network packets out of an EC2 instance
* **C.** Disk Reads activity of an EC2 instance
* **D.** Memory Utilization of an EC2 instance (Correct)

**Explanation:** CloudWatch has available Amazon EC2 Metrics for you to use for monitoring. **CPU Utilization** identifies the processing power required to run an application upon a selected instance. **Network Utilization** identifies the volume of incoming and outgoing network traffic to a single instance. **Disk Reads metric** is used to determine the volume of the data the application reads from the hard disk of the instance. This can be used to determine the speed of the application. However, there are certain metrics that are not readily available in CloudWatch such as ***memory utilization***, ***disk space utilization***, and many others which can be collected by setting up a custom metric.

You need to prepare a custom metric using CloudWatch Monitoring Scripts which is written in Perl. You can also install CloudWatch Agent to collect more system-level metrics from Amazon EC2 instances. Here’s the list of custom metrics that you can set up:

– Memory utilization

– Disk swap utilization

– Disk space utilization

– Page file utilization

– Log collection

References:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_ec2.html>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html#using_put_script>

Check out this Amazon EC2 Cheat Sheet:

<https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/>

Check out this Amazon CloudWatch Cheat Sheet:

<https://tutorialsdojo.com/amazon-cloudwatch/>

**2. QUESTION**

A payment processing company plans to migrate its on-premises application to an Amazon EC2 instance. An IPv6 CIDR block is attached to the company’s Amazon VPC. Strict security policy mandates that the production VPC must only allow outbound communication over IPv6 between the instance and the internet but should prevent the internet from initiating an inbound IPv6 connection. The new architecture should also allow traffic flow inspection and traffic filtering.

What should a solutions architect do to meet these requirements?

* **A.** Launch the EC2 instance to a public subnet and attach an Internet Gateway to the VPC to allow outbound IPv6 communication to the internet. Use Traffic Mirroring to set up the required rules for traffic inspection and traffic filtering.
* **B.** Launch the EC2 instance to a private subnet and attach AWS PrivateLink interface endpoint to the VPC to control outbound IPv6 communication to the internet. Use Amazon GuardDuty to set up the required rules for traffic inspection and traffic filtering.
* **C.** Launch the EC2 instance to a private subnet and attach an Egress-Only Internet Gateway to the VPC to allow outbound IPv6 communication to the internet. Use AWS Network Firewall to set up the required rules for traffic inspection and traffic filtering. (Correct)
* **D.** Launch the EC2 instance to a private subnet and attach a NAT Gateway to the VPC to allow outbound IPv6 communication to the internet. Use AWS Firewall Manager to set up the required rules for traffic inspection and traffic filtering.

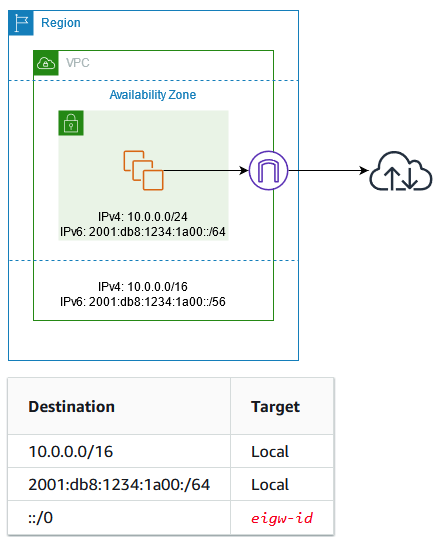
**Explanation:**

An egress-only internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows outbound communication over IPv6 from instances in your VPC to the internet and prevents it from initiating an IPv6 connection with your instances.

IPv6 addresses are globally unique and are therefore public by default. If you want your instance to be able to access the internet, but you want to prevent resources on the internet from initiating communication with your instance, you can use an egress-only internet gateway.

A subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet and a private subnet for resources that won’t be connected to the internet.

AWS Network Firewall is a managed service that makes it easy to deploy essential network protections for all of your Amazon Virtual Private Clouds (VPCs). The service can be set up with just a few clicks and scales automatically with your network traffic, so you don’t have to worry about deploying and managing any infrastructure. AWS Network Firewall includes features that provide protection from common network threats.



IPv6 addresses are globally unique and are therefore public by default. If you want your instance to be able to access the internet, but you want to prevent resources on the internet from initiating communication with your instance, you can use an egress-only internet gateway.

A subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet and a private subnet for resources that won’t be connected to the internet.

AWS Network Firewall is a managed service that makes it easy to deploy essential network protections for all of your Amazon Virtual Private Clouds (VPCs). The service can be set up with just a few clicks and scales automatically with your network traffic, so you don’t have to worry about deploying and managing any infrastructure. AWS Network Firewall includes features that provide protection from common network threats.

AWS Network Firewall’s stateful firewall can incorporate context from traffic flows, like tracking connections and protocol identification, to enforce policies such as preventing your VPCs from accessing domains using an unauthorized protocol. AWS Network Firewall’s intrusion prevention system (IPS) provides active traffic flow inspection so you can identify and block vulnerability exploits using signature-based detection. AWS Network Firewall also offers web filtering that can stop traffic to known bad URLs and monitor fully qualified domain names.

In this scenario, you can use an egress-only internet gateway to allow outbound IPv6 communication to the internet and then use the AWS Network Firewall to set up the required rules for traffic inspection and traffic filtering.

Hence, the correct answer for the scenario is: Launch the EC2 instance to a private subnet and attach an Egress-Only Internet Gateway to the VPC to allow outbound IPv6 communication to the internet. Use AWS Network Firewall to set up the required rules for traffic inspection and traffic filtering.

The option that says: Launch the EC2 instance to a private subnet and attach AWS PrivateLink interface endpoint to the VPC to control outbound IPv6 communication to the internet. Use Amazon GuardDuty to set up the required rules for traffic inspection and traffic filtering is incorrect because the AWS PrivateLink (which is also known as VPC Endpoint) is just a highly available, scalable technology that enables you to privately connect your VPC to the AWS services as if they were in your VPC. This service is not capable of controlling outbound IPv6 communication to the Internet. Furthermore, the Amazon GuardDuty service doesn’t have the features to do traffic inspection or filtering.

The option that says: Launch the EC2 instance to a public subnet and attach an Internet Gateway to the VPC to allow outbound IPv6 communication to the internet. Use Traffic Mirroring to set up the required rules for traffic inspection and traffic filtering is incorrect because an Internet Gateway does not limit or control any outgoing IPv6 connection. Take note that the requirement is to prevent the Internet from initiating an inbound IPv6 connection to your instance. This solution allows all kinds of traffic to initiate a connection to your EC2 instance hence, this option is wrong. In addition, the use of Traffic Mirroring is not appropriate as well. This is just an Amazon VPC feature that you can use to copy network traffic from an elastic network interface of type interface, not to filter or inspect the incoming/outgoing traffic.

The option that says: Launch the EC2 instance to a private subnet and attach a NAT Gateway to the VPC to allow outbound IPv6 communication to the internet. Use AWS Firewall Manager to set up the required rules for traffic inspection and traffic filtering is incorrect. While NAT Gateway has a NAT64 feature that translates an IPv6 address to IPv4, it will not prevent inbound IPv6 traffic from reaching the EC2 instance. You have to use the egress-only Internet Gateway instead. Moreover, the AWS Firewall Manager is neither capable of doing traffic inspection nor traffic filtering.

References:

<https://docs.aws.amazon.com/vpc/latest/userguide/egress-only-internet-gateway.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/configure-subnets.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Internet_Gateway.html>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**3. QUESTION**

A company has a cloud architecture that is composed of Linux and Windows EC2 instances that process high volumes of financial data 24 hours a day, 7 days a week. To ensure high availability of the systems, the Solutions Architect needs to create a solution that allows them to monitor the memory and disk utilization metrics of all the instances.

Which of the following is the most suitable monitoring solution to implement?

* **A.** Enable the Enhanced Monitoring option in EC2 and install CloudWatch agent to all the EC2 instances to be able to view the memory and disk utilization in the CloudWatch dashboard.
* **B.** Install the CloudWatch agent to all the EC2 instances that gather the memory and disk utilization data. View the custom metrics in the Amazon CloudWatch console. (Correct)
* **C.** Use Amazon Inspector and install the Inspector agent to all EC2 instances.
* **D.** Use the default CloudWatch configuration to EC2 instances where the memory and disk utilization metrics are already available. Install the AWS Systems Manager (SSM) Agent to all the EC2 instances.

**Explanation:**Enable the Enhanced Monitoring option in EC2 and install CloudWatch agent to all the EC2 instances to be able to view the memory and disk utilization in the CloudWatch dashboard.

Install the CloudWatch agent to all the EC2 instances that gather the memory and disk utilization data. View the custom metrics in the Amazon CloudWatch console.

Use Amazon Inspector and install the Inspector agent to all EC2 instances.

Use the default CloudWatch configuration to EC2 instances where the memory and disk utilization metrics are already available. Install the AWS Systems Manager (SSM) Agent to all the EC2 instances.  
  
Amazon CloudWatch has available Amazon EC2 Metrics for you to use for monitoring CPU utilization, Network utilization, Disk performance, and Disk Reads/Writes. In case you need to monitor the below items, you need to prepare a custom metric using a Perl or other shell script, as there are no ready to use metrics for:

1. Memory utilization
2. Disk swap utilization
3. Disk space utilization
4. Page file utilization
5. Log collection

Take note that there is a multi-platform CloudWatch agent which can be installed on both Linux and Windows-based instances. You can use a single agent to collect both system metrics and log files from Amazon EC2 instances and on-premises servers. This agent supports both Windows Server and Linux and enables you to select the metrics to be collected, including sub-resource metrics such as per-CPU core. It is recommended that you use the new agent instead of the older monitoring scripts to collect metrics and logs.

Hence, the correct answer is: Install the CloudWatch agent to all the EC2 instances that gathers the memory and disk utilization data. View the custom metrics in the Amazon CloudWatch console.

The option that says: Use the default CloudWatch configuration to EC2 instances where the memory and disk utilization metrics are already available. Install the AWS Systems Manager (SSM) Agent to all the EC2 instances is incorrect because, by default, CloudWatch does not automatically provide memory and disk utilization metrics of your instances. You have to set up custom CloudWatch metrics to monitor the memory, disk swap, disk space, and page file utilization of your instances.

The option that says: Enable the Enhanced Monitoring option in EC2 and install CloudWatch agent to all the EC2 instances to be able to view the memory and disk utilization in the CloudWatch dashboard is incorrect because Enhanced Monitoring is a feature of Amazon RDS. By default, Enhanced Monitoring metrics are stored for 30 days in the CloudWatch Logs.

The option that says: Use Amazon Inspector and install the Inspector agent to all EC2 instances is incorrect because Amazon Inspector is an automated security assessment service that helps you test the network accessibility of your Amazon EC2 instances and the security state of your applications running on the instances. It does not provide a custom metric to track the memory and disk utilization of each and every EC2 instance in your VPC.

References:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_ec2.html>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html#using_put_script>

Check out this Amazon CloudWatch Cheat Sheet:

<https://tutorialsdojo.com/amazon-cloudwatch/>

CloudWatch Agent vs SSM Agent vs Custom Daemon Scripts:

<https://tutorialsdojo.com/cloudwatch-agent-vs-ssm-agent-vs-custom-daemon-scripts/>

Comparison of AWS Services Cheat Sheets:

<https://tutorialsdojo.com/comparison-of-aws-services/>

**4. QUESTION**

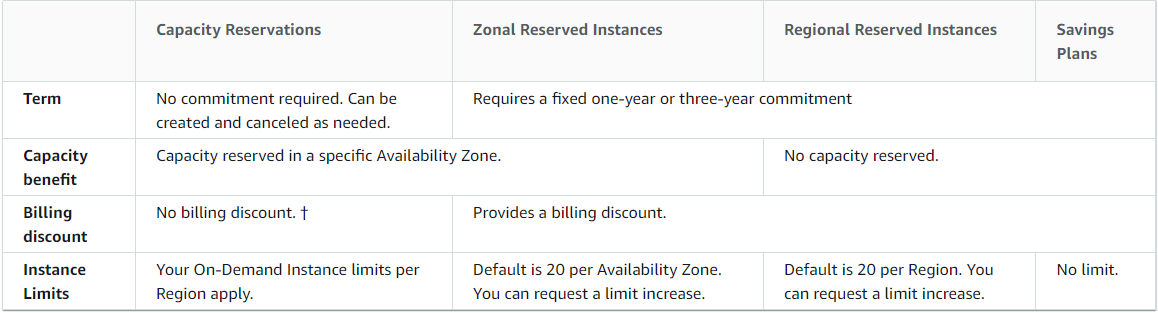
A multinational corporate and investment bank is regularly processing steady workloads of accruals, loan interests, and other critical financial calculations every night from 10 PM to 3 AM on their on-premises data center for their corporate clients. Once the process is done, the results are then uploaded to the Oracle General Ledger which means that the processing should not be delayed or interrupted. The CTO has decided to move its IT infrastructure to AWS to save costs. The company needs to reserve compute capacity in a specific Availability Zone to properly run their workloads.

As the Senior Solutions Architect, how can you implement a cost-effective architecture in AWS for their financial system?

* **A.** Use Regional Reserved Instances to reserve capacity on a specific Availability Zone and lower down the operating cost through its billing discounts.
* **B.** Use Dedicated Hosts which provide a physical host that is fully dedicated to running your instances, and bring your existing per-socket, per-core, or per-VM software licenses to reduce costs.
* **C.** Use On-Demand EC2 instances which allows you to pay for the instances that you launch and use by the second. Reserve compute capacity in a specific Availability Zone to avoid any interruption.
* **D.** Use On-Demand Capacity Reservations, which provide compute capacity that is always available on the specified recurring schedule. (Correct)

On-Demand Capacity Reservations enable you to reserve compute capacity for your Amazon EC2 instances in a specific Availability Zone for any duration. This gives you the ability to create and manage Capacity Reservations independently from the billing discounts offered by Savings Plans or Regional Reserved Instances.

By creating Capacity Reservations, you ensure that you always have access to EC2 capacity when you need it, for as long as you need it. You can create Capacity Reservations at any time, without entering into a one-year or three-year term commitment, and the capacity is available immediately. Billing starts as soon as the capacity is provisioned and the Capacity Reservation enters the active state. When you no longer need it, cancel the Capacity Reservation to stop incurring charges.



When you create a Capacity Reservation, you specify:

– The Availability Zone in which to reserve the capacity

– The number of instances for which to reserve capacity

– The instance attributes, including the instance type, tenancy, and platform/OS

Capacity Reservations can only be used by instances that match their attributes. By default, they are automatically used by running instances that match the attributes. If you don’t have any running instances that match the attributes of the Capacity Reservation, it remains unused until you launch an instance with matching attributes.

In addition, you can use Savings Plans and Regional Reserved Instances with your Capacity Reservations to benefit from billing discounts. AWS automatically applies your discount when the attributes of a Capacity Reservation match the attributes of a Savings Plan or Regional Reserved Instance.

Hence, the correct answer is to use On-Demand Capacity Reservations, which provide compute capacity that is always available on the specified recurring schedule.

Using On-Demand EC2 instances which allows you to pay for the instances that you launch and use by the second. Reserve compute capacity in a specific Availability Zone to avoid any interruption is incorrect because although an On-Demand instance is stable and suitable for processing critical data, it costs more than any other option. Moreover, the critical financial calculations are only done every night from 10 PM to 3 AM only and not 24/7. This means that your compute capacity will not be utilized for a total of 19 hours every single day. On-Demand instances cannot reserve compute capacity at all. So this option is incorrect.

Using Regional Reserved Instances to reserve capacity on a specific Availability Zone and lower down the operating cost through its billing discounts is incorrect because this feature is available in Zonal Reserved Instances only and not on Regional Reserved Instances.

Using Dedicated Hosts which provide a physical host that is fully dedicated to running your instances, and bringing your existing per-socket, per-core, or per-VM software licenses to reduce costs is incorrect because the use of a fully dedicated physical host is not warranted in this scenario. Moreover, this will be underutilized since you only run the process for 5 hours (from 10 PM to 3 AM only), wasting 19 hours of compute capacity every single day.

References:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-capacity-reservations.html>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html>

Check out this Amazon EC2 Cheat Sheet:

<https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/>

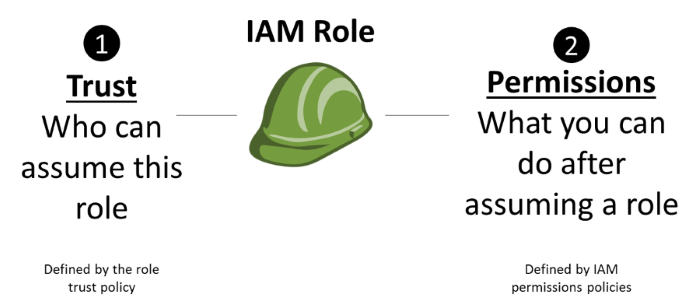
**5. QUESTION**

A company developed a meal planning application that provides meal recommendations for the week as well as the food consumption of the users. The application resides on an EC2 instance which requires access to various AWS services for its day-to-day operations.

Which of the following is the best way to allow the EC2 instance to access the S3 bucket and other AWS services?

* A. Store the API credentials in the EC2 instance.
* B. Add the API Credentials in the Security Group and assign it to the EC2 instance.
* C. Create a role in IAM and assign it to the EC2 instance. (Correct)
* D. Store the API credentials in a bastion host.

The best practice in handling API Credentials is to create a new role in the Identity Access Management (IAM) service and then assign it to a specific EC2 instance. In this way, you have a secure and centralized way of storing and managing your credentials.



Storing the API credentials in the EC2 instance, adding the API Credentials in the Security Group and assigning it to the EC2 instance, and storing the API credentials in a bastion host are incorrect because it is not secure to store nor use the API credentials from an EC2 instance. You should use IAM service instead.

Reference:

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

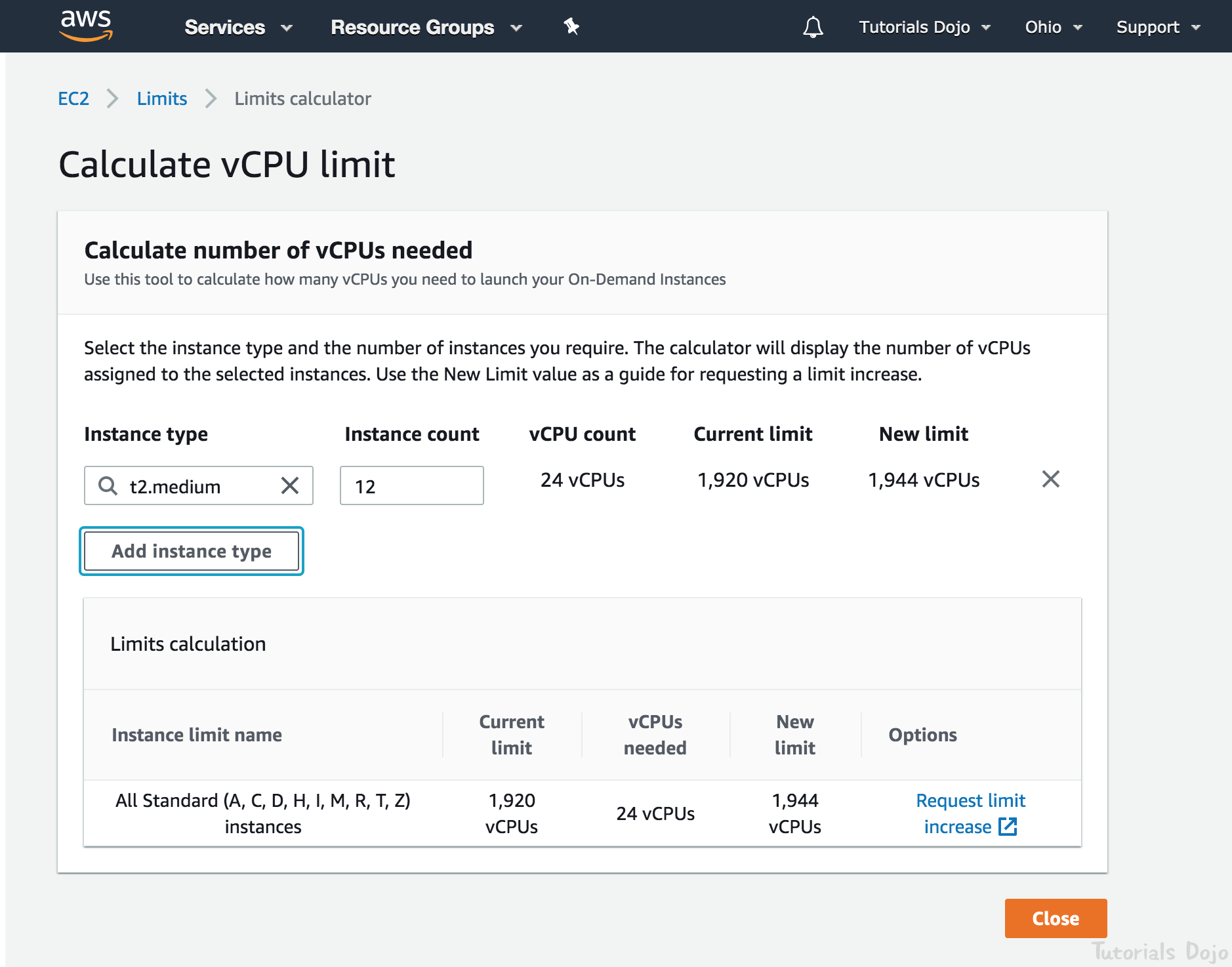
**6. QUESTION**

You are automating the creation of EC2 instances in your VPC. Hence, you wrote a python script to trigger the Amazon EC2 API to request 50 EC2 instances in a single Availability Zone. However, you noticed that after 20 successful requests, subsequent requests failed.

