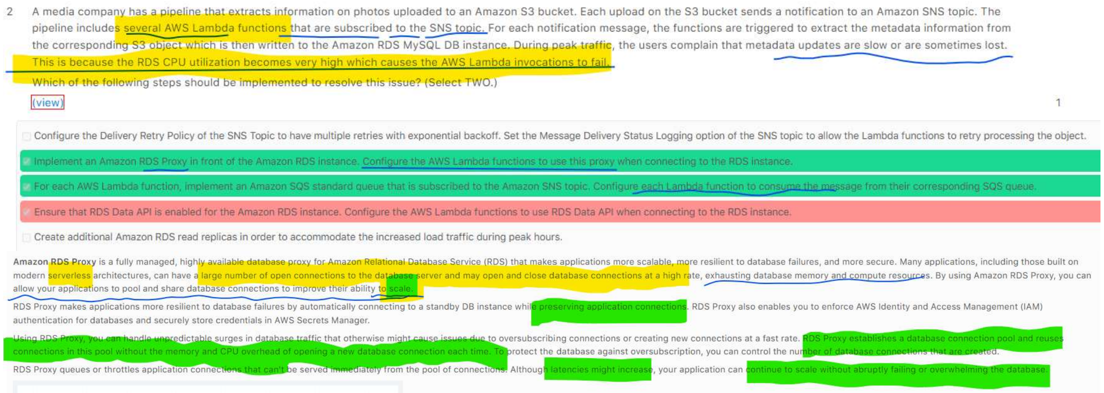
5-simulacoes profissionais

Notebook:	solutions profissional_ef8e1c0f-5d3c-42ab-ab04-9217e53e12cd		
Created:	18/10/2025 20:11	**Updated:**	19/10/2025 18:35
Author:	erikachen19@gmail.com		
URL:	https://portal.tutorialsdojo.com/courses/aws-certified-solutions-architect-professional...		



2 A media company has a pipeline that extracts information on photos uploaded to an Amazon S3 bucket. Each upload on the S3 bucket sends a notification to an Amazon SNS topic. The pipeline includes several AWS Lambda functions that are subscribed to the SNS topic. For each notification message, the functions are triggered to extract the metadata information from the corresponding S3 object which is then written to the Amazon RDS MySQL DB instance. During peak traffic, the users complain that metadata updates are slow or are sometimes lost. This is because the RDS CPU utilization becomes very high which causes the AWS Lambda invocations to fail.

Which of the following steps should be implemented to resolve this issue? (Select TWO.)

(view) 1

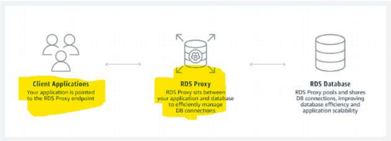
- ☐ Configure the Delivery Retry Policy of the SNS Topic to have multiple retries with exponential backoff. Set the Message Delivery Status Logging option of the SNS topic to allow the Lambda functions to retry processing the object.
- ☑ Implement an Amazon RDS Proxy in front of the Amazon RDS instance. Configure the AWS Lambda functions to use this proxy when connecting to the RDS instance.
- ☑ For each AWS Lambda function, implement an Amazon SQS standard queue that is subscribed to the Amazon SNS topic. Configure each Lambda function to consume the message from their corresponding SQS queue.
- ☑ Ensure that RDS Data API is enabled for the Amazon RDS instance. Configure the AWS Lambda functions to use RDS Data API when connecting to the RDS instance.
- ☐ Create additional Amazon RDS read replicas in order to accommodate the increased load traffic during peak hours.

Amazon RDS Proxy is a fully managed, highly available database proxy for Amazon Relational Database Service (RDS) that makes applications more scalable, more resilient to database failures, and more secure. Many applications, including those built on modern serverless architectures, can have a large number of open connections to the database server and may open and close database connections at a high rate, exhausting database memory and compute resources. By using Amazon RDS Proxy, you can allow your applications to pool and share database connections to improve their ability to scale.

RDS Proxy makes applications more resilient to database failures by automatically connecting to a standby DB instance while preserving application connections. RDS Proxy also enables you to enforce AWS Identity and Access Management (IAM) authentication for databases and securely store credentials in AWS Secrets Manager.

Using RDS Proxy, you can handle unpredictable surges in database traffic that otherwise might cause issues due to oversubscribing connections or creating new connections at a fast rate. RDS Proxy establishes a database connection pool and reuses connections in this pool without the memory and CPU overhead of opening a new database connection each time. To protect the database against oversubscription, you can control the number of database connections that are created.

RDS Proxy queues or throttles application connections that can't be served immediately from the pool of connections. Although latencies might increase, your application can continue to scale without abruptly failing or overwhelming the database.

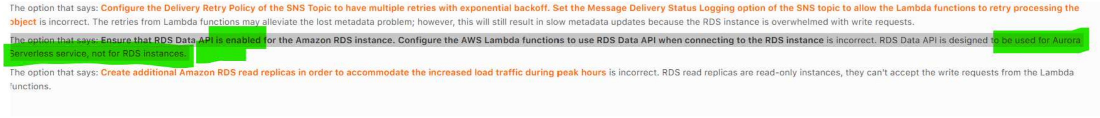

Client Applications
Your application is pointed to the RDS Proxy endpoint
RDS Proxy
RDS Proxy sits between your application and database to efficiently manage DB connections
RDS Database
RDS Proxy pools and shares DB connections, improving database efficiency and application scalability

Amazon RDS 代理是 Amazon Relational Database Service (RDS) 的完全托管、高可用性数据库代理，可提高应用程序的可扩展性、应对数据库故障的弹性以及安全性。许多应用程序（包括基于现代无服务器架构构建的应用程序）都可能与数据库服务器建立大量开放连接，并且可能以极高的频率打开和关闭数据库连接，从而耗尽数据库内存和计算资源。通过使用 Amazon RDS 代理，您可以允许应用程序池化和共享数据库连接，以提高其扩展能力。