What could be a reason for this issue and how would you resolve it?

* **A.** There was an issue with the Amazon EC2 API. Just resend the requests and these will be provisioned successfully.
* **B.** There is a vCPU-based On-Demand Instance limit per region which is why subsequent requests failed. Just submit the limit increase form to AWS and retry the failed requests once approved. (Correct)
* **C.** By default, AWS allows you to provision a maximum of 20 instances per region. Select a different region and retry the failed request.
* **D.** By default, AWS allows you to provision a maximum of 20 instances per Availability Zone. Select a different Availability Zone and retry the failed request.

You are limited to running On-Demand Instances per your vCPU-based On-Demand Instance limit, purchasing 20 Reserved Instances, and requesting Spot Instances per your dynamic Spot limit per region. New AWS accounts may start with limits that are lower than the limits described here.



If you need more instances, complete the Amazon EC2 limit increase request form with your use case, and your limit increase will be considered. Limit increases are tied to the region they were requested for.

Hence, the correct answer is: There is a vCPU-based On-Demand Instance limit per region which is why subsequent requests failed. Just submit the limit increase form to AWS and retry the failed requests once approved.

The option that says: There was an issue with the Amazon EC2 API. Just resend the requests and these will be provisioned successfully is incorrect because you are limited to running On-Demand Instances per your vCPU-based On-Demand Instance limit. There is also a limit of purchasing 20 Reserved Instances and requesting Spot Instances per your dynamic Spot limit per region hence, there is no problem with the EC2 API.

The option that says: By default, AWS allows you to provision a maximum of 20 instances per region. Select a different region and retry the failed request is incorrect. There is no need to select a different region since this limit can be increased after submitting a request form to AWS.

The option that says: By default, AWS allows you to provision a maximum of 20 instances per Availability Zone. Select a different Availability Zone and retry the failed request is incorrect because the vCPU-based On-Demand Instance limit is set per region and not per Availability Zone. This can be increased after submitting a request form to AWS.

References:

<https://docs.aws.amazon.com/general/latest/gr/aws_service_limits.html#limits_ec2>

<https://aws.amazon.com/ec2/faqs/#How_many_instances_can_I_run_in_Amazon_EC2>

**7. QUESTION**

A company needs to deploy at least 2 EC2 instances to support the normal workloads of its application and automatically scale up to 6 EC2 instances to handle the peak load. The architecture must be highly available and fault-tolerant as it is processing mission-critical workloads.

As the Solutions Architect of the company, what should you do to meet the above requirement?

* **A.** Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 4. Deploy 2 instances in Availability Zone A and 2 instances in Availability Zone B.
* **B.** Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 6. Deploy 4 instances in Availability Zone A.
* **C.** Create an Auto Scaling group of EC2 instances and set the minimum capacity to 4 and the maximum capacity to 6. Deploy 2 instances in Availability Zone A and another 2 instances in Availability Zone B. (Correct)
* **D.** Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 6. Use 2 Availability Zones and deploy 1 instance for each AZ.

Amazon EC2 Auto Scaling helps ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. You create collections of EC2 instances, called Auto Scaling groups. You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size. You can also specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size.

To achieve highly available and fault-tolerant architecture for your applications, you must deploy all your instances in different Availability Zones. This will help you isolate your resources if an outage occurs. Take note that to achieve fault tolerance, you need to have redundant resources in place to avoid any system degradation in the event of a server fault or an Availability Zone outage. Having a fault-tolerant architecture entails an extra cost in running additional resources than what is usually needed. This is to ensure that the mission-critical workloads are processed.

Since the scenario requires at least 2 instances to handle regular traffic, you should have 2 instances running all the time even if an AZ outage occurred. You can use an Auto Scaling Group to automatically scale your compute resources across two or more Availability Zones. You have to specify the minimum capacity to 4 instances and the maximum capacity to 6 instances. If each AZ has 2 instances running, even if an AZ fails, your system will still run a minimum of 2 instances.

Hence, the correct answer in this scenario is: Create an Auto Scaling group of EC2 instances and set the minimum capacity to 4 and the maximum capacity to 6. Deploy 2 instances in Availability Zone A and another 2 instances in Availability Zone B.

The option that says: Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 6. Deploy 4 instances in Availability Zone A is incorrect because the instances are only deployed in a single Availability Zone. It cannot protect your applications and data from datacenter or AZ failures.

The option that says: Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 6. Use 2 Availability Zones and deploy 1 instance for each AZ is incorrect. It is required to have 2 instances running all the time. If an AZ outage happened, ASG will launch a new instance on the unaffected AZ. This provisioning does not happen instantly, which means that for a certain period of time, there will only be 1 running instance left.

The option that says: Create an Auto Scaling group of EC2 instances and set the minimum capacity to 2 and the maximum capacity to 4. Deploy 2 instances in Availability Zone A and 2 instances in Availability Zone B is incorrect. Although this fulfills the requirement of at least 2 EC2 instances and high availability, the maximum capacity setting is wrong. It should be set to 6 to properly handle the peak load. If an AZ outage occurs and the system is at its peak load, the number of running instances in this setup will only be 4 instead of 6 and this will affect the performance of your application.

References:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html>

<https://docs.aws.amazon.com/documentdb/latest/developerguide/regions-and-azs.html>

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**8. QUESTION**

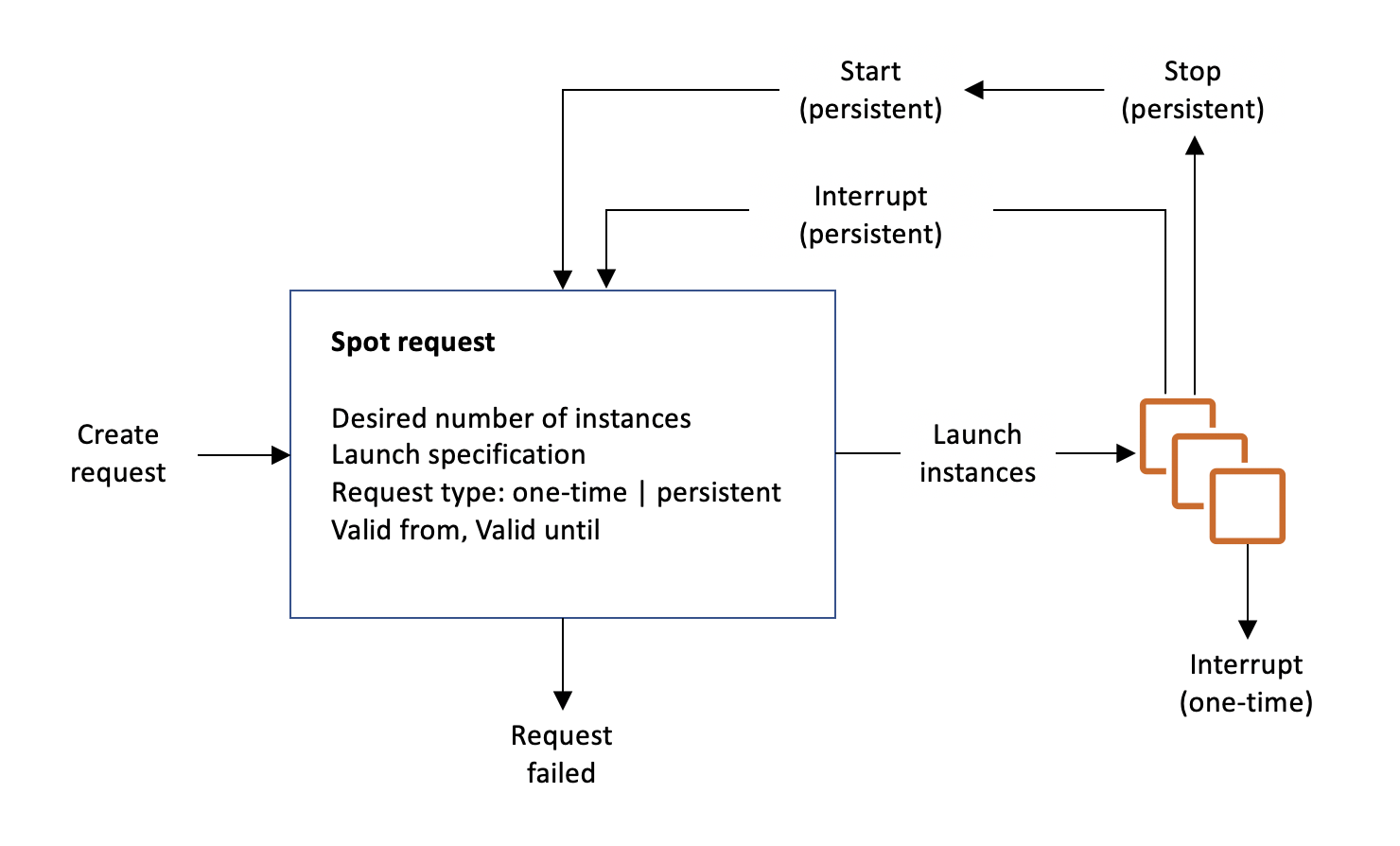
The media company that you are working for has a video transcoding application running on Amazon EC2. Each EC2 instance polls a queue to find out which video should be transcoded, and then runs a transcoding process. If this process is interrupted, the video will be transcoded by another instance based on the queuing system. This application has a large backlog of videos which need to be transcoded. Your manager would like to reduce this backlog by adding more EC2 instances, however, these instances are only needed until the backlog is reduced.

In this scenario, which type of Amazon EC2 instance is the most cost-effective type to use?

* **A.** On-demand instances
* **B.** Dedicated instances
* **C.** Spot instances (Correct)
* **D.** Reserved instances

**Explanation:**

You require an instance that will be used not as a primary server but as a spare compute resource to augment the transcoding process of your application. These instances should also be terminated once the backlog has been significantly reduced. In addition, the scenario mentions that if the current process is interrupted, the video can be transcoded by another instance based on the queuing system. This means that the application can gracefully handle an unexpected termination of an EC2 instance, like in the event of a Spot instance termination when the Spot price is greater than your set maximum price. Hence, an Amazon EC2 Spot instance is the best and cost-effective option for this scenario.



Amazon EC2 Spot instances are spare compute capacity in the AWS cloud available to you at steep discounts compared to On-Demand prices. EC2 Spot enables you to optimize your costs on the AWS cloud and scale your application’s throughput up to 10X for the same budget. By simply selecting Spot when launching EC2 instances, you can save up to 90% on On-Demand prices. The only difference between On-Demand instances and Spot Instances is that Spot instances can be interrupted by EC2 with two minutes of notification when the EC2 needs the capacity back.

You can specify whether Amazon EC2 should hibernate, stop, or terminate Spot Instances when they are interrupted. You can choose the interruption behavior that meets your needs.

Take note that there is no “bid price” anymore for Spot EC2 instances since March 2018. You simply have to set your maximum price instead.

Reserved instances and Dedicated instances are incorrect as both do not act as spare compute capacity.

On-demand instances is a valid option but a Spot instance is much cheaper than On-Demand.

References:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-interruptions.html>

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/how-spot-instances-work.html>

<https://aws.amazon.com/blogs/compute/new-amazon-ec2-spot-pricing>

Check out this Amazon EC2 Cheat Sheet:

<https://tutorialsdojo.com/amazon-elastic-compute-cloud-amazon-ec2/>

# **AUTOSCALING**

**1. QUESTION**

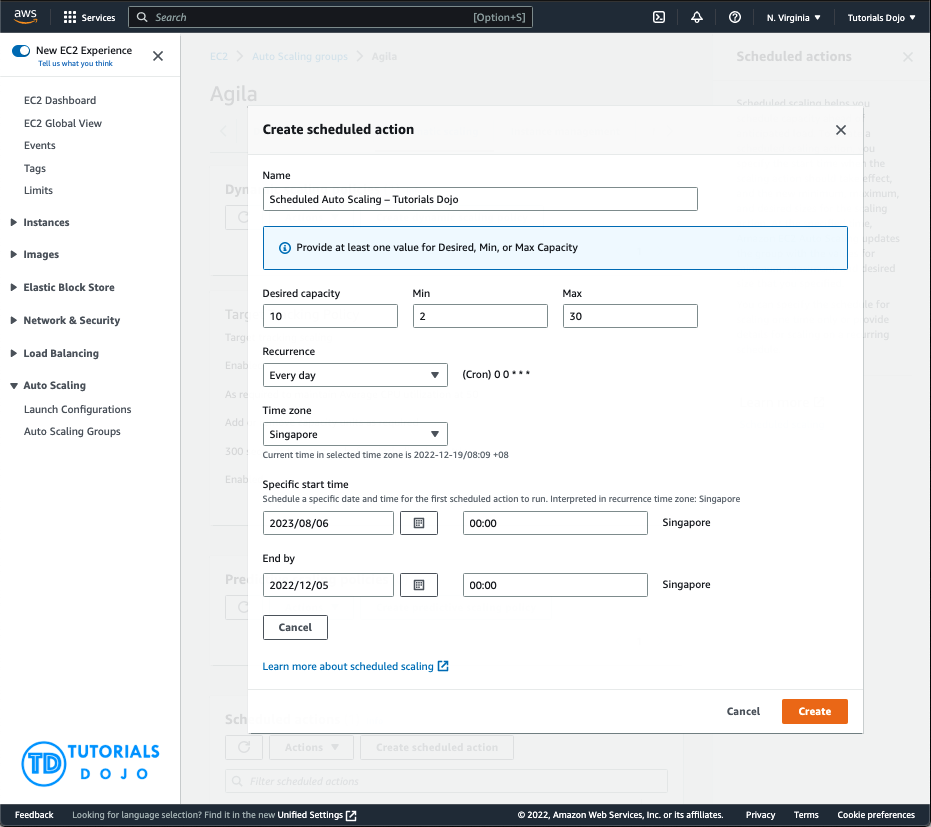
A tech company has a CRM application hosted on an Auto Scaling group of On-Demand EC2 instances with different instance types and sizes. The application is extensively used during office hours from 9 in the morning to 5 in the afternoon. Their users are complaining that the performance of the application is slow during the start of the day but then works normally after a couple of hours.

Which of the following is the MOST operationally efficient solution to implement to ensure the application works properly at the beginning of the day?

* **A.** Configure a Dynamic scaling policy for the Auto Scaling group to launch new instances based on the CPU utilization.
* **B.** Configure a Dynamic scaling policy for the Auto Scaling group to launch new instances based on the Memory utilization.
* **C.** Configure a Scheduled scaling policy for the Auto Scaling group to launch new instances before the start of the day. (Correct)
* **D.** Configure a Predictive scaling policy for the Auto Scaling group to automatically adjust the number of Amazon EC2 instances

**Explanation:**

Scaling based on a schedule allows you to scale your application in response to predictable load changes. For example, every week the traffic to your web application starts to increase on Wednesday, remains high on Thursday, and starts to decrease on Friday. You can plan your scaling activities based on the predictable traffic patterns of your web application.



To configure your Auto Scaling group to scale based on a schedule, you create a scheduled action. The scheduled action tells Amazon EC2 Auto Scaling to perform a scaling action at specified times. To create a scheduled scaling action, you specify the start time when the scaling action should take effect and the new minimum, maximum, and desired sizes for the scaling action. At the specified time, Amazon EC2 Auto Scaling updates the group with the values for minimum, maximum, and desired size specified by the scaling action. You can create scheduled actions for scaling one time only or for scaling on a recurring schedule.

Hence, configuring a Scheduled scaling policy for the Auto Scaling group to launch new instances before the start of the day is the correct answer. You need to configure a Scheduled scaling policy. This will ensure that the instances are already scaled up and ready before the start of the day since this is when the application is used the most.

The following options are both incorrect. Although these are valid solutions, it is still better to configure a Scheduled scaling policy as you already know the exact peak hours of your application. By the time either the CPU or Memory hits a peak, the application already has performance issues, so you need to ensure the scaling is done beforehand using a Scheduled scaling policy:

-Configure a Dynamic scaling policy for the Auto Scaling group to launch new instances based on the CPU utilization

-Configure a Dynamic scaling policy for the Auto Scaling group to launch new instances based on the Memory utilization

The option that says: Configure a Predictive scaling policy for the Auto Scaling group to automatically adjust the number of Amazon EC2 instances is incorrect. Although this type of scaling policy can be used in this scenario, it is not the most operationally efficient option. Take note that the scenario mentioned that the Auto Scaling group consists of Amazon EC2 instances with different instance types and sizes. Predictive scaling assumes that your Auto Scaling group is homogenous, which means that all EC2 instances are of equal capacity. The forecasted capacity can be inaccurate if you are using a variety of EC2 instance sizes and types on your Auto Scaling group.

References:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/schedule_time.html>

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-scheduled-scaling.html>

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-predictive-scaling.html#predictive-scaling-limitations>

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**2. QUESTION**

A suite of web applications is hosted in an Auto Scaling group of EC2 instances across three Availability Zones and is configured with default settings. There is an Application Load Balancer that forwards the request to the respective target group on the URL path. The scale-in policy has been triggered due to the low number of incoming traffic to the application.

Which EC2 instance will be the first one to be terminated by your Auto Scaling group?

* **A.** The EC2 instance which has the least number of user sessions
* **B.** The EC2 instance which has been running for the longest time
* **C.** The EC2 instance launched from the oldest launch configuration (Correct)
* **D.** The instance will be randomly selected by the Auto Scaling group

**Explanation:**

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:

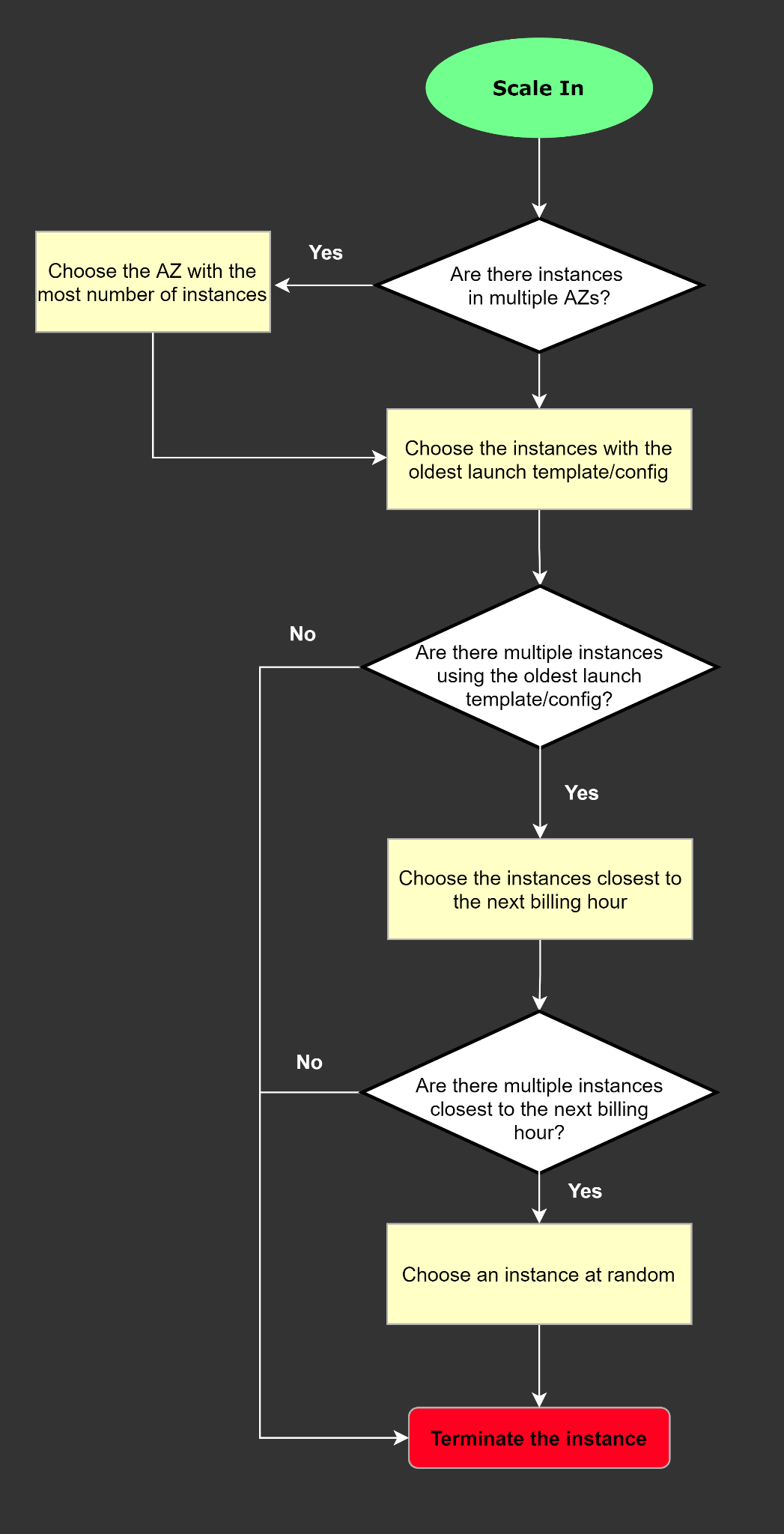
1. If there are instances in multiple Availability Zones, choose the Availability Zone with the most instances and at least one instance that is not protected from scale in. If there is more than one Availability Zone with this number of instances, choose the Availability Zone with the instances that 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 to terminate based on the above criteria, 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, choose one of these instances at random.

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



References:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html#default-termination-policy>

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

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**3. QUESTION**

A commercial bank has a forex trading application. They created an Auto Scaling group of EC2 instances that allow the bank to cope with the current traffic and achieve cost-efficiency. They want the Auto Scaling group to behave in such a way that it will follow a predefined set of parameters before it scales down the number of EC2 instances, which protects the system from unintended slowdown or unavailability.

Which of the following statements are true regarding the cooldown period? (Select TWO.)

* **A.** It ensures that the Auto Scaling group launches or terminates additional EC2 instances without any downtime.
* **B.** It ensures that the Auto Scaling group does not launch or terminate additional EC2 instances before the previous scaling activity takes effect. (Correct)
* **C.** Its default value is 600 seconds.
* **D.** It ensures that before the Auto Scaling group scales out, the EC2 instances have an ample time to cooldown.
* **E.** Its default value is 300 seconds. (Correct)

Explanation:

In Auto Scaling, the following statements are correct regarding the cooldown period:

It ensures that the Auto Scaling group does not launch or terminate additional EC2 instances before the previous scaling activity takes effect.

Its default value is 300 seconds.

It is a configurable setting for your Auto Scaling group.

The following options are incorrect:

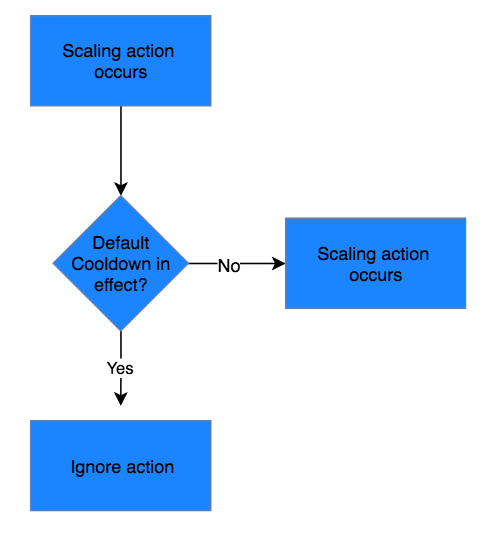
– It ensures that before the Auto Scaling group scales out, the EC2 instances have ample time to cooldown.

– It ensures that the Auto Scaling group launches or terminates additional EC2 instances without any downtime.

– Its default value is 600 seconds.

These statements are inaccurate and don’t depict what the word “cooldown” actually means for Auto Scaling. The cooldown period is a configurable setting for your Auto Scaling group that helps to ensure that it doesn’t launch or terminate additional instances before the previous scaling activity takes effect. After the Auto Scaling group dynamically scales using a simple scaling policy, it waits for the cooldown period to complete before resuming scaling activities.

The figure below demonstrates the scaling cooldown:



Reference:

<http://docs.aws.amazon.com/autoscaling/latest/userguide/as-instance-termination.html>

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**4. QUESTION**

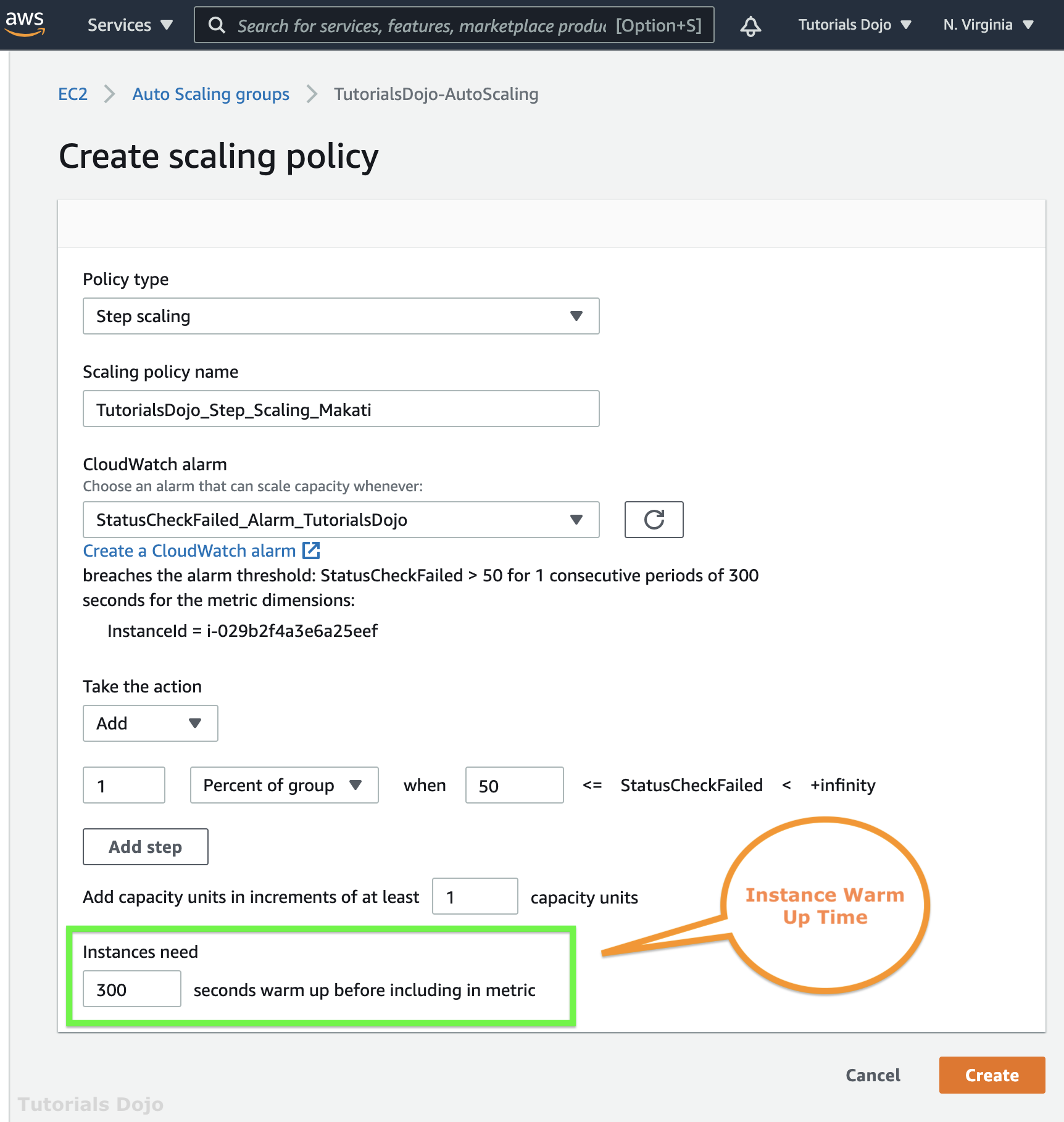
A web application hosted in an Auto Scaling group of EC2 instances in AWS. The application receives a burst of traffic every morning, and a lot of users are complaining about request timeouts. The EC2 instance takes 1 minute to boot up before it can respond to user requests. The cloud architecture must be redesigned to better respond to the changing traffic of the application.

How should the Solutions Architect redesign the architecture?

* **A.** Create a CloudFront distribution and set the EC2 instance as the origin.
* **B.** Create a step scaling policy and configure an instance warm-up time condition. (Correct)
* **C.** Create a new launch template and upgrade the size of the instance.
* **D.** Create a Network Load Balancer with slow-start mode.

**Explanation:**

Amazon EC2 Auto Scaling helps you maintain application availability and allows you to automatically add or remove EC2 instances according to conditions you define. You can use the fleet management features of EC2 Auto Scaling to maintain the health and availability of your fleet. You can also use the dynamic and predictive scaling features of EC2 Auto Scaling to add or remove EC2 instances. Dynamic scaling responds to changing demand and predictive scaling automatically schedules the right number of EC2 instances based on predicted demand. Dynamic scaling and predictive scaling can be used together to scale faster.



Step scaling applies “step adjustments” which means you can set multiple actions to vary the scaling depending on the size of the alarm breach. When you create a step scaling policy, you can also specify the number of seconds that it takes for a newly launched instance to warm up.

Hence, the correct answer is: Create a step scaling policy and configure an instance warm-up time condition.

The option that says: Create a Network Load Balancer with slow start mode is incorrect because Network Load Balancer does not support slow start mode. If you need to enable slow start mode, you should use Application Load Balancer.

The option that says: Create a new launch template and upgrade the size of the instance is incorrect because a larger instance does not always improve the boot time. Instead of upgrading the instance, you should create a step scaling policy and add a warm-up time.

The option that says: Create a CloudFront distribution and set the EC2 instance as the origin is incorrect because this approach only resolves the traffic latency. Take note that the requirement in the scenario is to resolve the timeout issue and not the traffic latency.

References:

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

<https://aws.amazon.com/ec2/autoscaling/faqs/>

Check out these AWS Cheat Sheets:

<https://tutorialsdojo.com/aws-auto-scaling/>

<https://tutorialsdojo.com/step-scaling-vs-simple-scaling-policies-in-amazon-ec2/>

**5. QUESTION**

A tech company is currently using Auto Scaling for their web application. A new AMI now needs to be used for launching a fleet of EC2 instances. Which of the following changes needs to be done?

* **A.** Do nothing. You can start directly launching EC2 instances in the Auto Scaling group with the same launch configuration.
* **B.** Create a new launch configuration.(Correct)
* **C.** Create a new target group.
* **D.** Create a new target group and launch configuration.

**Explanation:**

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.

For this scenario, you have to create a new launch configuration. Remember that you can’t modify a launch configuration after you’ve created it.

Hence, the correct answer is: Create a new launch configuration.

The option that says: Do nothing. You can start directly launching EC2 instances in the Auto Scaling group with the same launch configuration is incorrect because what you are trying to achieve is change the AMI being used by your fleet of EC2 instances. Therefore, you need to change the launch configuration to update what your instances are using.

The option that says: create a new target group and create a new target group and launch configuration are both incorrect because you only want to change the AMI being used by your instances, and not the instances themselves. Target groups are primarily used in ELBs and not in Auto Scaling. The scenario didn’t mention that the architecture has a load balancer. Therefore, you should be updating your launch configuration, not the target group.

References:

<http://docs.aws.amazon.com/autoscaling/latest/userguide/LaunchConfiguration.html>

[https://docs.aws.amazon.com/autoscaling/ec2/userguide/AutoScalingGroup.html](http://docs.aws.amazon.com/autoscaling/latest/userguide/LaunchConfiguration.html)

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**6. QUESTION**

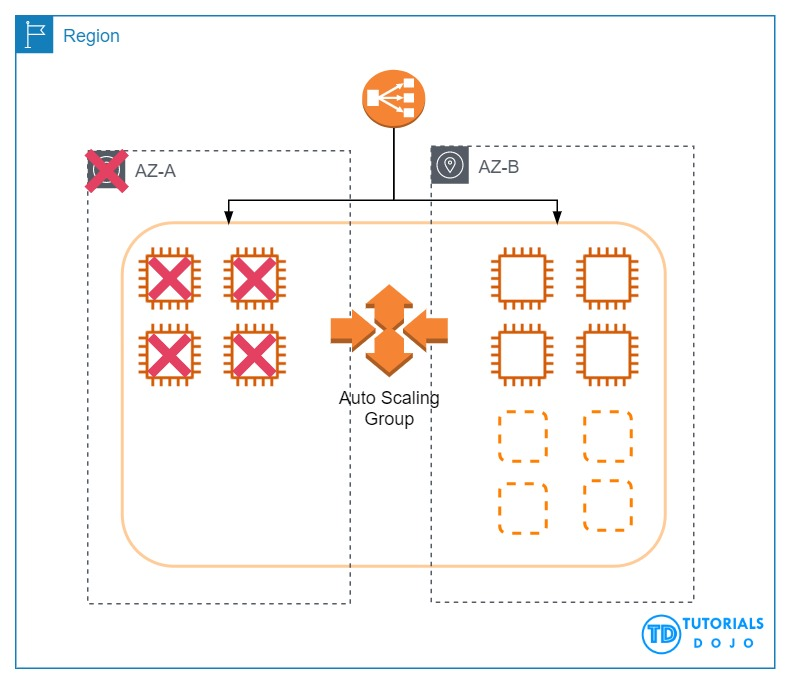
A major TV network has a web application running on eight Amazon T3 EC2 instances behind an application load balancer. The number of requests that the application processes are consistent and do not experience spikes. A Solutions Architect must configure an Auto Scaling group for the instances to ensure that the application is running at all times.

Which of the following options can satisfy the given requirements?

* **A.** Deploy two EC2 instances with Auto Scaling in four regions behind an Amazon Elastic Load Balancer.
* **B.** Deploy four EC2 instances with Auto Scaling in one Availability Zone and four in another availability zone in the same region behind an Amazon Elastic Load Balancer.(Correct)
* **C.** Deploy four EC2 instances with Auto Scaling in one region and four in another region behind an Amazon Elastic Load Balancer.
* **D.** Deploy eight EC2 instances with Auto Scaling in one Availability Zone behind an Amazon Elastic Load Balancer.

**Explanation:**

The best option to take is to deploy four EC2 instances in one Availability Zone and four in another availability zone in the same region behind an Amazon Elastic Load Balancer. In this way, if one availability zone goes down, there is still another available zone that can accommodate traffic.



When the first AZ goes down, the second AZ will only have an initial 4 EC2 instances. This will eventually be scaled up to 8 instances since the solution is using Auto Scaling.

The 110% compute capacity for the 4 servers might cause some degradation of the service but not a total outage since there are still some instances that handle the requests. Depending on your scale-up configuration in your Auto Scaling group, the additional 4 EC2 instances can be launched in a matter of minutes.

T3 instances also have a Burstable Performance capability to burst or go beyond the current compute capacity of the instance to higher performance as required by your workload. So your 4 servers will be able to manage 110% compute capacity for a short period of time. This is the power of cloud computing versus our on-premises network architecture. It provides elasticity and unparalleled scalability.

Take note that Auto Scaling will launch additional EC2 instances to the remaining Availability Zone/s in the event of an Availability Zone outage in the region. Hence, the correct answer is the option that says: Deploy four EC2 instances with Auto Scaling in one Availability Zone and four in another availability zone in the same region behind an Amazon Elastic Load Balancer.

The option that says: Deploy eight EC2 instances with Auto Scaling in one Availability Zone behind an Amazon Elastic Load Balancer is incorrect because this architecture is not highly available. If that Availability Zone goes down, then your web application will be unreachable.

The options that say: Deploy four EC2 instances with Auto Scaling in one region and four in another region behind an Amazon Elastic Load Balancer and Deploy two EC2 instances with Auto Scaling in four regions behind an Amazon Elastic Load Balancer are incorrect because the ELB is designed to only run in one region and not across multiple regions.



References:

<https://aws.amazon.com/elasticloadbalancing/>

[https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-increase-availability.html](https://aws.amazon.com/elasticloadbalancing/)

Check out this AWS Elastic Load Balancing (ELB) Cheat Sheet:

<https://tutorialsdojo.com/aws-elastic-load-balancing-elb/>

**7. QUESTION**

A commercial bank has designed its next-generation online banking platform to use a distributed system architecture. As their Software Architect, you have to ensure that their architecture is highly scalable, yet still cost-effective.

Which of the following will provide the most suitable solution for this scenario?

* **A.** Launch an Auto-Scaling group of EC2 instances to host your application services and an SQS queue. Include an Auto Scaling trigger to watch the SQS queue size which will either scale in or scale out the number of EC2 instances based on the queue. (Correct)
* **B.** Launch multiple EC2 instances behind an Application Load Balancer to host your application services, and SWF which will act as a highly-scalable buffer that stores messages as they travel between distributed applications.
* **C.** Launch multiple On-Demand EC2 instances to host your application services and an SQS queue which will act as a highly-scalable buffer that stores messages as they travel between distributed applications.
* **D.** Launch multiple EC2 instances behind an Application Load Balancer to host your application services and SNS which will act as a highly-scalable buffer that stores messages as they travel between distributed applications.

There are three main parts in a distributed messaging system: the components of your distributed system which can be hosted on EC2 instance; your queue (distributed on Amazon SQS servers); and the messages in the queue.

To improve the scalability of your distributed system, you can add Auto Scaling group to your EC2 instances.

References:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-using-sqs-queue.html>

[https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-basic-architecture.html](https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-using-sqs-queue.html)

Check out this AWS Auto Scaling Cheat Sheet:

<https://tutorialsdojo.com/aws-auto-scaling/>

**8. QUESTION**

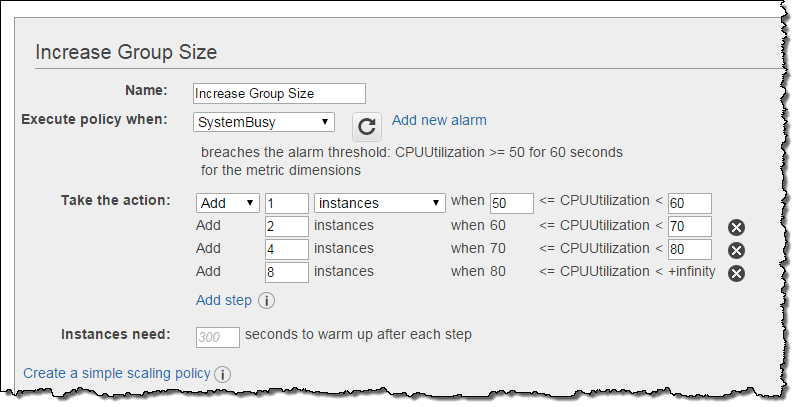
An application is hosted in an Auto Scaling group of EC2 instances. To improve the monitoring process, you have to configure the current capacity to increase or decrease based on a set of scaling adjustments. This should be done by specifying the scaling metrics and threshold values for the CloudWatch alarms that trigger the scaling process.