RDS Proxy 可自动连接到备用数据库实例，同时保留应用程序连接，从而增强应用程序对数据库故障的恢复能力。RDS Proxy 还允许您对数据库强制执行 AWS Identity and Access Management (IAM) 身份验证，并将凭证安全地存储在 AWS Secrets Manager 中。

使用 RDS Proxy，您可以应对数据库流量不可预测的激增，否则这些流量可能会因连接超额订阅或快速创建新连接而导致问题。RDS Proxy 会建立数据库连接池并重复使用池中的连接，而无需每次打开新的数据库连接而产生内存和 CPU 开销。为了防止数据库超额订阅，您可以控制创建的数据库连接数量。

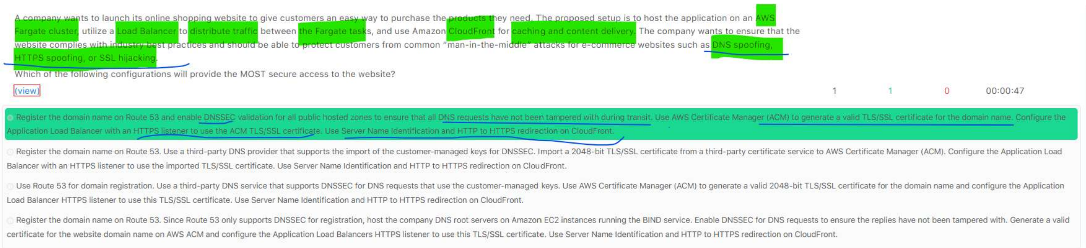
RDS Proxy 会对无法立即从连接池中获取服务的应用程序连接进行排队或限制。尽管延迟可能会增加，但您的应用程序可以继续扩展，而不会突然出现故障或数据库不堪重负。



The option that says: **Configure the Delivery Retry Policy of the SNS Topic to have multiple retries with exponential backoff. Set the Message Delivery Status Logging option of the SNS topic to allow the Lambda functions to retry processing the object** is incorrect. The retries from Lambda functions may alleviate the lost metadata problem; however, this will still result in slow metadata updates because the RDS instance is overwhelmed with write requests.

The option that says: **Ensure that RDS Data API is enabled for the Amazon RDS instance. Configure the AWS Lambda functions to use RDS Data API when connecting to the RDS instance** is incorrect. RDS Data API is designed to be used for Aurora Serverless service, not for RDS instances.

The option that says: **Create additional Amazon RDS read replicas in order to accommodate the increased load traffic during peak hours** is incorrect. RDS read replicas are read-only instances, they can't accept the write requests from the Lambda functions.



7 A company wants to launch its online shopping website to give customers an easy way to purchase the products they need. The proposed setup is to host the application on an AWS Fargate cluster, utilize a Load Balancer to distribute traffic between the Fargate tasks, and use Amazon CloudFront for caching and content delivery. The company wants to ensure that the website complies with industry best practices and should be able to protect customers from common "man-in-the-middle" attacks for e-commerce websites such as DNS spoofing, HTTPS spoofing, or SSL hijacking.

Which of the following configurations will provide the MOST secure access to the website?

(view)

1 1 0 00:00:47

● Register the domain name on Route 53 and enable DNSSEC validation for all public hosted zones to ensure that all DNS requests have not been tampered with during transit. Use AWS Certificate Manager (ACM) to generate a valid TLS/SSL certificate for the domain name. Configure the Application Load Balancer with an HTTPS listener to use the ACM TLS/SSL certificate. Use Server Name Identification and HTTP to HTTPS redirection on CloudFront.

○ Register the domain name on Route 53. Use a third-party DNS provider that supports the import of the customer-managed keys for DNSSEC. Import a 2048-bit TLS/SSL certificate from a third-party certificate service to AWS Certificate Manager (ACM). Configure the Application Load Balancer with an HTTPS listener to use the imported TLS/SSL certificate. Use Server Name Identification and HTTP to HTTPS redirection on CloudFront.

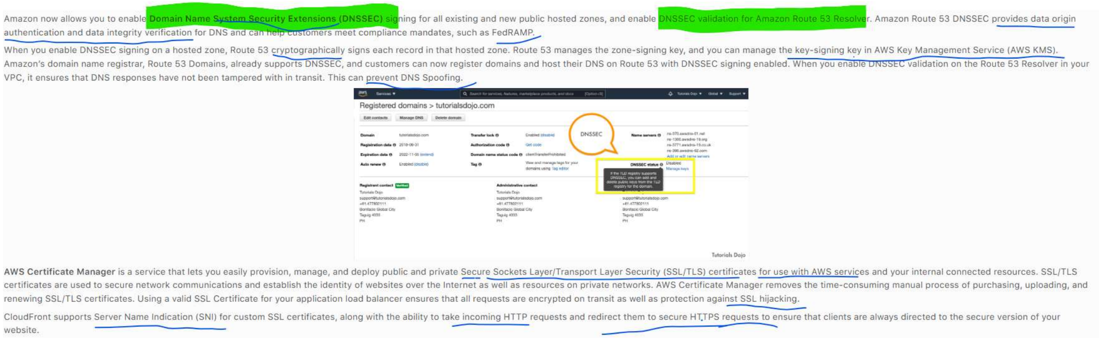
○ Use Route 53 for domain registration. Use a third-party DNS service that supports DNSSEC for DNS requests that use the customer-managed keys. Use AWS Certificate Manager (ACM) to generate a valid 2048-bit TLS/SSL certificate for the domain name and configure the Application Load Balancer HTTPS listener to use this TLS/SSL certificate. Use Server Name Identification and HTTP to HTTPS redirection on CloudFront.