Which of the following is the most suitable type of scaling policy that you should use?

* **A.** Target tracking scaling
* **B.** Step scaling (Correct)
* **C.** Simple scaling
* **D.** Scheduled Scaling

With step scaling, you choose scaling metrics and threshold values for the CloudWatch alarms that trigger the scaling process as well as define how your scalable target should be scaled when a threshold is in breach for a specified number of evaluation periods. Step scaling policies increase or decrease the current capacity of a scalable target based on a set of scaling adjustments, known as step adjustments. The adjustments vary based on the size of the alarm breach. After a scaling activity is started, the policy continues to respond to additional alarms, even while a scaling activity is in progress. Therefore, all alarms that are breached are evaluated by Application Auto Scaling as it receives the alarm messages.

When you configure dynamic scaling, you must define how to scale in response to changing demand. For example, you have a web application that currently runs on two instances and you want the CPU utilization of the Auto Scaling group to stay at around 50 percent when the load on the application changes. This gives you extra capacity to handle traffic spikes without maintaining an excessive amount of idle resources. You can configure your Auto Scaling group to scale automatically to meet this need. The policy type determines how the scaling action is performed.



Amazon EC2 Auto Scaling supports the following types of scaling policies:

Target tracking scaling – Increase or decrease the current capacity of the group based on a target value for a specific metric. This is similar to the way that your thermostat maintains the temperature of your home – you select a temperature and the thermostat does the rest.

Step scaling – Increase or decrease the current capacity of the group based on a set of scaling adjustments, known as step adjustments, that vary based on the size of the alarm breach.

Simple scaling – Increase or decrease the current capacity of the group based on a single scaling adjustment.

If you are scaling based on a utilization metric that increases or decreases proportionally to the number of instances in an Auto Scaling group, then it is recommended that you use target tracking scaling policies. Otherwise, it is better to use step scaling policies instead.

Hence, the correct answer in this scenario is Step Scaling.

Target tracking scaling is incorrect because the target tracking scaling policy increases or decreases the current capacity of the group based on a target value for a specific metric instead of a set of scaling adjustments.

Simple scaling is incorrect because the simple scaling policy increases or decreases the current capacity of the group based on a single scaling adjustment instead of a set of scaling adjustments.

Scheduled Scaling is incorrect because the scheduled scaling policy is based on a schedule that allows you to set your own scaling schedule for predictable load changes. This is not considered as one of the types of dynamic scaling.

References:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scale-based-on-demand.html>

[https://docs.aws.amazon.com/autoscaling/application/userguide/application-auto-scaling-step-scaling-policies.html](https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scale-based-on-demand.html)

# **Identity and Access Management**

**1. QUESTION**

A company has multiple AWS sandbox accounts that are used by its development team. All developers must be given access to the contents of one of the main account’s S3 buckets. For security purposes, any personally identifiable information (PII) or financial data uploaded in the bucket must be continuously monitored and removed.

How can this be done at the lowest possible cost and with the least amount of configuration effort?

* **A.** Create an S3 bucket policy that grants access from the sandbox accounts. Use Amazon Macie to discover personally identifiable information (PII) or financial data. (Correct)
* **B.** Configure cross-account replication on the S3 bucket. Integrate AWS Audit Manager with the S3 bucket to discover any personally identifiable information (PII) or financial data.
* **C.** Add S3 read permission to the IAM policy of each IAM user from the sandbox accounts. Use Amazon Detective to discover personally identifiable information (PII) or financial data.
* **D.** Generate a pre-signed URL for the objects on the S3 bucket. Use the Amazon S3 Storage Lens to discover personally identifiable information (PII) or financial data.

**Explanation:**

In Amazon S3, you can grant users in another AWS account (Account B) granular cross-account access to objects owned by your account (Account A). Depending on the type of access that you want to provide, use one of the following solutions to grant cross-account access to objects:

– AWS Identity and Access Management (IAM) policies and resource-based bucket policies ( for programmatic-only access to S3 bucket objects

– IAM policies and resource-based Access Control Lists (ACLs) for programmatic-only access to S3 bucket objects

– Cross-account IAM roles for programmatic and console access to S3 bucket objects.

Not all AWS services support resource-based policies. Therefore, you can use cross-account IAM roles to centralize permission management when providing cross-account access to multiple services. Using cross-account IAM roles simplifies provisioning cross-account access to S3 objects that are stored in multiple S3 buckets. As a result, you don’t need to manage multiple policies for S3 buckets. This method allows cross-account access to objects owned or uploaded by another AWS account or AWS services. If you don’t use cross-account IAM roles, then the object ACL must be modified.



In the scenario, the best approach to granting the developers access to the main account’s S3 bucket is by configuring the bucket policy to allow IAM users from different accounts to call the GetObject method. This is a neater and simpler solution than the rest because you control access from a single location without any additional costs.

Hence, the correct answer is: Create an S3 bucket policy that grants access from the sandbox accounts. Use Amazon Macie to discover personally identifiable information (PII) or financial data.

The option that says: **Configure cross-account replication on the S3 bucket. Integrate AWS Audit Manager with the S3 bucket to discover any personally identifiable information (PII) or financial data** is incorrect. This can work, but it is an inefficient way of solving the problem. The developers only need to access the S3 objects in another account; they do not need to own a copy of them. On top of that, replication incurs additional costs. In addition, the AWS Audit Manager simply helps you continuously audit your AWS usage to simplify how you assess risk and compliance with regulations and industry standards. AWS Audit Manager is not capable of discovering personally identifiable information (PII) or financial data in your S3 bucket.

The option that says: **Generate a pre-signed URL for the objects on the S3 bucket. Use the Amazon S3 Storage Lens to discover personally identifiable information (PII) or financial data** is incorrect. Since objects shared using presigned URLs are time-limited, you’d have to regenerate the URL for each object every time it expires and resend the new link to the developers. This approach does not scale well and is not a good use for the S3 presigned URL. Moreover, the Amazon S3 Storage Lens feature just provides a single view of object storage usage and activity across your entire Amazon S3 storage.

The option that says: **Add S3 read permission to the IAM policy of each IAM user from the sandbox accounts. Use Amazon Detective to discover personally identifiable information (PII) or financial data** is incorrect. You would have to jump from one account to another to set this up. It works, but depending on the number of accounts and IAM users, it will entail a lot of configuration overhead. Although Amazon Detective is a security service, it does not have any capability to discover any PII or financial data in your S3 bucket. Its primary purpose is to analyze and visualize security data to rapidly get to the root cause of potential security issues.

References:

<https://aws.amazon.com/premiumsupport/knowledge-center/cross-account-access-s3/>

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-walkthroughs-managing-access-example2.html>

<https://aws.amazon.com/macie/>

Check out this Amazon S3 Cheat Sheet:

[https://tutorialsdojo.com/amazon-s3/](https://portal.tutorialsdojo.com/courses/aws-certified-solutions-architect-associate-practice-exams/lessons/practice-exams-topic-based/quizzes/aws-certified-solutions-architect-associate-practice-exam-topic-based-iam/%C2%A0https://tutorialsdojo.com/amazon-s3/)

**2. QUESTION**

A company needs to integrate the Lightweight Directory Access Protocol (LDAP) directory service from the on-premises data center to the AWS VPC using IAM. The identity store which is currently being used is not compatible with SAML.

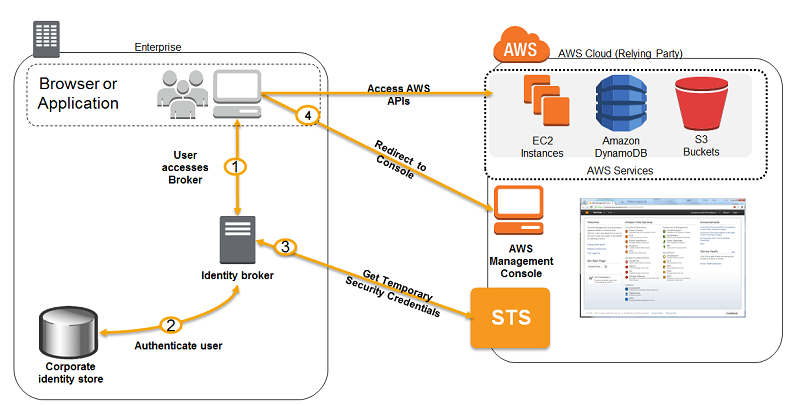
Which of the following provides the most valid approach to implement the integration?

* **A.** Develop an on-premises custom identity broker application and use STS to issue short-lived AWS credentials. (Correct)
* **B.** Use AWS Single Sign-On (SSO) service to enable single sign-on between AWS and your LDAP.
* **C.** Use IAM roles to rotate the IAM credentials whenever LDAP credentials are updated.
* **D.** Use an IAM policy that references the LDAP identifiers and AWS credentials.

If your identity store is not compatible with SAML 2.0 then you can build a custom identity broker application to perform a similar function. The broker application authenticates users, requests temporary credentials for users from AWS, and then provides them to the user to access AWS resources.

The application verifies that employees are signed into the existing corporate network’s identity and authentication system, which might use LDAP, Active Directory, or another system. The identity broker application then obtains temporary security credentials for the employees.

To get temporary security credentials, the identity broker application calls either AssumeRole or GetFederationToken to obtain temporary security credentials, depending on how you want to manage the policies for users and when the temporary credentials should expire. The call returns temporary security credentials consisting of an AWS access key ID, a secret access key, and a session token. The identity broker application makes these temporary security credentials available to the internal company application. The app can then use the temporary credentials to make calls to AWS directly. The app caches the credentials until they expire, and then requests a new set of temporary credentials.



**Using an IAM policy that references the LDAP identifiers and AWS credentials** is incorrect because using an IAM policy is not enough to integrate your LDAP service to IAM. You need to use SAML, STS, or a custom identity broker.

**Using AWS Single Sign-On (SSO) service to enable single sign-on between AWS and your LDAP** is incorrect because the scenario did not require SSO and in addition, the identity store that you are using is not SAML-compatible.

**Using IAM roles to rotate the IAM credentials whenever LDAP credentials are updated** is incorrect because manually rotating the IAM credentials is not an optimal solution to integrate your on-premises and VPC network. You need to use SAML, STS, or a custom identity broker.

References:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_common-scenarios_federated-users.html>

<https://aws.amazon.com/blogs/aws/aws-identity-and-access-management-now-with-identity-federation/>

Tutorials Dojo’s AWS Certified Solutions Architect Associate Exam Study Guide:

<https://tutorialsdojo.com/aws-certified-solutions-architect-associate/>

**3. QUESTION**

A Solutions Architect is managing a company’s AWS account of approximately 300 IAM users. They have a new company policy that requires changing the associated permissions of all 100 IAM users that control the access to Amazon S3 buckets.

What will the Solutions Architect do to avoid the time-consuming task of applying the policy to each user?

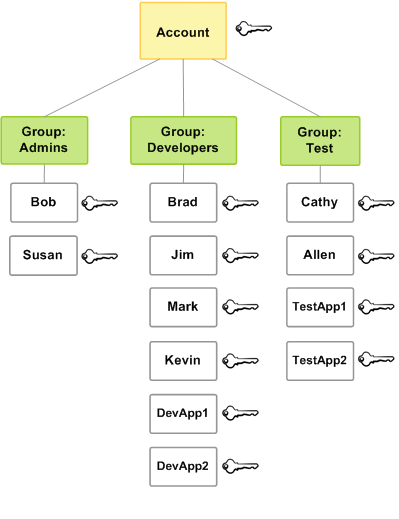
* **A.** Create a new policy and apply it to multiple IAM users using a shell script.
* **B.** Create a new IAM role and add each user to the IAM role.
* **C.** Create a new IAM group and then add the users that require access to the S3 bucket. Afterwards, apply the policy to IAM group. (Correct)
* **D.** Create a new S3 bucket access policy with unlimited access for each IAM user.

In this scenario, the best option is to **group the set of users in an IAM Group and then apply a policy with the required access to the Amazon S3 bucket**. This will enable you to easily add, remove, and manage the users instead of manually adding a policy to each and every 100 IAM users.

**Creating a new policy and applying it to multiple IAM users using a shell script** is incorrect because you need a new IAM Group for this scenario and not assign a policy to each user via a shell script. This method can save you time but afterward, it will be difficult to manage all 100 users that are not contained in an IAM Group.

**Creating a new S3 bucket access policy with unlimited access for each IAM user** is incorrect because you need a new IAM Group and the method is also time-consuming.

**Creating a new IAM role and adding each user to the IAM role** is incorrect because you need to use an IAM Group and not an IAM role.



Reference:

<http://docs.aws.amazon.com/IAM/latest/UserGuide/id_groups.html>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

**4. QUESTION**

An Intelligence Agency developed a missile tracking application that is hosted on both development and production AWS accounts. The Intelligence agency’s junior developer only has access to the development account. She has received security clearance to access the agency’s production account but the access is only temporary and only write access to EC2 and S3 is allowed.

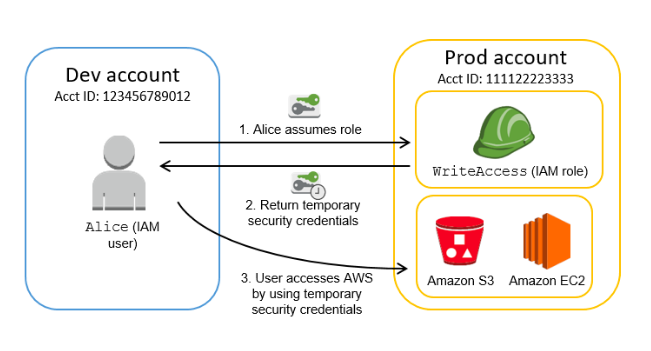
Which of the following allows you to issue short-lived access tokens that act as temporary security credentials to allow access to your AWS resources?

* **A.** All of the given options are correct.
* **B.** Use AWS IAM Identity Center
* **C.** Use AWS STS (Correct)
* **D.** Use AWS Cognito to issue JSON Web Tokens (JWT)

**AWS Security Token Service (AWS STS)** is the service that you can use to create and provide trusted users with temporary security credentials that can control access to your AWS resources. Temporary security credentials work almost identically to the long-term access key credentials that your IAM users can use.

In this diagram, IAM user Alice in the Dev account (the role-assuming account) needs to access the Prod account (the role-owning account). Here’s how it works:

1. Alice in the Dev account assumes an IAM role (WriteAccess) in the Prod account by calling AssumeRole.
2. STS returns a set of temporary security credentials.
3. Alice uses the temporary security credentials to access services and resources in the Prod account. Alice could, for example, make calls to Amazon S3 and Amazon EC2, which are granted by the WriteAccess role.



**Using AWS Cognito to issue JSON Web Tokens (JWT)** is incorrect because the Amazon Cognito service is primarily used for user authentication and not for providing access to your AWS resources. A JSON Web Token (JWT) is meant to be used for user authentication and session management.

**Using AWS AWS IAM Identity Center** is incorrect because this is simply a successor to the AWS Single Sign-On service that helps you securely create or connect your workforce identities and manage their access centrally across AWS accounts and applications. IAM Identity Center is the recommended approach for workforce authentication and authorization on AWS for organizations of any size and type, but not for generating tokens.

The option that says **All of the above** is incorrect as only STS has the ability to provide temporary security credentials.

References:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_temp.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_temp_request.html>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

**5. QUESTION**

A Solutions Architect created a brand new IAM User with a default setting using AWS CLI. This is intended to be used to send API requests to Amazon S3, DynamoDB, Lambda, and other AWS resources of the company’s cloud infrastructure.

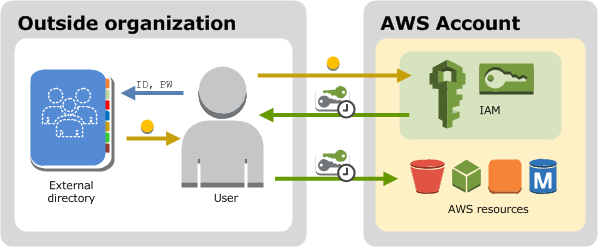
Which of the following must be done to allow the user to make API calls to the AWS resources?

* **A.** Create a set of Access Keys for the user and attach the necessary permissions. (Correct)
* **B.** Enable Multi-Factor Authentication for the user.
* **C.** Assign an IAM Policy to the user to allow it to send API calls.
* **D.** Do nothing as the IAM User is already capable of sending API calls to your AWS resources.

You can choose the credentials that are right for your IAM user. When you use the AWS Management Console to create a user, you must choose to include at least a console password or access keys. By default, a brand new IAM user created using the AWS CLI or AWS API has no credentials of any kind. You must create the type of credentials for an IAM user based on the needs of your user.

Access keys are long-term credentials for an IAM user or the AWS account root user. You can use access keys to sign programmatic requests to the AWS CLI or AWS API (directly or using the AWS SDK). Users need their own access keys to make programmatic calls to AWS from the AWS Command Line Interface (AWS CLI), Tools for Windows PowerShell, the AWS SDKs, or direct HTTP calls using the APIs for individual AWS services.

To fill this need, you can create, modify, view, or rotate access keys (access key IDs and secret access keys) for IAM users. When you create an access key, IAM returns the access key ID and secret access key. You should save these in a secure location and give them to the user.



The option that says: **Do nothing as the IAM User is already capable of sending API calls to your AWS resources** is incorrect because by default, a brand new IAM user created using the AWS CLI or AWS API has no credentials of any kind. Take note that in the scenario, you created the new IAM user using the AWS CLI and not via the AWS Management Console, where you must choose to at least include a console password or access keys when creating a new IAM user.

**Enabling Multi-Factor Authentication for the user** is incorrect because this will still not provide the required Access Keys needed to send API calls to your AWS resources. You have to grant the IAM user with Access Keys to meet the requirement.

**Assigning an IAM Policy to the user to allow it to send API calls** is incorrect because adding a new IAM policy to the new user will not grant the needed Access Keys needed to make API calls to the AWS resources.

References:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html#id_users_creds>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

**6. QUESTION**

A company has an application that continually sends encrypted documents to Amazon S3. The company requires that the configuration for data access is in line with their strict compliance standards. They should also be alerted if there is any risk of unauthorized access or suspicious access patterns.

Which step is needed to meet the requirements?

* **A.** Use Amazon Inspector to alert whenever a security violation is detected on S3.
* **B.** Use Amazon GuardDuty to monitor malicious activity on S3. (Correct)
* **C.** Use Amazon Macie to monitor and detect access patterns on S3.
* **D.** Use Amazon Rekognition to monitor and recognize patterns on S3.

Amazon GuardDuty can generate findings based on suspicious activities such as requests coming from known malicious IP addresses, changing of bucket policies/ACLs to expose an S3 bucket publicly, or suspicious API call patterns that attempt to discover misconfigured bucket permissions.



To detect possibly malicious behavior, GuardDuty uses a combination of anomaly detection, machine learning, and continuously updated threat intelligence.

Hence, the correct answer is:Use Amazon GuardDuty to monitor malicious activity on S3.

The option that says: **Use Amazon Rekognition to monitor and recognize patterns on S3** is incorrect because Amazon Rekognition is simply a service that can identify the objects, people, text, scenes, and activities on your images or videos, as well as detect any inappropriate content.

The option that says: **Use Amazon Macie to monitor and detect access patterns on S3** is incorrect because Macie cannot detect usage patterns on S3 data. While Amazon Macie is capable of detecting policy changes in S3 buckets, this is not enough to detect unauthorized or suspicious access patterns.

The option that says: Use Amazon Inspector to alert whenever a security violation is detected on S3is incorrect because Inspector is basically an automated security assessment service that helps improve the security and compliance of applications deployed on AWS.

References:

<https://aws.amazon.com/guardduty/>

<https://aws.amazon.com/blogs/aws/new-using-amazon-guardduty-to-protect-your-s3-buckets/>

Check out this Amazon GuardDuty Cheat Sheet:

<https://tutorialsdojo.com/amazon-guardduty/>

**7. QUESTION**

A tech company that you are working for has undertaken a Total Cost Of Ownership (TCO) analysis evaluating the use of Amazon S3 versus acquiring more storage hardware. The result was that all 1200 employees would be granted access to use Amazon S3 for the storage of their personal documents.