○ Register the domain name on Route 53. Since Route 53 only supports DNSSEC for registration, host the company DNS root servers on Amazon EC2 instances running the BIND service. Enable DNSSEC for DNS requests to ensure the replies have not been tampered with. Generate a valid certificate for the website domain name on AWS ACM and configure the Application Load Balancers HTTPS listener to use this TLS/SSL certificate. Use Server Name Identification and HTTP to HTTPS redirection on CloudFront.

该公司希望推出其在线购物网站，让客户能够轻松购买所需产品。建议的设置是将应用程序托管在 AWS Fargate 集群上，使用负载均衡器在 Fargate 任务之间分配流量，并使用 Amazon CloudFront 进行缓存和内容交付。该公司希望确保网站符合行业最佳实践，并能够保护客户免受电子商务网站常见的“中间人”攻击，例如 DNS 欺骗、HTTPS 欺骗或 SSL 劫持。

Register the domain name on Route 53 and enable **DNSSEC** validation for all public hosted zones to ensure that all DNS requests have not been tampered 篡改with during transit. Use AWS Certificate Manager (ACM) to generate a valid TLS/SSL certificate for the domain name. Configure the Application Load Balancer with an HTTPS listener to use the ACM TLS/SSL certificate. Use **Server Name Identification** and HTTP to HTTPS redirection on CloudFront.在 Route 53 上注册域名，并为所有公共托管区域启用 DNSSEC 验证，以确保所有 DNS 请求在传输过程中均未被篡改。使用 AWS 证书管理器 (ACM) 为域名生成有效的 TLS/SSL 证书。配置带有 HTTPS 侦听器的应用程序负载均衡器以使用 ACM TLS/SSL 证书。在 CloudFront 上使用服务器名称标识和 HTTP 到 HTTPS 重定向。

DNS Spoofing.DNS 欺骗。



Amazon now allows you to enable **Domain Name System Security Extensions (DNSSEC)** signing for all existing and new public hosted zones, and enable DNSSEC validation for Amazon Route 53 Resolver. Amazon Route 53 DNSSEC provides data origin authentication and data integrity verification for DNS and can help customers meet compliance mandates, such as FedRAMP.

When you enable DNSSEC signing on a hosted zone, Route 53 cryptographically signs each record in that hosted zone. Route 53 manages the zone-signing key, and you can manage the key-signing key in AWS Key Management Service (AWS KMS). Amazon's domain name registrar, Route 53 Domains, already supports DNSSEC, and customers can now register domains and host their DNS on Route 53 with DNSSEC signing enabled. When you enable DNSSEC validation on the Route 53 Resolver in your VPC, it ensures that DNS responses have not been tampered with in transit. This can prevent DNS Spoofing.

AWS Certificate Manager is a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources. SSL/TLS certificates are used to secure network communications and establish the identity of websites over the Internet as well as resources on private networks. AWS Certificate Manager removes the time-consuming manual process of purchasing, uploading, and renewing SSL/TLS certificates. Using a valid SSL Certificate for your application load balancer ensures that all requests are encrypted on transit as well as protection against SSL hijacking.

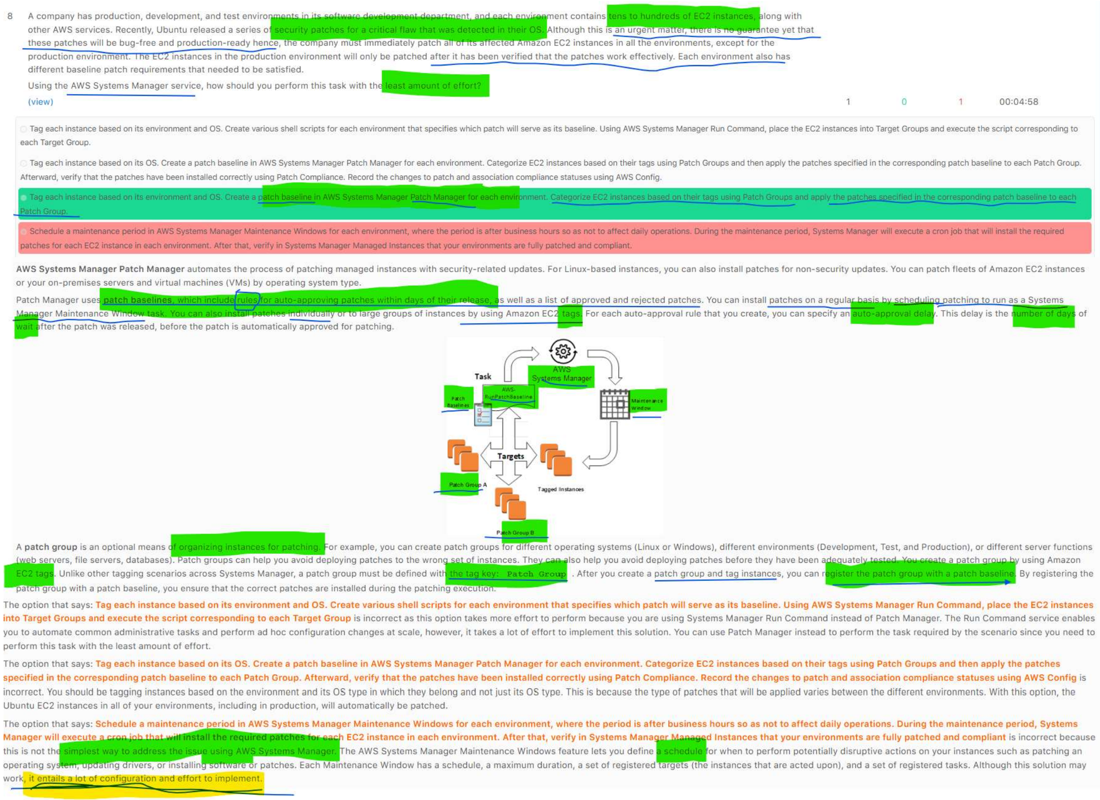
CloudFront supports Server Name Indication (SNI) for custom SSL certificates, along with the ability to take incoming HTTP requests and redirect them to secure HTTPS requests to ensure that clients are always directed to the secure version of your website.

Amazon 现在允许您为所有现有和新的公共托管区域启用域名系统安全扩展 (DNSSEC) 签名，并为 Amazon Route 53 解析器启用 DNSSEC 验证。Amazon Route 53 DNSSEC 为 DNS 提供数据源身份验证和数据完整性验证，并可帮助客户满足 FedRAMP 等合规性要求。

在托管区域启用 DNSSEC 签名后，Route 53 会以加密方式对该托管区域中的每个记录进行签名。Route 53 管理区域签名密钥，您可以在 AWS Key Management Service (AWS KMS) 中管理密钥签名密钥。Amazon 的域名注册商 Route 53 Domains 已支持 DNSSEC，客户现在可以在启用 DNSSEC 签名的情况下在 Route 53 上注册域名并托管其 DNS。在 VPC 中的 Route 53 解析器上启用 DNSSEC 验证后，可确保 DNS 响应在传输过程中未被篡改。这可以防止 DNS 欺骗。

AWS Certificate Manager 是一项服务，可让您轻松预置、管理和部署公有和私有安全套接字层/传输层安全性 (SSL/TLS) 证书，以便用于 AWS 服务和您的内部连接资源。SSL/TLS 证书用于保护网络通信，并确立互联网上网站以及私有网络上资源的身份。AWS Certificate Manager 消除了购买、上传和续订 SSL/TLS 证书的耗时手动流程。为您的应用程序负载均衡器使用有效的 SSL 证书可确保所有请求在传输过程中都经过加密，并防止 SSL 劫持。

CloudFront 支持自定义 SSL 证书的服务器名称指示 (SNI)，以及接收传入 HTTP 请求并将其重定向到安全的 HTTPS 请求的功能，以确保客户端始终定向到您网站的安全版本。



8 A company has production, development, and test environments in its software development department, and each environment contains tens to hundreds of EC2 instances, along with other AWS services. Recently, Ubuntu released a series of security patches for a critical flaw that was detected in their OS. Although this is an urgent matter, there is no guarantee yet that these patches will be bug-free and production-ready hence, the company must immediately patch all of its affected Amazon EC2 instances in all the environments, except for the production environment. The EC2 instances in the production environment will only be patched after it has been verified that the patches work effectively. Each environment also has different baseline patch requirements that needed to be satisfied.

Using the AWS Systems Manager service, how should you perform this task with the least amount of effort?

(view) 1 0 1 00:04:58

○ Tag each instance based on its environment and OS. Create various shell scripts for each environment that specifies which patch will serve as its baseline. Using AWS Systems Manager Run Command, place the EC2 instances into Target Groups and execute the script corresponding to each Target Group.

○ Tag each instance based on its OS. Create a patch baseline in AWS Systems Manager Patch Manager for each environment. Categorize EC2 instances based on their tags using Patch Groups and then apply the patches specified in the corresponding patch baseline to each Patch Group. Afterward, verify that the patches have been installed correctly using Patch Compliance. Record the changes to patch and association compliance statuses using AWS Config.

● Tag each instance based on its environment and OS. Create a patch baseline in AWS Systems Manager Patch Manager for each environment. Categorize EC2 instances based on their tags using Patch Groups and apply the patches specified in the corresponding patch baseline to each Patch Group.

○ Schedule a maintenance period in AWS Systems Manager Maintenance Windows for each environment, where the period is after business hours so as not to affect daily operations. During the maintenance period, Systems Manager will execute a cron job that will install the required patches for each EC2 instance in each environment. After that, verify in Systems Manager Managed Instances that your environments are fully patched and compliant.

AWS Systems Manager Patch Manager automates the process of patching managed instances with security-related updates. For Linux-based instances, you can also install patches for non-security updates. You can patch fleets of Amazon EC2 instances or your on-premises servers and virtual machines (VMs) by operating system type.

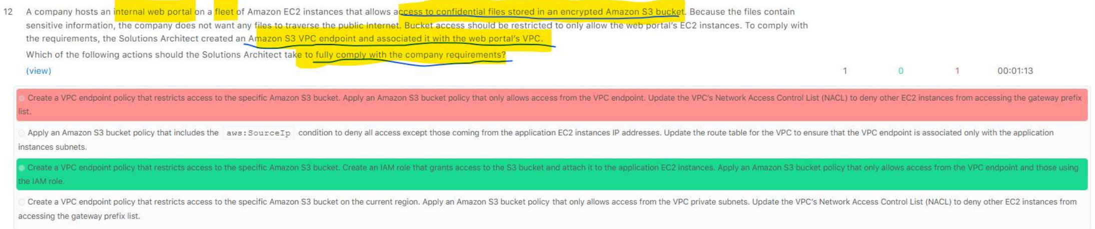
Patch Manager uses **patch baselines**, which include rules for auto-approving patches within days of their release, as well as a list of approved and rejected patches. You can install patches on a regular basis by scheduling patching to run as a Systems Manager Maintenance Window task. You can also install patches individually or to large groups of instances by using Amazon EC2 tags. For each auto-approval rule that you create, you can specify an auto-approval delay. This delay is the number of days of wait after the patch was released, before the patch is automatically approved for patching.