Which of the following will you need to consider so you can set up a solution that incorporates a single sign-on feature from your corporate AD or LDAP directory and also restricts access for each individual user to a designated user folder in an S3 bucket? (Select TWO.)

* **A.** Use 3rd party Single Sign-On solutions such as Atlassian Crowd, OKTA, OneLogin and many others.
* **B.** Set up a Federation proxy or an Identity provider, and use AWS Security Token Service to generate temporary tokens. (Correct)
* **C.** Set up a matching IAM user for each of the 1200 users in your corporate directory that needs access to a folder in the S3 bucket.
* **D.** Map each individual user to a designated user folder in S3 using Amazon WorkDocs to access their personal documents.
* **E.** Configure an IAM role and an IAM Policy to access the bucket. (Correct)

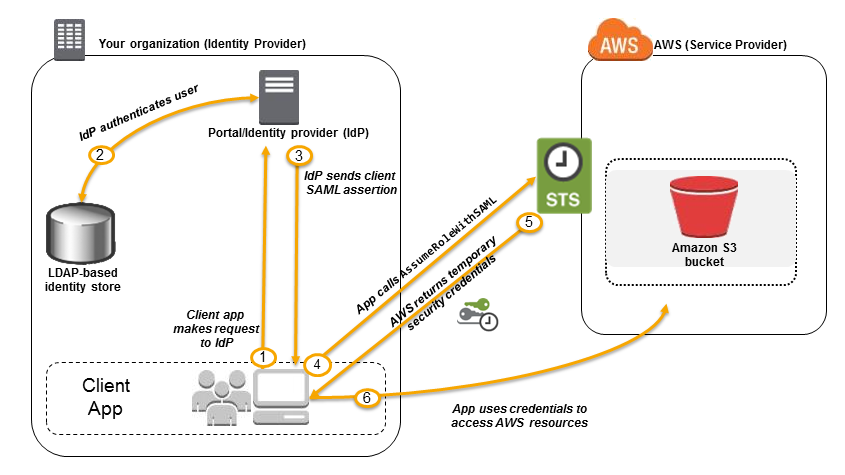
The question refers to one of the common scenarios for temporary credentials in AWS. Temporary credentials are useful in scenarios that involve identity federation, delegation, cross-account access, and IAM roles. In this example, it is called enterprise identity federation considering that you also need to set up a single sign-on (SSO) capability.

The correct answers are:

**– Setup a Federation proxy or an Identity provider**

**– Setup an AWS Security Token Service to generate temporary tokens**

**– Configure an IAM role and an IAM Policy to access the bucket.**



In an enterprise identity federation, you can authenticate users in your organization’s network, and then provide those users access to AWS without creating new AWS identities for them and requiring them to sign in with a separate user name and password. This is known as the *single sign-on* (SSO) approach to temporary access. AWS STS supports open standards like Security Assertion Markup Language (SAML) 2.0, with which you can use Microsoft AD FS to leverage your Microsoft Active Directory. You can also use SAML 2.0 to manage your own solution for federating user identities.

Using 3rd party Single Sign-On solutions such as Atlassian Crowd, OKTA, OneLogin and many others is incorrect since you don’t have to use 3rd party solutions to provide the access. AWS already provides the necessary tools that you can use in this situation.

**Mapping each individual user to a designated user folder in S3 using Amazon WorkDocs to access their personal documents** is incorrect as there is no direct way of integrating Amazon S3 with Amazon WorkDocs for this particular scenario. Amazon WorkDocs is simply a fully managed, secure content creation, storage, and collaboration service. With Amazon WorkDocs, you can easily create, edit, and share content. And because it’s stored centrally on AWS, you can access it from anywhere on any device.

**Setting up a matching IAM user for each of the 1200 users in your corporate directory that needs access to a folder in the S3 bucket** is incorrect since creating that many IAM users would be unnecessary. Also, you want the account to integrate with your AD or LDAP directory, hence, IAM Users does not fit these criteria.

References:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_saml.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_oidc.html>

<https://aws.amazon.com/premiumsupport/knowledge-center/iam-s3-user-specific-folder/>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

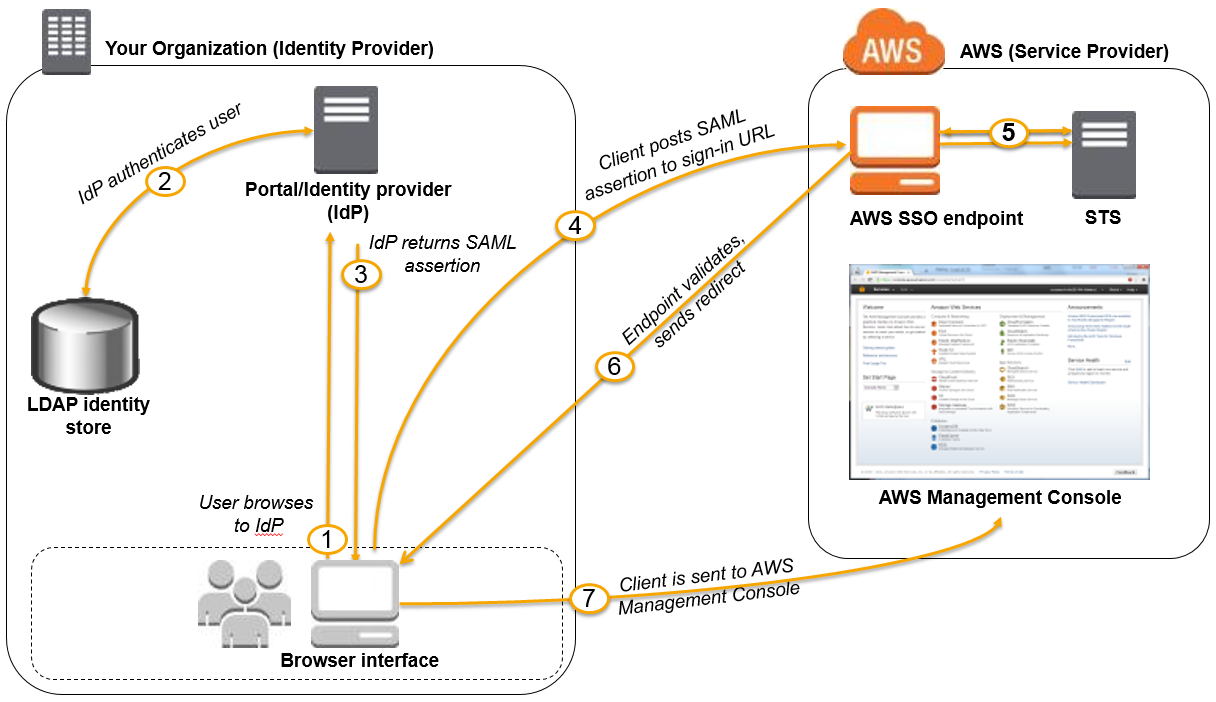
**8. QUESTION**

A pharmaceutical company has resources hosted on both their on-premises network and in AWS cloud. They want all of their Software Architects to access resources on both environments using their on-premises credentials, which is stored in Active Directory.

In this scenario, which of the following can be used to fulfill this requirement?

* **A.** Use IAM users
* **B.** Set up SAML 2.0-Based Federation by using a Microsoft Active Directory Federation Service (AD FS). (Correct)
* **C.** Set up SAML 2.0-Based Federation by using a Web Identity Federation.
* **D.** Use Amazon VPC

Since the company is using Microsoft Active Directory which implements Security Assertion Markup Language (SAML), you can set up a SAML-Based Federation for API Access to your AWS cloud. In this way, you can easily connect to AWS using the login credentials of your on-premises network.



AWS supports identity federation with SAML 2.0, an open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS APIs without you having to create an IAM user for everyone in your organization. By using SAML, you can simplify the process of configuring federation with AWS, because you can use the IdP’s service instead of writing custom identity proxy code.

Before you can use SAML 2.0-based federation as described in the preceding scenario and diagram, you must configure your organization’s IdP and your AWS account to trust each other. The general process for configuring this trust is described in the following steps. Inside your organization, you must have an IdP that supports SAML 2.0, like Microsoft Active Directory Federation Service (AD FS, part of Windows Server), Shibboleth, or another compatible SAML 2.0 provider.

Hence, the correct answer is: **Set up SAML 2.0-Based Federation by using a Microsoft Active Directory Federation Service (AD FS).**

**Setting up SAML 2.0-Based Federation by using a Web Identity Federation** is incorrect because this is primarily used to let users sign in via a well-known external identity provider (IdP), such as Login with Amazon, Facebook, Google. It does not utilize Active Directory.

**Using IAM users** is incorrect because the situation requires you to use the existing credentials stored in their Active Directory, and not user accounts that will be generated by IAM.

**Using Amazon VPC** is incorrect because this only lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define. This has nothing to do with user authentication or Active Directory.

References:

<http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_saml.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers.html>

Check out this AWS IAM Cheat Sheet:

<https://tutorialsdojo.com/aws-identity-and-access-management-iam/>

# **Virtual Private Cloud (VPC)**

**1. QUESTION:** An insurance company utilizes SAP HANA for its day-to-day ERP operations. Since they can’t migrate this database due to customer preferences, they need to integrate it with the current AWS workload in the VPC in which they are required to establish a site-to-site VPN connection.

What needs to be configured outside of the VPC for them to have a successful site-to-site VPN connection?

* **A.** An EIP to the Virtual Private Gateway
* **B.** An Internet-routable IP address (static) of the customer gateway's external interface for the on-premises network (Correct)
* **C.** A dedicated NAT instance in a public subnet
* **D.** The main route table in your VPC to route traffic through a NAT instance

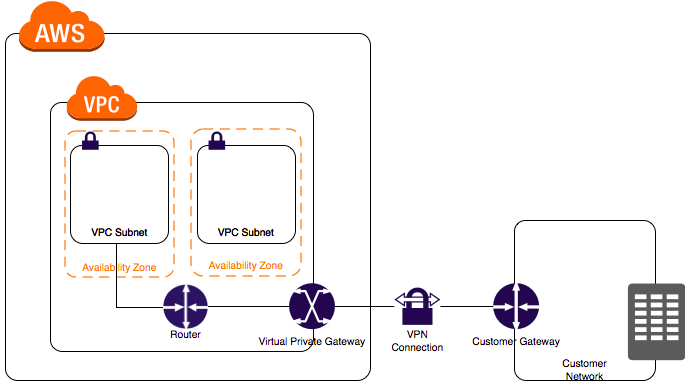
By default, instances that you launch into a virtual private cloud (VPC) can’t communicate with your own network. You can enable access to your network from your VPC by attaching a virtual private gateway to the VPC, creating a custom route table, updating your security group rules, and creating an AWS managed VPN connection.

Although the term VPN connection is a general term, in the Amazon VPC documentation, a VPN connection refers to the connection between your VPC and your own network. AWS supports Internet Protocol security (IPsec) VPN connections.

A customer gateway is a physical device or software application on your side of the VPN connection.

To create a VPN connection, you must create a customer gateway resource in AWS, which provides information to AWS about your customer gateway device. Next, you have to set up an Internet-routable IP address (static) of the customer gateway’s external interface.

The following diagram illustrates single VPN connections. The VPC has an attached virtual private gateway, and your remote network includes a customer gateway, which you must configure to enable the VPN connection. You set up the routing so that any traffic from the VPC bound for your network is routed to the virtual private gateway.



The options that say: **A dedicated NAT instance in a public subnet** and **the main route table in your VPC to route traffic through a NAT instance** are incorrect since you don’t need a NAT instance for you to be able to create a VPN connection.

**An EIP to the Virtual Private Gateway** is incorrect since you do not attach an EIP to a VPG.

References:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_VPN.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/SetUpVPNConnections.html>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**2. QUESTION**

A local bank has an in-house application that handles sensitive financial data in a private subnet. After the data is processed by the EC2 worker instances, they will be delivered to S3 for ingestion by other services.

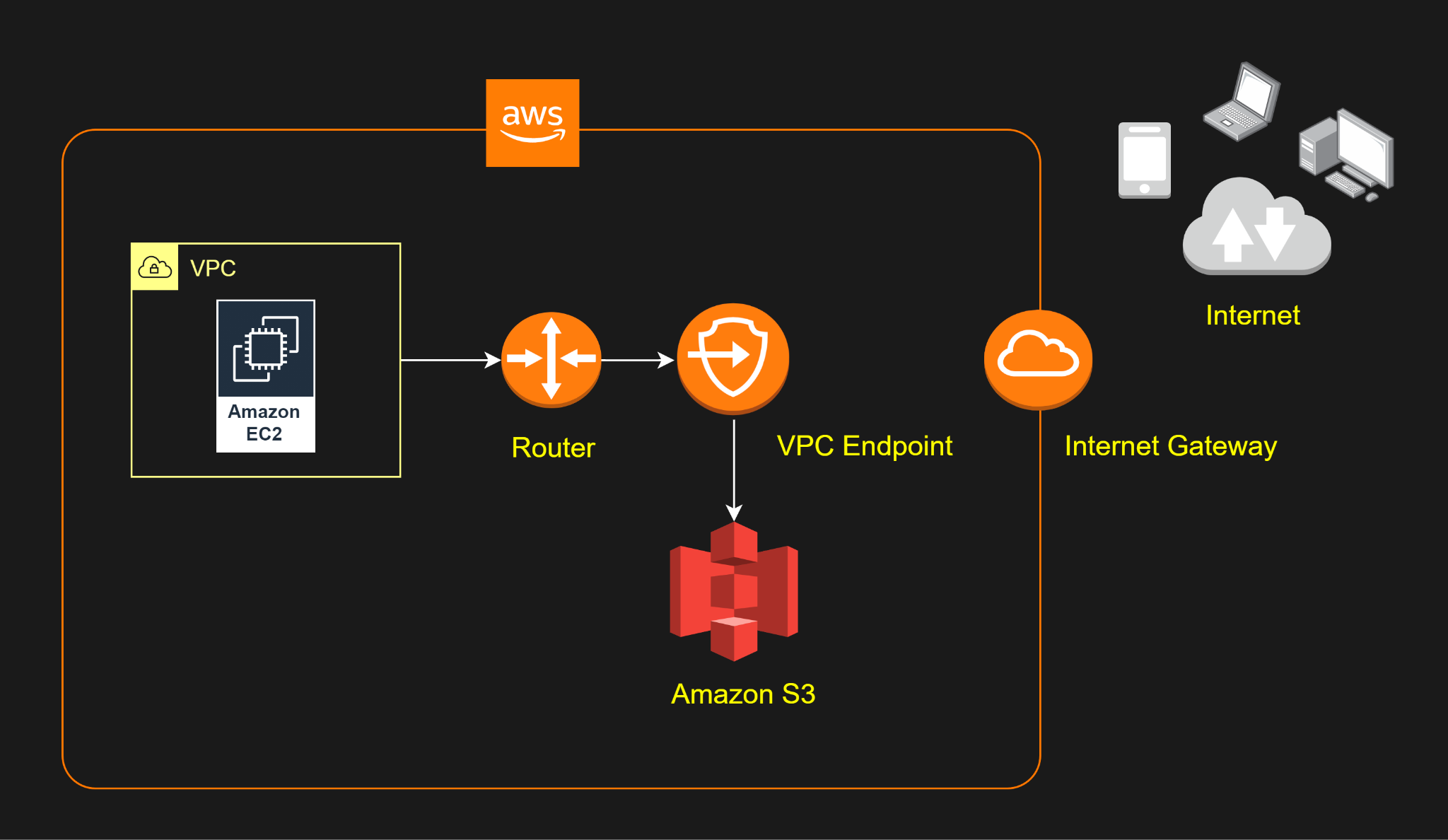
How should you design this solution so that the data does not pass through the public Internet?

* **A.** Configure a VPC Endpoint along with a corresponding route entry that directs the data to S3. (Correct)
* **B.** Configure a Transit gateway along with a corresponding route entry that directs the data to S3.
* **C.** Provision a NAT gateway in the private subnet with a corresponding route entry that directs the data to S3.
* **D.** Create an Internet gateway in the public subnet with a corresponding route entry that directs the data to S3.

The important concept that you have to understand in this scenario is that your VPC and your S3 bucket are located within the larger AWS network. However, the traffic coming from your VPC to your S3 bucket is traversing the public Internet by default. To better protect your data in transit, you can set up a VPC endpoint so the incoming traffic from your VPC will not pass through the public Internet, but instead through the private AWS network.

A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by PrivateLink without requiring an Internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other services does not leave the Amazon network.

Endpoints are virtual devices. They are horizontally scaled, redundant, and highly available VPC components that allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.



Hence, the correct answer is: **Configure a VPC Endpoint along with a corresponding route entry that directs the data to S3.**

The option that says: **Create an Internet gateway in the public subnet with a corresponding route entry that directs the data to S3** is incorrect because the Internet gateway is used for instances in the public subnet to have accessibility to the Internet.

The option that says: **Configure a Transit gateway along with a corresponding route entry that directs the data to S3** is incorrect because the Transit Gateway is used for interconnecting VPCs and on-premises networks through a central hub. Since Amazon S3 is outside of VPC, you still won’t be able to connect to it privately.

The option that says: **Provision a NAT gateway in the private subnet with a corresponding route entry that directs the data to S3** is incorrect because NAT Gateway allows instances in the private subnet to gain access to the Internet, but not vice versa.

References:

<https://docs.aws.amazon.com/vpc/latest/userguide/vpc-endpoints.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/vpce-gateway.html>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**3. QUESTION:** A web application is hosted on an EC2 instance that processes sensitive financial information which is launched in a private subnet. All of the data are stored in an Amazon S3 bucket. Financial information is accessed by users over the Internet. The security team of the company is concerned that the Internet connectivity to Amazon S3 is a security risk.

In this scenario, what will you do to resolve this security vulnerability in the most cost-effective manner?

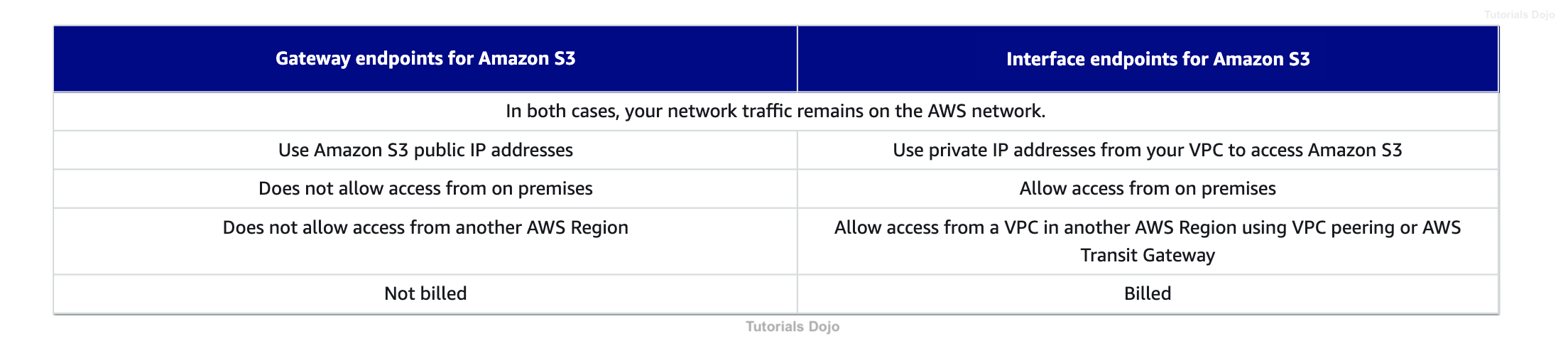
* **A.** Change the web architecture to access the financial data through a Gateway VPC Endpoint. (Correct)
* **B.** Change the web architecture to access the financial data in S3 through an interface VPC endpoint, which is powered by AWS PrivateLink.
* **C.** Change the web architecture to access the financial data hosted in your S3 bucket by creating a custom VPC endpoint service.
* **D.** Change the web architecture to access the financial data in your S3 bucket through a VPN connection.