A **patch group** is an optional means of organizing instances for patching. For example, you can create patch groups for different operating systems (Linux or Windows), different environments (Development, Test, and Production), or different server functions (web servers, file servers, databases). Patch groups can help you avoid deploying patches to the wrong set of instances. They can also help you avoid deploying patches before they have been adequately tested. You create a patch group by using Amazon EC2 tags. Unlike other tagging scenarios across Systems Manager, a patch group must be defined with the tag key: `Patch Group` . After you create a patch group and tag instances, you can register the patch group with a patch baseline. By registering the patch group with a patch baseline, you ensure that the correct patches are installed during the patching execution.

The option that says: **Tag each instance based on its environment and OS. Create various shell scripts for each environment that specifies which patch will serve as its baseline. Using AWS Systems Manager Run Command, place the EC2 instances into Target Groups and execute the script corresponding to each Target Group** is incorrect as this option takes more effort to perform because you are using Systems Manager Run Command instead of Patch Manager. The Run Command service enables you to automate common administrative tasks and perform ad hoc configuration changes at scale, however, it takes a lot of effort to implement this solution. You can use Patch Manager instead to perform the task required by the scenario since you need to perform this task with the least amount of effort.

The option that says: **Tag each instance based on its OS. Create a patch baseline in AWS Systems Manager Patch Manager for each environment. Categorize EC2 instances based on their tags using Patch Groups and then apply the patches specified in the corresponding patch baseline to each Patch Group. Afterward, verify that the patches have been installed correctly using Patch Compliance. Record the changes to patch and association compliance statuses using AWS Config** is incorrect. You should be tagging instances based on the environment and its OS type in which they belong and not just its OS type. This is because the type of patches that will be applied varies between the different environments. With this option, the Ubuntu EC2 instances in all of your environments, including in production, will automatically be patched.

The option that says: **Schedule a maintenance period in AWS Systems Manager Maintenance Windows for each environment, where the period is after business hours so as not to affect daily operations. During the maintenance period, Systems Manager will execute a cron job that will install the required patches for each EC2 instance in each environment. After that, verify in Systems Manager Managed Instances that your environments are fully patched and compliant** is incorrect because this is not the simplest way to address the issue using AWS Systems Manager. The AWS Systems Manager Maintenance Windows feature lets you define a schedule for when to perform potentially disruptive actions on your instances such as patching an operating system, updating drivers, or installing software or patches. Each Maintenance Window has a schedule, a maximum duration, a set of registered targets (the instances that are acted upon), and a set of registered tasks. Although this solution may work, it entails a lot of configuration and effort to implement.



12 A company hosts an internal web portal on a fleet of Amazon EC2 instances that allows access to confidential files stored in an encrypted Amazon S3 bucket. Because the files contain sensitive information, the company does not want any files to traverse the public Internet. Bucket access should be restricted to only allow the web portal's EC2 instances. To comply with the requirements, the Solutions Architect created an Amazon S3 VPC endpoint and associated it with the web portal's VPC.

Which of the following actions should the Solutions Architect take to fully comply with the company requirements?

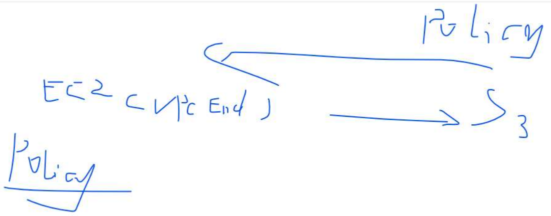
(view) 1 0 1 00:01:13

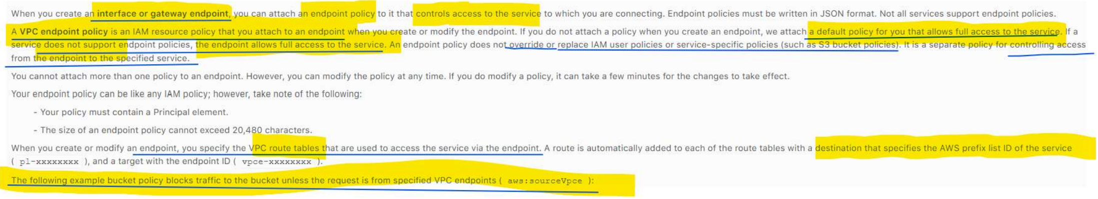
○ Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket. Apply an Amazon S3 bucket policy that only allows access from the VPC endpoint. Update the VPC's Network Access Control List (NACL) to deny other EC2 instances from accessing the gateway prefix list.

○ Apply an Amazon S3 bucket policy that includes the `aws:SourceIp` condition to deny all access except those coming from the application EC2 instances IP addresses. Update the route table for the VPC to ensure that the VPC endpoint is associated only with the application instances subnets.

● Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket. Create an IAM role that grants access to the S3 bucket and attach it to the application EC2 instances. Apply an Amazon S3 bucket policy that only allows access from the VPC endpoint and those using the IAM role.

○ Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket on the current region. Apply an Amazon S3 bucket policy that only allows access from the VPC private subnets. Update the VPC's Network Access Control List (NACL) to deny other EC2 instances from accessing the gateway prefix list.





When you create an **interface or gateway endpoint**, you can attach an endpoint policy to it that controls access to the service to which you are connecting. Endpoint policies must be written in JSON format. Not all services support endpoint policies.

A **VPC endpoint policy** is an IAM resource policy that you attach to an endpoint when you create or modify the endpoint. If you do not attach a policy when you create an endpoint, we attach a default policy for you that allows full access to the service. If a service does not support endpoint policies, the endpoint allows full access to the service. An endpoint policy does not override or replace IAM user policies or service-specific policies (such as S3 bucket policies). It is a separate policy for controlling access from the endpoint to the specified service.

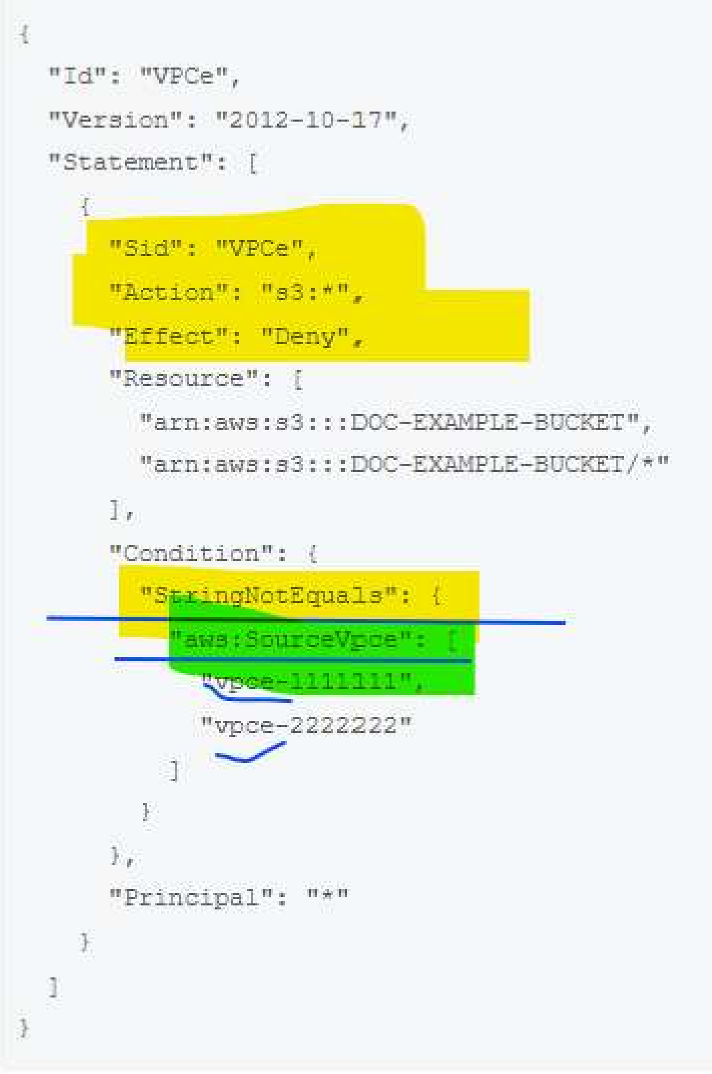
You cannot attach more than one policy to an endpoint. However, you can modify the policy at any time. If you do modify a policy, it can take a few minutes for the changes to take effect.

Your endpoint policy can be like any IAM policy; however, take note of the following:

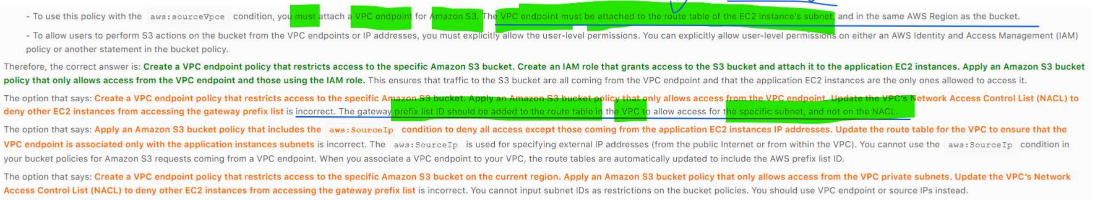
- Your policy must contain a Principal element.
- The size of an endpoint policy cannot exceed 20,480 characters.

When you create or modify an endpoint, you specify the VPC route tables that are used to access the service via the endpoint. A route is automatically added to each of the route tables with a destination that specifies the AWS prefix list ID of the service (`pl-xxxxxxxx`), and a target with the endpoint ID (`vpce-xxxxxxxx`).

The following example bucket policy blocks traffic to the bucket unless the request is from specified VPC endpoints (`aws:sourceVpce`):



```
{
  "Id": "VPCe",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VPCe",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:SourceVpce": [
            "vpce-1111111",
            "vpce-2222222"
          ]
        }
      },
      "Principal": "*"
    }
  ]
}
```



- To use this policy with the `aws:sourceVpce` condition, you must attach a VPC endpoint for Amazon S3. The VPC endpoint must be attached to the route table of the EC2 instance's subnet, and in the same AWS Region as the bucket.

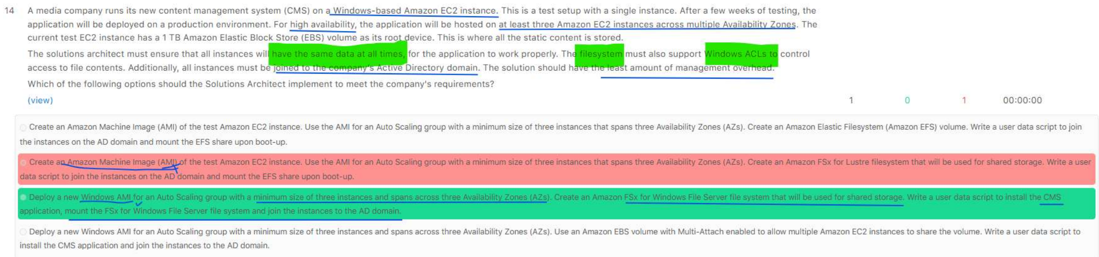
- To allow users to perform S3 actions on the bucket from the VPC endpoints or IP addresses, you must explicitly allow the user-level permissions. You can explicitly allow user-level permissions on either an AWS Identity and Access Management (IAM) policy or another statement in the bucket policy.