Take note that your VPC lives within a larger AWS network and the services, such as S3, DynamoDB, RDS, and many others, are located outside of your VPC, but still within the AWS network. By default, the connection that your VPC uses to connect to your S3 bucket or any other service traverses the public Internet via your Internet Gateway.

A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by PrivateLink without requiring an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other service does not leave the Amazon network.

There are two types of VPC endpoints: *interface endpoints* and *gateway endpoints*. You have to create the type of VPC endpoint required by the supported service.

An interface endpoint is an elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported service. A gateway endpoint is a gateway that is a target for a specified route in your route table, used for traffic destined to a supported AWS service.



Hence, the correct answer is: **Change the web architecture to access the financial data through a Gateway VPC Endpoint.**

The option that says: **Changing the web architecture to access the financial data in your S3 bucket through a VPN connection** is incorrect because a VPN connection still goes through the public Internet. You have to use a VPC Endpoint in this scenario and not VPN, to privately connect your VPC to supported AWS services such as S3.

The option that says: **Changing the web architecture to access the financial data hosted in your S3 bucket by creating a custom VPC endpoint service** is incorrect because a “VPC endpoint service” is quite different from a “VPC endpoint”. With the VPC endpoint service, you are the service provider where you can create your own application in your VPC and configure it as an AWS PrivateLink-powered service (referred to as an endpoint service). Other AWS principals can create a connection from their VPC to your endpoint service using an interface VPC endpoint.

The option that says: **Changing the web architecture to access the financial data in S3 through an interface VPC endpoint, which is powered by AWS PrivateLink** is incorrect. Although you can use an Interface VPC Endpoint to satisfy the requirement, this type entails an associated cost, unlike a Gateway VPC Endpoint. Remember that you won’t get billed if you use a Gateway VPC endpoint for your Amazon S3 bucket, unlike an Interface VPC endpoint that is billed for hourly usage and data processing charges. Take note that the scenario explicitly asks for the most cost-effective solution.

References:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-endpoints.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/vpce-gateway.html>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**4. QUESTION**

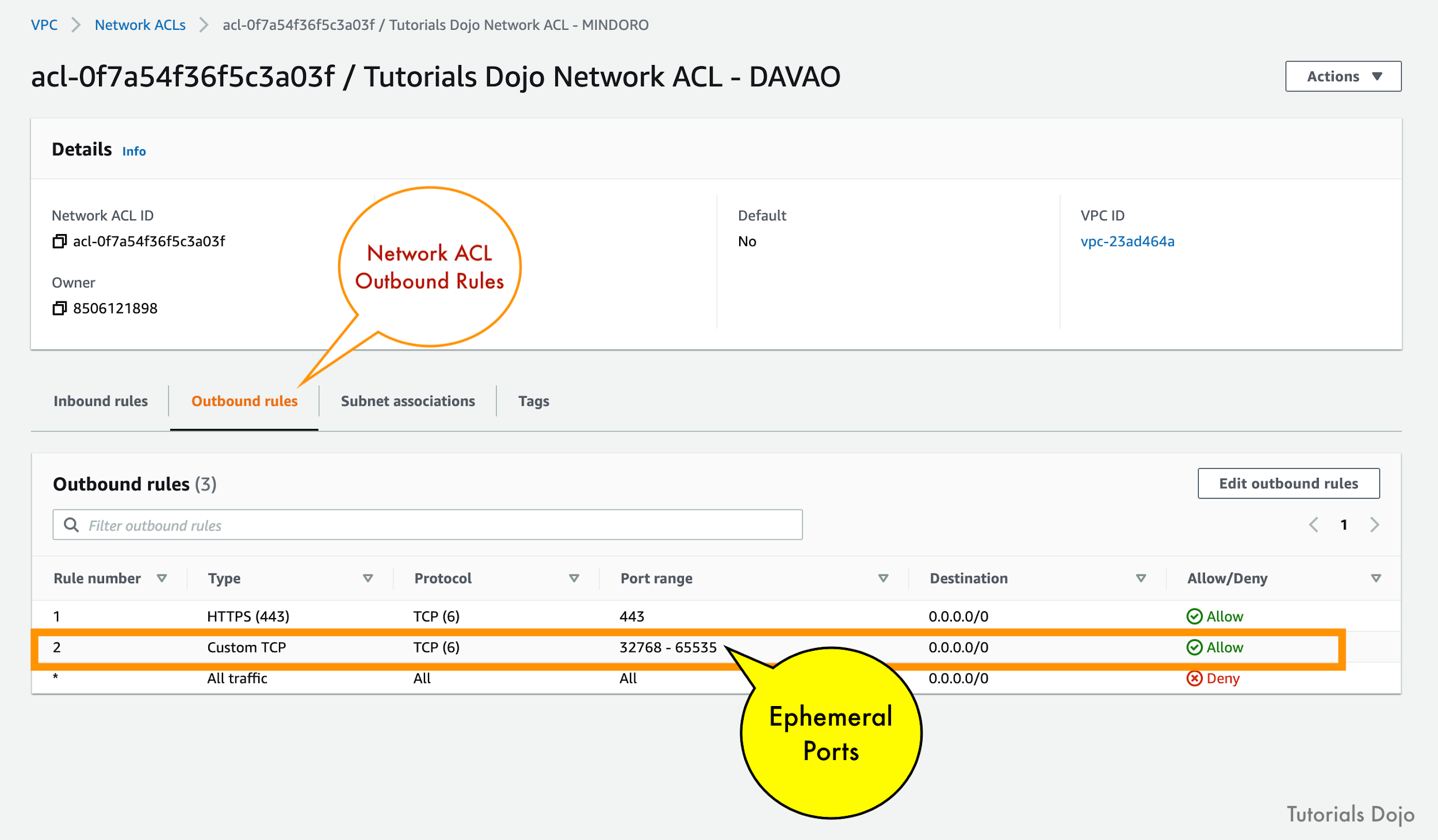
A company hosted a web application on a Linux Amazon EC2 instance in the public subnet that uses a non-default network ACL. The instance uses a default security group and has an attached Elastic IP address. The network ACL is configured to block all inbound and outbound traffic. The Solutions Architect must allow incoming traffic on port 443 to access the application from any source.

Which combination of steps will accomplish this requirement? (Select TWO.)

* **A.** In the Security Group, add a new rule to allow TCP connection on port 443 from source 0.0.0.0/0 (Correct)
* **B.** In the Network ACL, update the rule to allow outbound TCP connection on port 32768 - 65535 to destination 0.0.0.0/0
* **C.** In the Network ACL, update the rule to allow inbound TCP connection on port 443 from source 0.0.0.0/0 and outbound TCP connection on port 32768 - 65535 to destination 0.0.0.0/0 (Correct)
* **D.** In the Network ACL, update the rule to allow both inbound and outbound TCP connection on port 443 from source 0.0.0.0/0 and to destination 0.0.0.0/0
* **E.** In the Security Group, create a new rule to allow TCP connection on port 443 to destination 0.0.0.0/0

To enable the connection to a service running on an instance, the associated network ACL must allow both inbound traffic on the port that the service is listening on as well as outbound traffic from ephemeral ports. When a client connects to a server, a random port from the ephemeral port range (1024-65535) becomes the client’s source port.

The designated ephemeral port then becomes the destination port for return traffic from the service, so outbound traffic from the ephemeral port must be allowed in the network ACL. By default, network ACLs allow all inbound and outbound traffic. If your network ACL is more restrictive, then you need to explicitly allow traffic from the ephemeral port range.



The client that initiates the request chooses the ephemeral port range. The range varies depending on the client’s operating system.

– Many Linux kernels (including the Amazon Linux kernel) use ports 32768-61000.

– Requests originating from Elastic Load Balancing use ports 1024-65535.

– Windows operating systems through Windows Server 2003 use ports 1025-5000.

– Windows Server 2008 and later versions use ports 49152-65535.

– A NAT gateway uses ports 1024-65535.

– AWS Lambda functions use ports 1024-65535.

For example, if a request comes into a web server in your VPC from a Windows 10 client on the Internet, your network ACL must have an outbound rule to enable traffic destined for ports 49152 – 65535. If an instance in your VPC is the client initiating a request, your network ACL must have an inbound rule to enable traffic destined for the ephemeral ports specific to the type of instance (Amazon Linux, Windows Server 2008, and so on).

In this scenario, you only need to allow the incoming traffic on port 443. Since security groups are stateful, you can apply any changes to an incoming rule and it will be automatically applied to the outgoing rule.

To enable the connection to a service running on an instance, the associated network ACL must allow both inbound traffic on the port that the service is listening on as well as outbound traffic from ephemeral ports. When a client connects to a server, a random port from the ephemeral port range (32768 – 65535) becomes the client’s source port. Since the return traffic will use an ephemeral port, outbound traffic must be allowed on these ports to destination 0.0.0.0/0.

Hence, the correct answers are:

**– In the Security Group, add a new rule to allow TCP connection on port 443 from source 0.0.0.0/0.**

**– In the Network ACL, update the rule to allow inbound TCP connection on port 443 from source 0.0.0.0/0 and outbound TCP connection on port 32768 - 65535 to destination 0.0.0.0/0.**

The option that says: **In the Security Group, create a new rule to allow TCP connection on port 443 to destination 0.0.0.0/0** is incorrect because this step just allows outbound connections from the EC2 instance out to the public Internet, which is unnecessary. Remember that a default security group already includes an outbound rule that allows all outbound traffic.

The option that says: **In the Network ACL, update the rule to allow both inbound and outbound TCP connection on port 443 from source 0.0.0.0/0 and to destination 0.0.0.0/0** is incorrect because your network ACL must have an outbound rule to allow ephemeral ports (32768 - 65535). These are the specific ports that will be used as the client’s source port for the traffic response.

The option that says: **In the Network ACL, update the rule to allow outbound TCP connection on port 32768 - 65535 to destination 0.0.0.0/0** is incorrect because this step is just partially right. You still need to add an inbound rule from port 443 and not just the outbound rule for the ephemeral ports (32768 - 65535).

References:

<https://aws.amazon.com/premiumsupport/knowledge-center/connect-http-https-ec2/>

<https://docs.amazonaws.cn/en_us/vpc/latest/userguide/vpc-network-acls.html#nacl-ephemeral-ports>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**5. QUESTION**

A media company has two VPCs: VPC-1 and VPC-2 with peering connection between each other. VPC-1 only contains private subnets while VPC-2 only contains public subnets. The company uses a single AWS Direct Connect connection and a virtual interface to connect their on-premises network with VPC-1.

Which of the following options increase the fault tolerance of the connection to VPC-1? (Select TWO.)

* **A.** Establish a new AWS Direct Connect connection and private virtual interface in the same region as VPC-2.
* **B.** Use the AWS VPN CloudHub to create a new AWS Direct Connect connection and private virtual interface in the same region as VPC-2.
* **C.** Establish a hardware VPN over the Internet between VPC-2 and the on-premises network.
* **D.**Establish another AWS Direct Connect connection and private virtual interface in the same AWS region as VPC-1. (Correct)
* **E.** Establish a hardware VPN over the Internet between VPC-1 and the on-premises network. (Correct)

In this scenario, you have two VPCs that have peering connections with each other. Note that a VPC peering connection does not support edge-to-edge routing. This means that if either VPC in a peering relationship has one of the following connections, you cannot extend the peering relationship to that connection:

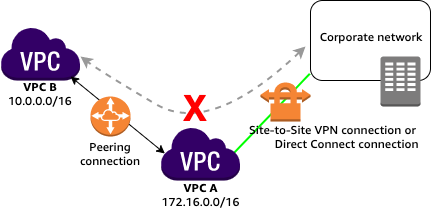
– A VPN connection or an AWS Direct Connect connection to a corporate network

– An Internet connection through an Internet gateway

– An Internet connection in a private subnet through a NAT device

– A gateway VPC endpoint to an AWS service; for example, an endpoint to Amazon S3.

– (IPv6) A ClassicLink connection. You can enable IPv4 communication between a linked EC2-Classic instance and instances in a VPC on the other side of a VPC peering connection. However, IPv6 is not supported in EC2-Classic, so you cannot extend this connection for IPv6 communication.

  
For example, if VPC A and VPC B are peered, and VPC A has any of these connections, then instances in VPC B cannot use the connection to access resources on the other side of the connection. Similarly, resources on the other side of a connection cannot use the connection to access VPC B.

Hence, this implies that VPC-2 cannot extend the peering relationship between VPC-1 and the on-premises network. In other words, traffic originating from the corporate network cannot establish a direct connection to VPC-1 by routing it through VPC-2, whether using a VPN connection or an AWS Direct Connect connection. The VPC peering connection is a one-to-one relationship with directly peered entities (VPC-1 and the on-premises network), and it does not support transitive communication through intermediate VPCs like VPC-2, which is why the following options are incorrect:

**– Use the AWS VPN CloudHub to create a new AWS Direct Connect connection and private virtual interface in the same region as VPC-2.**

**– Establish a hardware VPN over the Internet between VPC-2 and the on-premises network.**

**– Establish a new AWS Direct Connect connection and private virtual interface in the same region as VPC-2.**

You can do the following to provide a highly available, fault-tolerant network connection:

**– Establish a hardware VPN over the Internet between the VPC and the on-premises network.**

**– Establish another AWS Direct Connect connection and private virtual interface in the same AWS region as VPC-1.**

References:

<https://docs.aws.amazon.com/vpc/latest/peering/invalid-peering-configurations.html#edge-to-edge-vgw>

<https://docs.aws.amazon.com/whitepapers/latest/hybrid-connectivity/vpn-connection-as-a-backup-to-aws-dx-connection-example.html>

<https://aws.amazon.com/answers/networking/aws-multiple-data-center-ha-network-connectivity/>

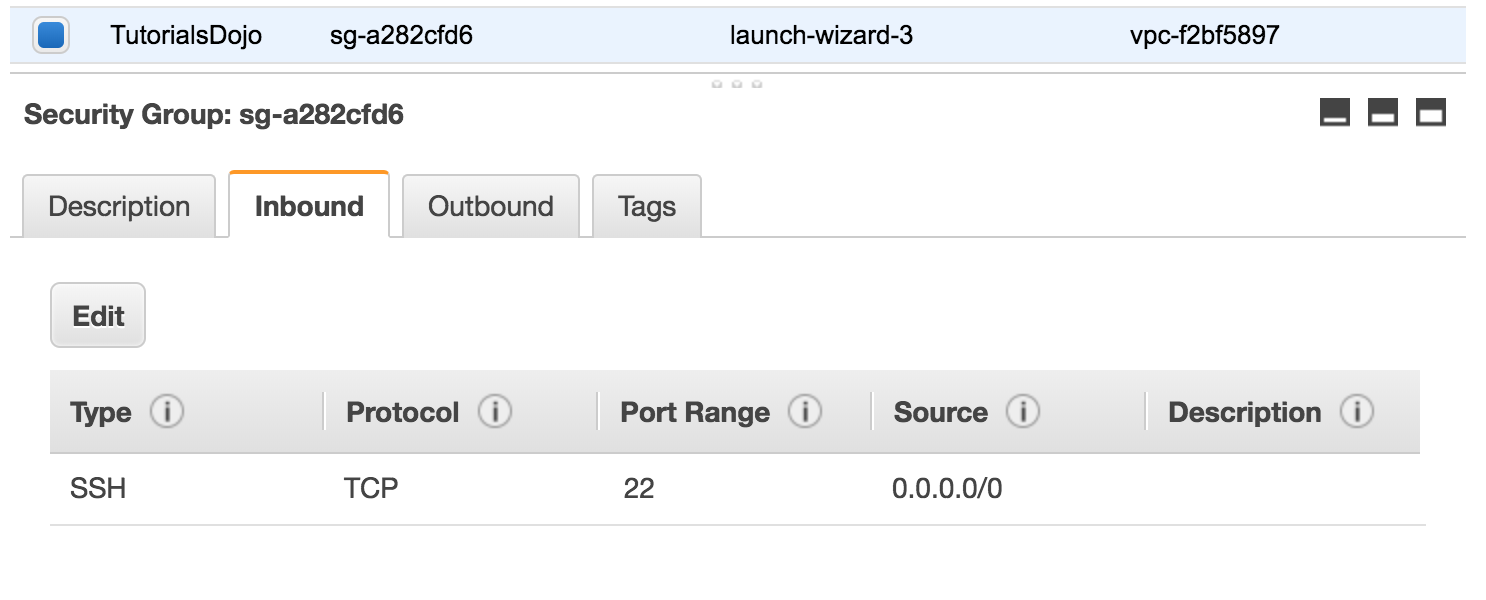
Check out these Amazon VPC and AWS Direct Connect Cheat Sheets:

<https://tutorialsdojo.com/amazon-vpc/>

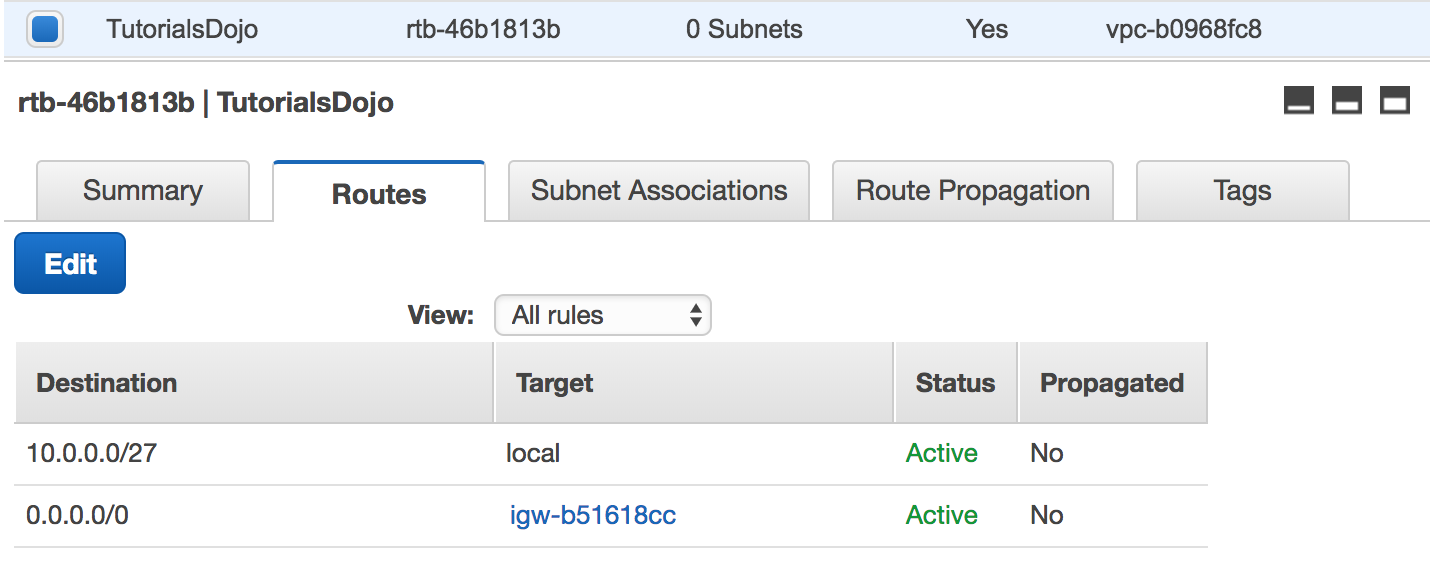
<https://tutorialsdojo.com/aws-direct-connect/>

**6. QUESTION**

A company has an On-Demand EC2 instance located in a subnet in AWS that hosts a web application. The security group attached to this EC2 instance has the following Inbound Rules:

​

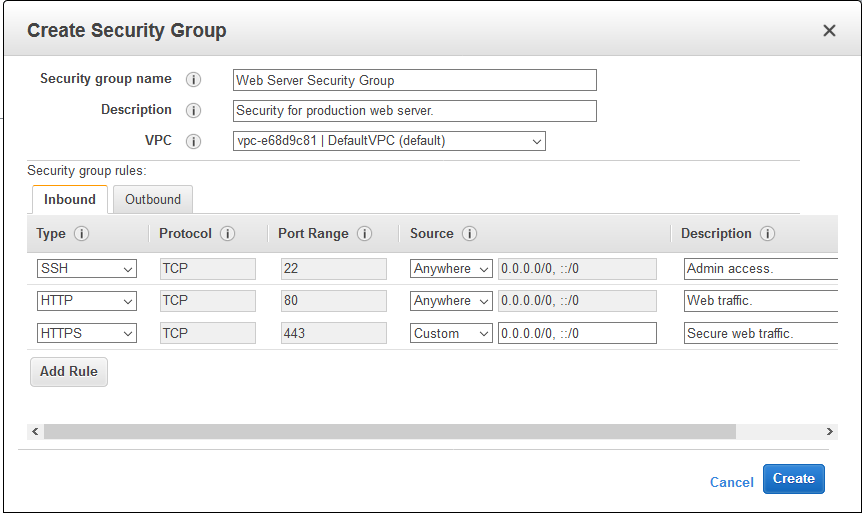
The Route table attached to the VPC is shown below. You can establish an SSH connection into the EC2 instance from the Internet. However, you are not able to connect to the web server using your Chrome browser.