Therefore, the correct answer is: **Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket. Create an IAM role that grants access to the S3 bucket and attach it to the application EC2 instances. Apply an Amazon S3 bucket policy that only allows access from the VPC endpoint and those using the IAM role.** This ensures that traffic to the S3 bucket are all coming from the VPC endpoint and that the application EC2 instances are the only ones allowed to access it.

The option that says: **Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket. Apply an Amazon S3 bucket policy that only allows access from the VPC endpoint. Update the VPC's Network Access Control List (NACL) to deny other EC2 instances from accessing the gateway prefix list** is incorrect. The gateway prefix list ID should be added to the route table in the VPC to allow access for the specific subnet, and not on the NACL.

The option that says: **Apply an Amazon S3 bucket policy that includes the `aws:SourceIp` condition to deny all access except those coming from the application EC2 instances IP addresses. Update the route table for the VPC to ensure that the VPC endpoint is associated only with the application instances subnets** is incorrect. The `aws:SourceIp` is used for specifying external IP addresses (from the public Internet or from within the VPC). You cannot use the `aws:SourceIp` condition in your bucket policies for Amazon S3 requests coming from a VPC endpoint. When you associate a VPC endpoint to your VPC, the route tables are automatically updated to include the AWS prefix list ID.

The option that says: **Create a VPC endpoint policy that restricts access to the specific Amazon S3 bucket on the current region. Apply an Amazon S3 bucket policy that only allows access from the VPC private subnets. Update the VPC's Network Access Control List (NACL) to deny other EC2 instances from accessing the gateway prefix list** is incorrect. You cannot input subnet IDs as restrictions on the bucket policies. You should use VPC endpoint or source IPs instead.



14 A media company runs its new content management system (CMS) on a Windows-based Amazon EC2 instance. This is a test setup with a single instance. After a few weeks of testing, the application will be deployed on a production environment. For high availability, the application will be hosted on at least three Amazon EC2 instances across multiple Availability Zones. The current test EC2 instance has a 1 TB Amazon Elastic Block Store (EBS) volume as its root device. This is where all the static content is stored.

The solutions architect must ensure that all instances will have the same data at all times, for the application to work properly. The filesystem must also support Windows ACLs to control access to file contents. Additionally, all instances must be joined to the company's Active Directory domain. The solution should have the least amount of management overhead.

Which of the following options should the Solutions Architect implement to meet the company's requirements?

(view) 1 0 1 00:00:00

○ Create an Amazon Machine Image (AMI) of the test Amazon EC2 instance. Use the AMI for an Auto Scaling group with a minimum size of three instances that spans three Availability Zones (AZs). Create an Amazon Elastic Filesystem (Amazon EFS) volume. Write a user data script to join the instances on the AD domain and mount the EFS share upon boot-up.

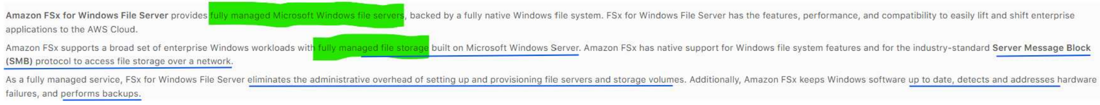
○ Create an Amazon Machine Image (AMI) of the test Amazon EC2 instance. Use the AMI for an Auto Scaling group with a minimum size of three instances that spans three Availability Zones (AZs). Create an Amazon FSx for Lustre filesystem that will be used for shared storage. Write a user data script to join the instances on the AD domain and mount the EFS share upon boot-up.

○ Deploy a new Windows AMI for an Auto Scaling group with a minimum size of three instances and spans across three Availability Zones (AZs). Create an Amazon FSx for Windows File Server file system that will be used for shared storage. Write a user data script to install the CMS application, mount the FSx for Windows File Server file system and join the instances to the AD domain.

○ Deploy a new Windows AMI for an Auto Scaling group with a minimum size of three instances and spans across three Availability Zones (AZs). Use an Amazon EBS volume with Multi-Attach enabled to allow multiple Amazon EC2 instances to share the volume. Write a user data script to install the CMS application and join the instances to the AD domain.

The solutions architect must ensure that all instances will have the same data at all times, for the application to work properly. The filesystem must also support Windows ACLs to control access to file contents. Additionally, all instances must be joined to the company's Active Directory domain. The solution should have the least amount费用 of management overhead开销.