​

Which of the below steps would resolve the issue?

* **A.** In the Route table, add this new route entry: 10.0.0.0/27 -> local
* **B.** In the Route table, add this new route entry: 0.0.0.0 -> igw-b51618cc
* **C.** In the Security Group, remove the SSH rule.
* **D.** In the Security Group, add an Inbound HTTP rule. (Correct)

In this particular scenario, you can already connect to the EC2 instance via SSH. This means that there is no problem in the Route Table of your VPC. To fix this issue, you simply need to update your Security Group and add an Inbound rule to allow HTTP traffic.



The option that says: **In the Security Group, remove the SSH rule** is incorrect as doing so will not solve the issue. It will just disable SSH traffic that is already available.

The options that say: **In the Route table, add this new route entry: 0.0.0.0 -> igw-b51618cc** and **In the Route table, add this new route entry: 10.0.0.0/27 -> local** are incorrect as there is no need to change the Route Tables.

Reference:

<http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

**7. QUESTION**

A large insurance company has an AWS account that contains three VPCs (DEV, UAT and PROD) in the same region. UAT is peered to both PROD and DEV using a VPC peering connection. All VPCs have non-overlapping CIDR blocks. The company wants to push minor code releases from Dev to Prod to speed up time to market.

Which of the following options helps the company accomplish this?

* Do nothing. Since these two VPCs are already connected via UAT, they already have a connection to each other.
* Create a new VPC peering connection between PROD and DEV with the appropriate routes. (Correct)
* Create a new entry to PROD in the DEV route table using the VPC peering connection as the target.
* Change the DEV and PROD VPCs to have overlapping CIDR blocks to be able to connect them.

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region.



AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection and does not rely on a separate piece of physical hardware. There is no single point of failure for communication or a bandwidth bottleneck.

**Creating a new entry to PROD in the DEV route table using the VPC peering connection as the target** is incorrect because even if you configure the route tables, the two VPCs will still be disconnected until you set up a VPC peering connection between them.

**Changing the DEV and PROD VPCs to have overlapping CIDR blocks to be able to connect them** is incorrect because you cannot peer two VPCs with overlapping CIDR blocks.

The option that says: **Do nothing. Since these two VPCs are already connected via UAT, they already have a connection to each other** is incorrect as transitive VPC peering is not allowed hence, even though DEV and PROD are both connected in UAT, these two VPCs do not have a direct connection to each other.

Reference:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-peering.html>

Check out these Amazon VPC and VPC Peering Cheat Sheets:

<https://tutorialsdojo.com/amazon-vpc/>

<https://tutorialsdojo.com/vpc-peering/>

**8. QUESTION**

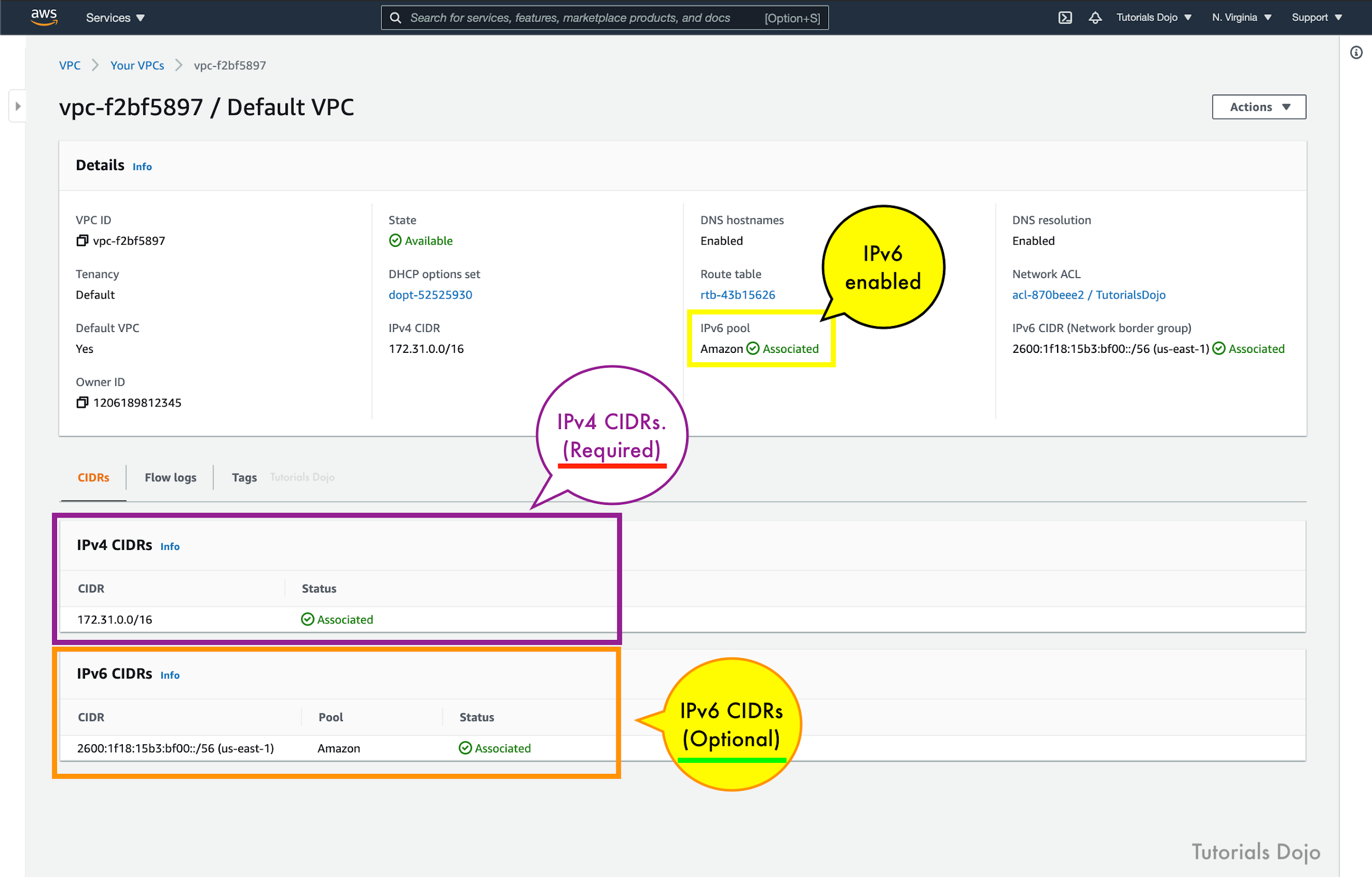
A company has multiple VPCs with IPv6 enabled for its suite of web applications. The Solutions Architect tried to deploy a new Amazon EC2 instance but she received an error saying that there is no IP address available on the subnet.

How should the Solutions Architect resolve this problem?

* **A.** Set up a new IPv4 subnet with a larger CIDR range. Associate the new subnet with the VPC and then launch the instance. (Correct)
* **B.** Disable the IPv4 support in the VPC and use the available IPv6 addresses.
* **C.** Ensure that the VPC has IPv6 CIDRs only. Remove any IPv4 CIDRs associated with the VPC.
* **D.** Set up a new IPv6-only subnet with a large CIDR range. Associate the new subnet with the VPC then launch the instance.

Amazon Virtual Private Cloud (VPC) is a service that lets you launch AWS resources in a logically isolated virtual network that you define. You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways. You can use both IPv4 and IPv6 for most resources in your virtual private cloud, helping to ensure secure and easy access to resources and applications.

A subnet is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. When you create a VPC, you must specify a range of IPv4 addresses for the VPC in the form of a CIDR block. Each subnet must reside entirely within one Availability Zone and cannot span zones. You can also optionally assign an IPv6 CIDR block to your VPC, and assign IPv6 CIDR blocks to your subnets.



If you have an existing VPC that supports IPv4 only and resources in your subnet that are configured to use IPv4 only, you can enable IPv6 support for your VPC and resources. Your VPC can operate in dual-stack mode — your resources can communicate over IPv4, or IPv6, or both. IPv4 and IPv6 communication are independent of each other. You cannot disable IPv4 support for your VPC and subnets since this is the default IP addressing system for Amazon VPC and Amazon EC2.

By default, a new EC2 instance uses an IPv4 addressing protocol. To fix the problem in the scenario, you need to create a new IPv4 subnet and deploy the EC2 instance in the new subnet.

Hence, the correct answer is: **Set up a new IPv4 subnet with a larger CIDR range. Associate the new subnet with the VPC and then launch the instance.**

The option that says: **Set up a new IPv6-only subnet with a large CIDR range. Associate the new subnet with the VPC then launch the instance** is incorrect because you need to add IPv4 subnet first before you can create an IPv6 subnet.

The option that says: **Ensure that the VPC has IPv6 CIDRs only. Remove any IPv4 CIDRs associated with the VPC** is incorrect because you can’t have a VPC with IPv6 CIDRs only. The default IP addressing system in VPC is IPv4. You can only change your VPC to dual-stack mode where your resources can communicate over IPv4, or IPv6, or both, but not exclusively with IPv6 only.

The option that says: **Disable the IPv4 support in the VPC and use the available IPv6 addresses** is incorrect because you cannot disable the IPv4 support for your VPC and subnets since this is the default IP addressing system.

References:

<https://docs.aws.amazon.com/vpc/latest/userguide/vpc-migrate-ipv6.html>

<https://docs.aws.amazon.com/vpc/latest/userguide/vpc-ip-addressing.html>

<https://aws.amazon.com/vpc/faqs/>

Check out this Amazon VPC Cheat Sheet:

<https://tutorialsdojo.com/amazon-vpc/>

# **Simple Storage Service S3**

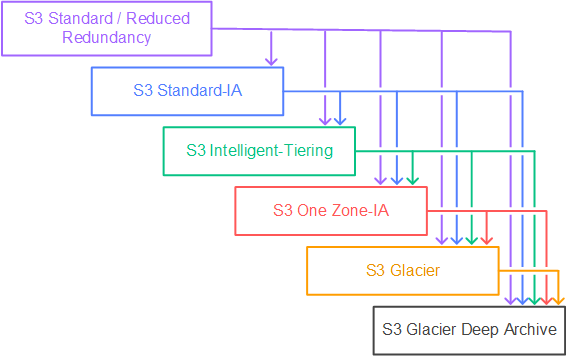
**1. QUESTION**

There are a few, easily reproducible but confidential files that your client wants to store in AWS without worrying about storage capacity. For the first month, all of these files will be accessed frequently but after that, they will rarely be accessed at all. The old files will only be accessed by developers so there is no set retrieval time requirement. However, the files under a specific alx-finance prefix in the S3 bucket will be used for post-processing that requires millisecond retrieval time.

Given these conditions, which of the following options would be the most cost-effective solution for your client’s storage needs?

* **A.** Store the files in S3 then after a month, change the storage class of the bucket to S3-IA using lifecycle policy.
* **B.** Store the files in S3 then after a month, change the storage class of the bucket to Intelligent-Tiering using lifecycle policy.
* **C.** Store the files in S3 then after a month, change the storage class of the alx-finance prefix to S3-IA while the remaining go to Glacier using lifecycle policy.
* **D.** Store the files in S3 then after a month, change the storage class of the alx-finance prefix to One Zone-IA while the remaining go to Glacier using lifecycle policy. (Correct)

Initially, the files will be accessed frequently, and S3 is a durable and highly available storage solution for that. After a month has passed, the files won’t be accessed frequently anymore, so it is a good idea to use lifecycle policies to move them to a storage class that would have a lower cost for storing them.



Since the files are easily reproducible and some of them are needed to be retrieved quickly based on a specific prefix filter (alx-finance), S3-One Zone IA would be a good choice for storing them. The other files that do not contain such prefix would then be moved to Glacier for low-cost archival. This setup would also be the most cost-effective for the client.

Hence, the correct answer is: **Store the files in S3 then after a month, change the storage class of the alx-finance prefix to One Zone-IA while the remaining go to Glacier using lifecycle policy**.

The option that says: **Storing the files in S3 then after a month, changing the storage class of the bucket to S3-IA using lifecycle policy** is incorrect. Although it is valid to move the files to S3-IA, this solution still costs more compared with using a combination of S3-One Zone IA and Glacier.

The option that says: **Storing the files in S3 then after a month, changing the storage class of the bucket to Intelligent-Tiering using lifecycle policy** is incorrect. While S3 Intelligent-Tiering can automatically move data between two access tiers (frequent access and infrequent access) when access patterns change, it is more suitable for scenarios where you don’t know the access patterns of your data. It may take some time for S3 Intelligent-Tiering to analyze the access patterns before it moves the data to a cheaper storage class like S3-IA which means you may still end up paying more in the beginning. In addition, you already know the access patterns of the files which means you can directly change the storage class immediately and save cost right away.

The option that says: **Storing the files in S3 then after a month, changing the storage class of the alx-finance prefix to S3-IA while the remaining go to Glacier using lifecycle policy** is incorrect. Even though S3-IA costs less than the S3 Standard storage class, it is still more expensive than S3-One Zone IA. Remember that the files are easily reproducible so you can safely move the data to S3-One Zone IA and in case there is an outage, you can simply generate the missing data again.

References:

<https://aws.amazon.com/blogs/compute/amazon-s3-adds-prefix-and-suffix-filters-for-lambda-function-triggering>

<https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html>

<https://docs.aws.amazon.com/AmazonS3/latest/dev/lifecycle-configuration-examples.html>

<https://aws.amazon.com/s3/pricing>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

**2. QUESTION**

An online medical system hosted in AWS stores sensitive Personally Identifiable Information (PII) of the users in an Amazon S3 bucket. Both the master keys and the unencrypted data should never be sent to AWS to comply with the strict compliance and regulatory requirements of the company.

Which S3 encryption technique should the Architect use?

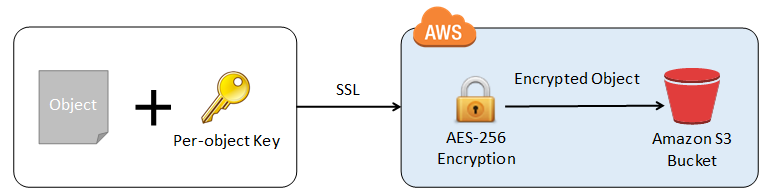
* **A.** Use S3 client-side encryption with a KMS-managed customer master key.
* **B.** Use S3 client-side encryption with a client-side master key. (Correct)
* **C.** Use S3 server-side encryption with a KMS managed key.
* **D.** Use S3 server-side encryption with customer provided key.

Client-side encryption is the act of encrypting data before sending it to Amazon S3. To enable client-side encryption, you have the following options:

– Use an AWS KMS-managed customer master key.

– Use a client-side master key.

When using an AWS KMS-managed customer master key to enable client-side data encryption, you provide an AWS KMS customer master key ID (CMK ID) to AWS. On the other hand, when you use client-side master key for client-side data encryption, **your client-side master keys and your unencrypted data are never sent to AWS**. It’s important that you safely manage your encryption keys because if you lose them, you can’t decrypt your data.



This is how client-side encryption using client-side master key works:

When uploading an object – You provide a client-side master key to the Amazon S3 encryption client. The client uses the master key only to encrypt the data encryption key that it generates randomly. The process works like this:

1. The Amazon S3 encryption client generates a one-time-use symmetric key (also known as a data encryption key or data key) locally. It uses the data key to encrypt the data of a single Amazon S3 object. The client generates a separate data key for each object.

2. The client encrypts the data encryption key using the master key that you provide. The client uploads the encrypted data key and its material description as part of the object metadata. The client uses the material description to determine which client-side master key to use for decryption.

3. The client uploads the encrypted data to Amazon S3 and saves the encrypted data key as object metadata (x-amz-meta-x-amz-key) in Amazon S3.

When downloading an object – The client downloads the encrypted object from Amazon S3. Using the material description from the object’s metadata, the client determines which master key to use to decrypt the data key. The client uses that master key to decrypt the data key and then uses the data key to decrypt the object.

Hence, the correct answer is to **use S3 client-side encryption with a client-side master key**.

**Using S3 client-side encryption with a KMS-managed customer master key** is incorrect because in client-side encryption with a KMS-managed customer master key, you provide an AWS KMS customer master key ID (CMK ID) to AWS. The scenario clearly indicates that both the master keys and the unencrypted data should never be sent to AWS.

**Using S3 server-side encryption with a KMS managed key** is incorrect because the scenario mentioned that the unencrypted data should never be sent to AWS, which means that you have to use client-side encryption in order to encrypt the data first before sending to AWS. In this way, you can ensure that there is no unencrypted data being uploaded to AWS. In addition, the master key used by Server-Side Encryption with AWS KMS–Managed Keys (SSE-KMS) is uploaded and managed by AWS, which directly violates the requirement of not uploading the master key.

**Using S3 server-side encryption with customer provided key** is incorrect because just as mentioned above, you have to use client-side encryption in this scenario instead of server-side encryption. For the S3 server-side encryption with customer-provided key (SSE-C), you actually provide the encryption key as part of your request to upload the object to S3. Using this key, Amazon S3 manages both the encryption (as it writes to disks) and decryption (when you access your objects).

References:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html>

<https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html>

**3. QUESTION**

There was an incident in your production environment where the user data stored in the S3 bucket has been accidentally deleted by one of the Junior DevOps Engineers. The issue was escalated to your manager and after a few days, you were instructed to improve the security and protection of your AWS resources.

What combination of the following options will protect the S3 objects in your bucket from both accidental deletion and overwriting? (Select TWO.)

* **A.** Enable Versioning (Correct)
* **B.** Provide access to S3 data strictly through pre-signed URL only
* **C.** Disallow S3 Delete using an IAM bucket policy
* **D.** Enable Amazon S3 Intelligent-Tiering
* **E.** Enable Multi-Factor Authentication Delete (Correct)

By using Versioning and enabling MFA (Multi-Factor Authentication) Delete, you can secure and recover your S3 objects from accidental deletion or overwrite.

Versioning is a means of keeping multiple variants of an object in the same bucket. Versioning-enabled buckets enable you to recover objects from accidental deletion or overwrite. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures.

You can also optionally add another layer of security by configuring a bucket to enable MFA (Multi-Factor Authentication) Delete, which requires additional authentication for either of the following operations:

– Change the versioning state of your bucket

– Permanently delete an object version

MFA Delete requires two forms of authentication together:

– Your security credentials

– The concatenation of a valid serial number, a space, and the six-digit code displayed on an approved authentication device

**Providing access to S3 data strictly through pre-signed URL only** is incorrect since a pre-signed URL gives access to the object identified in the URL. Pre-signed URLs are useful when customers perform an object upload to your S3 bucket, but does not help in preventing accidental deletes.

**Disallowing S3 Delete using an IAM bucket policy** is incorrect since you still want users to be able to delete objects in the bucket, and you just want to prevent accidental deletions. Disallowing S3 Delete using an IAM bucket policy will restrict all delete operations to your bucket.

**Enabling Amazon S3 Intelligent-Tiering** is incorrect since S3 intelligent tiering does not help in this situation.

Reference:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

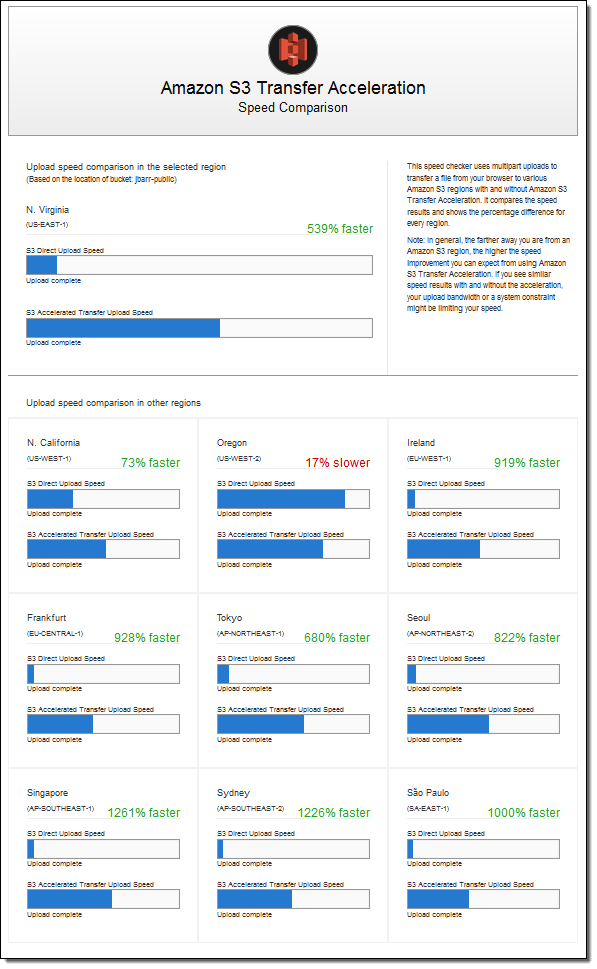
**4. QUESTION**

A company collects atmospheric data such as temperature, air pressure, and humidity from different countries. Each site location is equipped with various weather instruments and a high-speed Internet connection. The average collected data in each location is around 500 GB and will be analyzed by a weather forecasting application hosted in Northern Virginia. As the Solutions Architect, you need to aggregate all the data in the fastest way.

Which of the following options can satisfy the given requirement?

* **A.** Set up a Site-to-Site VPN connection.
* **B.** Use AWS Snowball Edge to transfer large amounts of data.
* **C.** Upload the data to the closest S3 bucket. Set up a cross-region replication and copy the objects to the destination bucket.
* **D.** Enable Transfer Acceleration in the destination bucket and upload the collected data using Multipart Upload. (Correct)

Amazon S3 is object storage built to store and retrieve any amount of data from anywhere on the Internet. It’s a simple storage service that offers industry-leading durability, availability, performance, security, and virtually unlimited scalability at very low costs. Amazon S3 is also designed to be highly flexible. Store any type and amount of data that you want; read the same piece of data a million times or only for emergency disaster recovery; build a simple FTP application or a sophisticated web application.



Since the weather forecasting application is located in N.Virginia, you need to transfer all the data in the same AWS Region. With Amazon S3 Transfer Acceleration, you can speed up content transfers to and from Amazon S3 by as much as 50-500% for long-distance transfer of larger objects. Multipart upload allows you to upload a single object as a set of parts. After all the parts of your object are uploaded, Amazon S3 then presents the data as a single object. This approach is the fastest way to aggregate all the data.

Hence, the correct answer is: **Enable Transfer Acceleration in the destination bucket and upload the collected data using Multipart Upload.**

The option that says: **Upload the data to the closest S3 bucket. Set up a cross-region replication and copy the objects to the destination bucket** is incorrect because replicating the objects to the destination bucket takes about 15 minutes. Take note that the requirement in the scenario is to aggregate the data in the fastest way.

The option that says: **Use AWS Snowball Edge to transfer large amounts of data** is incorrect because the end-to-end time to transfer up to 80 TB of data into AWS Snowball Edge is approximately one week.

The option that says: **Set up a Site-to-Site VPN connection** is incorrect because setting up a VPN connection is not needed in this scenario. Site-to-Site VPN is just used for establishing secure connections between an on-premises network and Amazon VPC. Also, this approach is not the fastest way to transfer your data. You must use Amazon S3 Transfer Acceleration.

References:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/replication.html>

<https://docs.aws.amazon.com/AmazonS3/latest/dev/transfer-acceleration.html>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

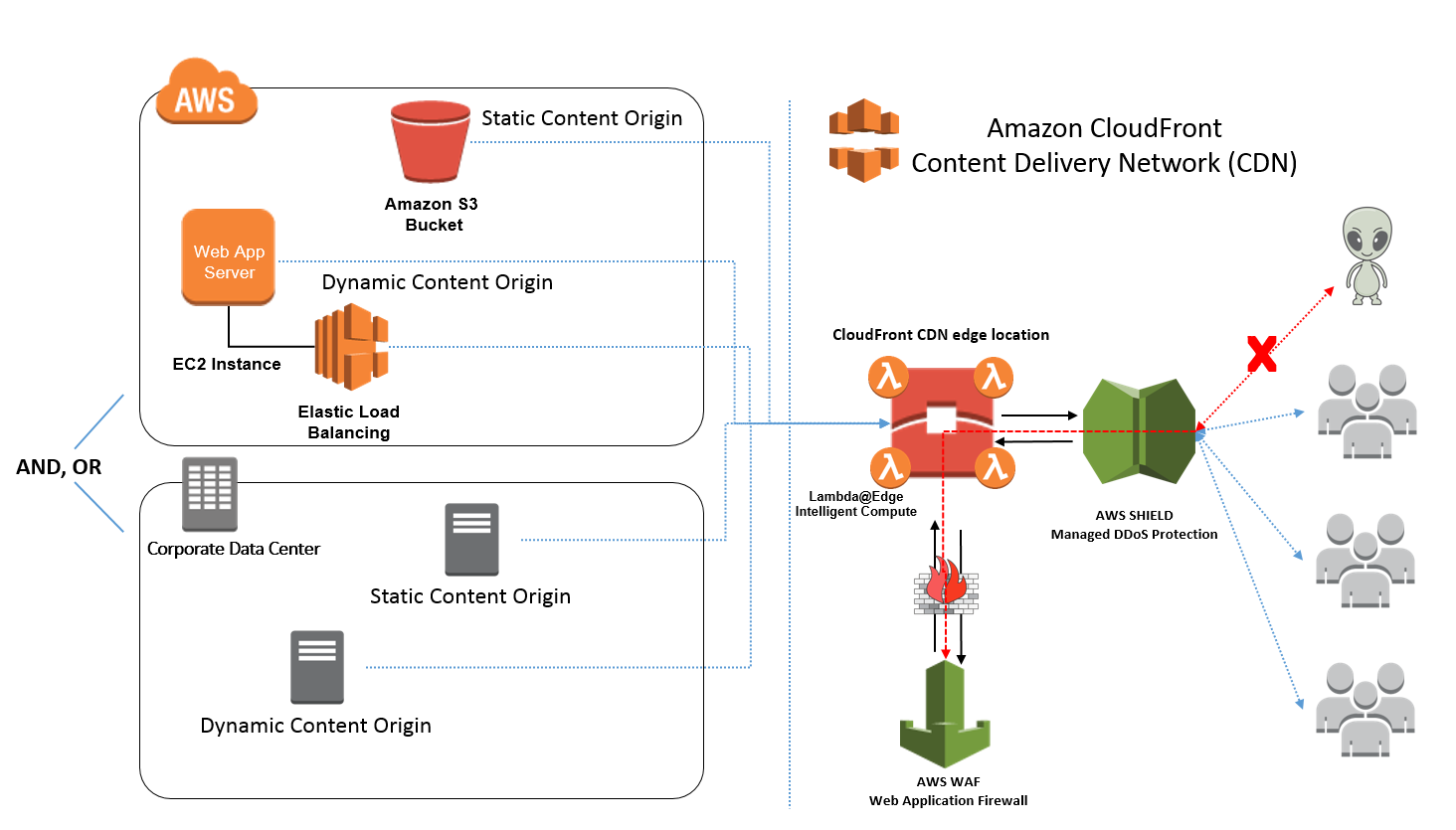
**5. QUESTION**

A start-up company that offers an intuitive financial data analytics service has consulted you about their AWS architecture. They have a fleet of Amazon EC2 worker instances that process financial data and then outputs reports which are used by their clients. You must store the generated report files in a durable storage. The number of files to be stored can grow over time as the start-up company is expanding rapidly overseas and hence, they also need a way to distribute the reports faster to clients located across the globe.

Which of the following is a cost-efficient and scalable storage option that you should use for this scenario?

* **A.** Use Amazon Redshift as the data storage and CloudFront as the CDN.
* **B.** Use multiple EC2 instance stores for data storage and ElastiCache as the CDN.
* **C.** Use Amazon S3 Glacier as the data storage and ElastiCache as the CDN.
* **D.** Use Amazon S3 as the data storage and CloudFront as the CDN. (Correct)

A Content Delivery Network (CDN) is a critical component of nearly any modern web application. It used to be that CDN merely improved the delivery of content by replicating commonly requested files (static content) across a globally distributed set of caching servers. However, CDNs have become much more useful over time.

For caching, a CDN will reduce the load on an application origin and improve the experience of the requestor by delivering a local copy of the content from a nearby cache edge, or Point of Presence (PoP). The application origin is off the hook for opening the connection and delivering the content directly as the CDN takes care of the heavy lifting. The end result is that the application origins don’t need to scale to meet demands for static content.

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment. CloudFront is integrated with AWS – both physical locations that are directly connected to the AWS global infrastructure, as well as other AWS services.

**Amazon S3** offers a highly durable, scalable, and secure destination for backing up and archiving your critical data. This is the correct option as the start-up company is looking for a durable storage to store the audio and text files. In addition, ElastiCache is only used for caching and not specifically as a Global Content Delivery Network (CDN).

**Using Amazon Redshift as the data storage and CloudFront as the CDN** is incorrect as Amazon Redshift is usually used as a Data Warehouse.

**Using Amazon S3 Glacier as the data storage and ElastiCache as the CDN** is incorrect as Amazon S3 Glacier is usually used for data archives.

**Using multiple EC2 instance stores for data storage and ElastiCache as the CDN** is incorrect as data stored in an instance store is not durable.

References:

<https://aws.amazon.com/s3/>

<https://aws.amazon.com/caching/cdn/>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

**6. QUESTION**

A Solutions Architect created a new Standard-class S3 bucket to store financial reports that are not frequently accessed but should immediately be available when an auditor requests them. To save costs, the Architect changed the storage class of the S3 bucket from Standard to Infrequent Access storage class.

In Amazon S3 Standard – Infrequent Access storage class, which of the following statements are true? (Select TWO.)

* **A.** It is designed for data that is accessed less frequently. (Correct)
* **B.** It provides high latency and low throughput performance.
* **C.** Ideal to use for data archiving.
* **D.** It automatically moves data to the most cost-effective access tier without any operational overhead.
* **E.** It is designed for data that requires rapid access when needed. (Correct)

Amazon S3 Standard – Infrequent Access (Standard – IA) is an Amazon S3 storage class for data that is accessed less frequently, but requires rapid access when needed. Standard – IA offers the high durability, throughput, and low latency of Amazon S3 Standard, with a low per GB storage price and per GB retrieval fee.



This combination of low cost and high performance make Standard – IA ideal for long-term storage, backups, and as a data store for disaster recovery. The Standard – IA storage class is set at the object level and can exist in the same bucket as Standard, allowing you to use lifecycle policies to automatically transition objects between storage classes without any application changes.

Key Features:

– Same low latency and high throughput performance of Standard

– Designed for durability of 99.999999999% of objects

– Designed for 99.9% availability over a given year

– Backed with the Amazon S3 Service Level Agreement for availability

– Supports SSL encryption of data in transit and at rest

– Lifecycle management for automatic migration of objects

Hence, the correct answers are:

– **It is designed for data that is accessed less frequently.**

– **It is designed for data that requires rapid access when needed.**

The option that says: **It automatically moves data to the most cost-effective access tier without any operational overhead** is incorrect as it actually refers to Amazon S3 – Intelligent Tiering, which is the only cloud storage class that delivers automatic cost savings by moving objects between different access tiers when access patterns change.

The option that says: **It provides high latency and low throughput performance** is incorrect as it should be “low latency” and “high throughput” instead. S3 automatically scales performance to meet user demands.

The option that says: **Ideal to use for data archiving** is incorrect because this statement refers to Amazon S3 Glacier. Glacier is a secure, durable, and extremely low-cost cloud storage service for data archiving and long-term backup.

References:

<https://aws.amazon.com/s3/storage-classes/>

<https://aws.amazon.com/s3/faqs>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

**7. QUESTION**

For data privacy, a healthcare company has been asked to comply with the Health Insurance Portability and Accountability Act (HIPAA). The company stores all its backups on an Amazon S3 bucket. It is required that data stored on the S3 bucket must be encrypted.

What is the best option to do this? (Select TWO.)

* **A.** Enable Server-Side Encryption on an S3 bucket to make use of AES-128 encryption.
* **B.** Enable Server-Side Encryption on an S3 bucket to make use of AES-256 encryption.
* **C.** Store the data in encrypted EBS snapshots.
* **D.** Store the data on EBS volumes with encryption enabled instead of using Amazon S3.
* **E.** Before sending the data to Amazon S3 over HTTPS, encrypt the data locally first using your own encryption keys. (Correct)

Server-side encryption is about data encryption at rest—that is, Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects. For example, if you share your objects using a pre-signed URL, that URL works the same way for both encrypted and unencrypted objects.



You have three mutually exclusive options depending on how you choose to manage the encryption keys:

1. Use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)
2. Use Server-Side Encryption with AWS KMS-Managed Keys (SSE-KMS)
3. Use Server-Side Encryption with Customer-Provided Keys (SSE-C)

The options that say: **Before sending the data to Amazon S3 over HTTPS, encrypt the data locally first using your own encryption keys** and **Enable Server-Side Encryption on an S3 bucket to make use of AES-256 encryption** are correct because these options are using client-side encryption and Amazon S3-Managed Keys (SSE-S3) respectively. *Client-side encryption* is the act of encrypting data before sending it to Amazon S3 while SSE-S3 uses AES-256 encryption.

**Storing the data on EBS volumes with encryption enabled instead of using Amazon S3** and **storing the data in encrypted EBS snapshots** are incorrect because both options use EBS encryption and not S3.

**Enabling Server-Side Encryption on an S3 bucket to make use of AES-128 encryption** is incorrect as S3 doesn’t provide AES-128 encryption, only AES-256.

References:

<http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html>

<https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>

**8. QUESTION**

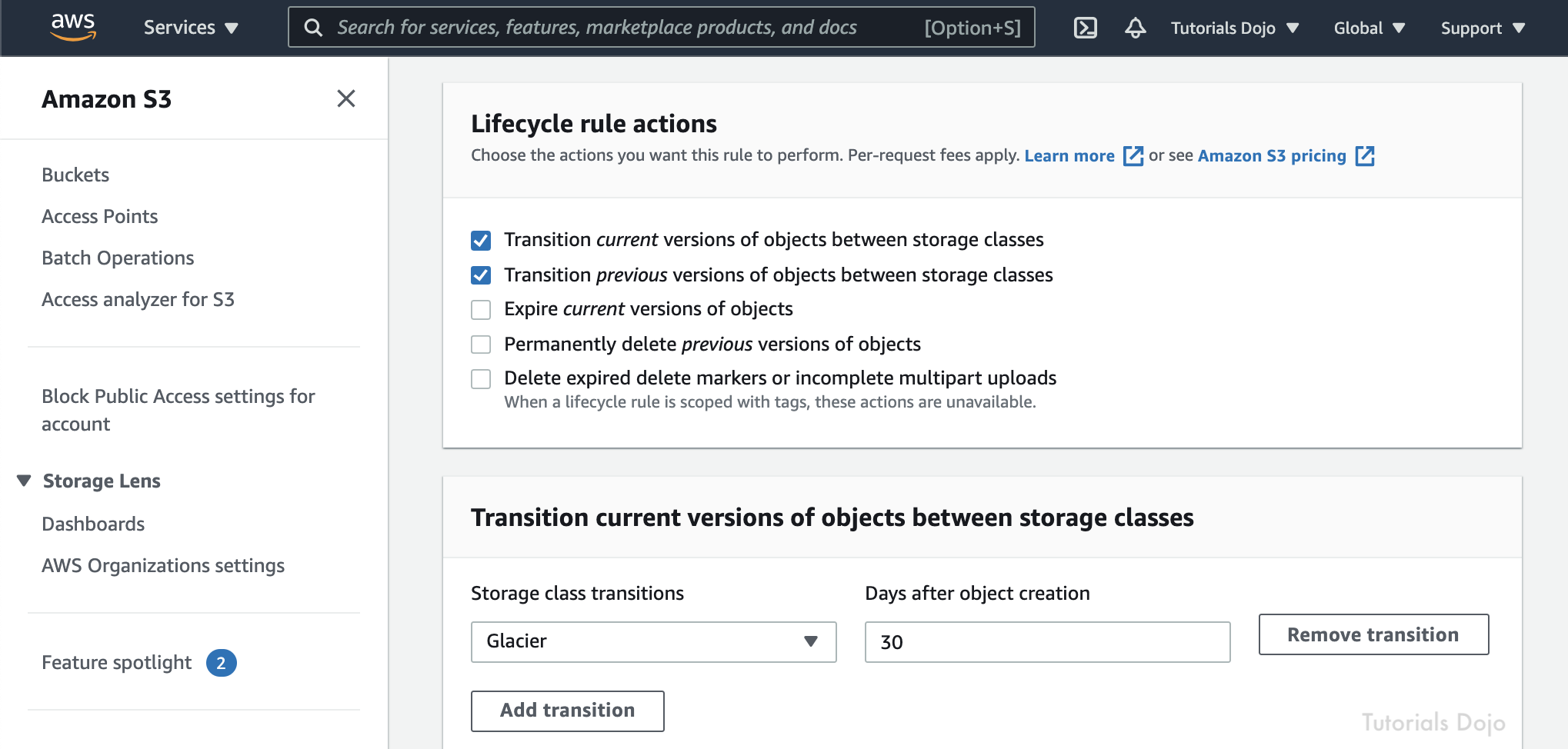
A Solutions Architect is working for a financial company. The manager wants to have the ability to automatically transfer obsolete data from their S3 bucket to a low-cost storage system in AWS after a certain period of time.

What is the best solution that the Architect can provide to them?

* **A.** Use Lifecycle Policies in S3 to move obsolete data to Glacier. (Correct)
* **B.** Use an EC2 instance and a scheduled job to transfer the obsolete data from their S3 location to Amazon S3 Glacier.
* **C.** Use Amazon SQS.
* **D.** Use Amazon Timestream.

In this scenario, you can use lifecycle policies in S3 to automatically move obsolete data to Glacier.

Lifecycle configuration in Amazon S3 enables you to specify the lifecycle management of objects in a bucket. The configuration is a set of one or more rules, where each rule defines an action for Amazon S3 to apply to a group of objects.



These actions can be classified as follows:

Transition actions – In which you define when objects transition to another storage class. For example, you may choose to transition objects to the STANDARD\_IA (IA, for infrequent access) storage class 30 days after creation, or archive objects to the GLACIER storage class one year after creation.

Expiration actions – In which you specify when the objects expire. Then Amazon S3 deletes the expired objects on your behalf.

The option that says: **Use an EC2 instance and a scheduled job to transfer the obsolete data from their S3 location to Amazon S3 Glacier** is incorrect because you don’t need to create a scheduled job in EC2 as you can simply use the lifecycle policy in S3.

The option that says: **Use Amazon SQS** is incorrect as SQS is not a storage service. Amazon SQS is primarily used to decouple your applications by queueing the incoming requests of your application.

The option that says: **Use Amazon Timestream** is incorrect. While Amazon Timestream is great for storing and analyzing time-series data, it doesn’t directly address the requirement of moving data from S3 to a lower-cost storage option based on the age of the data. The best solution for this specific use case would be to use Lifecycle Policies in S3 to move obsolete data to Glacier, which is a low-cost storage service in AWS. This can be done by setting up some rules (e.g., which folder) and it will transition the data.

References:

<http://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html>

<https://aws.amazon.com/blogs/aws/archive-s3-to-glacier/>

Check out this Amazon S3 Cheat Sheet:

<https://tutorialsdojo.com/amazon-s3/>