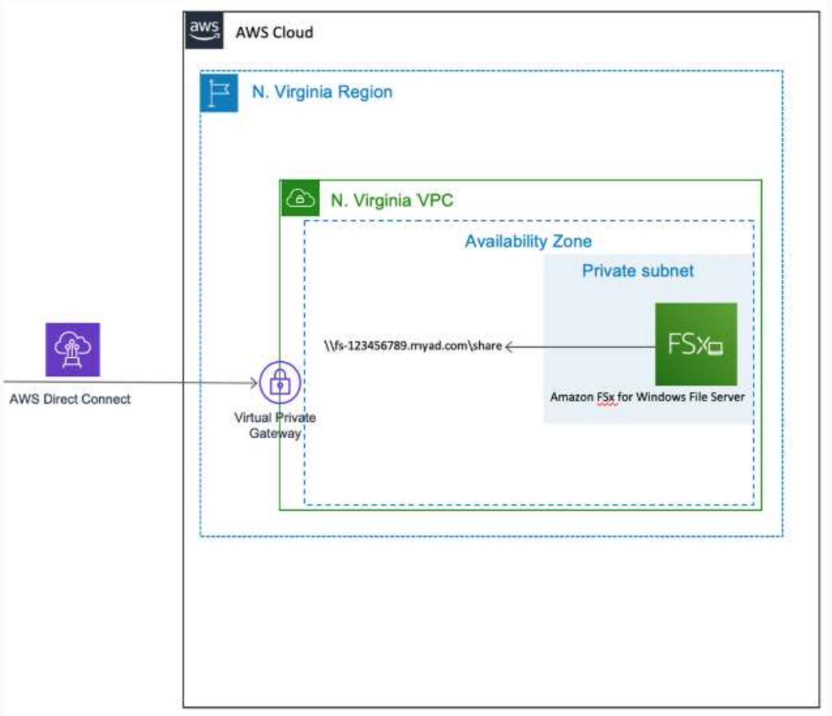
upon boot-up.启动时。



Amazon FSx for Windows File Server provides fully managed Microsoft Windows file servers, backed by a fully native Windows file system. FSx for Windows File Server has the features, performance, and compatibility to easily lift and shift enterprise applications to the AWS Cloud.

Amazon FSx supports a broad set of enterprise Windows workloads with fully managed file storage built on Microsoft Windows Server. Amazon FSx has native support for Windows file system features and for the industry-standard **Server Message Block (SMB)** protocol to access file storage over a network.

As a fully managed service, FSx for Windows File Server eliminates the administrative overhead of setting up and provisioning file servers and storage volumes. Additionally, Amazon FSx keeps Windows software up to date, detects and addresses hardware failures, and performs backups.



AWS recommends using a staging environment with the same configuration as your production environment. For example, use the same Active Directory (AD) and networking configurations, file system size and configuration, and Windows features, such as data deduplication and shadow copies. Running test workloads in a staging environment that simulates your desired production traffic helps the process run smoothly.

The option that says: **Create an Amazon Machine Image (AMI) of the test Amazon EC2 instance. Use the AMI for an Auto Scaling group with a minimum size of three instances that spans three Availability Zones (AZs). Create an Amazon Elastic Filesystem (Amazon EFS) volume. Write a user data script to join the instances on the AD domain and mount the EFS share upon boot-up** is incorrect. Amazon EFS uses the NFS protocol which is primarily used by Linux AMIs. This filesystem does not support Windows ACLs.

The option that says: **Create an Amazon Machine Image (AMI) of the test Amazon EC2 instance. Use the AMI for an Auto Scaling group with a minimum size of three instances that spans three Availability Zones (AZs). Create an Amazon FSx for Lustre filesystem that will be used for shared storage. Write a user data script to join the instances on the AD domain and mount the EFS share upon boot-up** is incorrect. Amazon FSx for Lustre is POSIX-compliant file system that runs on Lustre. It can only be used by Linux-based instances.

The option that says: **Deploy a new Windows AMI for an Auto Scaling group with a minimum size of three instances and spans across three Availability Zones (AZs). Use an Amazon EBS volume with Multi-Attach enabled to allow multiple Amazon EC2 instances to share the volume. Write a user data script to install the CMS application and join the instances to the AD domain** is incorrect. This is not an ideal solution because Multi-Attach EBS volumes can only be attached on instances within the same Availability Zone.

15 A cryptocurrency startup owns multiple AWS accounts which are all linked under AWS Organizations. Due to the financial nature of the business, the DevOps lead has been instructed by the CTO to prepare for IT auditing activities to meet industry compliance requirements.

Which of the following provides the most durable and secure logging solution that can be used to track changes made to all of the company's AWS resources globally?

(view) 1

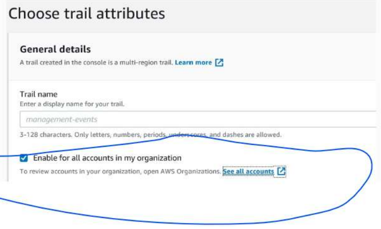
○ 1. Launch a new CloudTrail trail using the AWS console with an existing S3 bucket to store the logs and with the "Apply trail to all regions" checkbox 2. Enable MFA Delete on the S3 bucket.

○ 1. Launch a new CloudTrail trail using the AWS console with one new S3 bucket to store the logs and with the "Enable for all accounts in my organization" checkbox enabled. 2. Enable MFA Delete and Log Encryption on the S3 bucket.

○ 1. Launch a new CloudTrail with one new S3 bucket to store the logs. 2. Configure SNS to send log file delivery notifications to your management system. 3. Enable MFA Delete and Log Encryption on the S3 bucket.

○ 1. Launch three new CloudTrail trails using three new S3 buckets to store the logs for the AWS Management console, for AWS SDKs, and for the AWS CLI. 2. Enable MFA Delete and Log Encryption on the S3 bucket.

AWS CloudTrail is an AWS service that helps you enable governance, compliance, and operational and risk auditing of your AWS account. Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail. Events include actions taken in the AWS Management Console, AWS Command Line Interface, and AWS SDKs and APIs.



Choose trail attributes

General details

A trail created in the console is a multi-region trail. Learn more

Trail name
Enter a display name for your trail.
management-events
3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

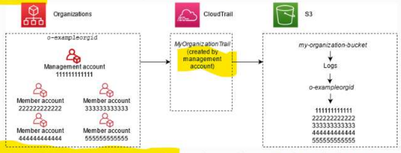
☑ Enable for all accounts in my organization
To review accounts in your organization, open AWS Organizations. See all accounts

CloudTrail is enabled on your AWS account when you create it. When activity occurs in your AWS account, that activity is recorded in a CloudTrail event. You can easily view recent events in the CloudTrail console by going to Event history. You can also enable the tracking of multi-region and global events. By default, the log files delivered by CloudTrail to your bucket are encrypted by Amazon server-side encryption with Amazon S3-managed encryption keys (SSE-S3). To provide a security layer that is directly manageable, you can instead use server-side encryption with AWS KMS keys (SSE-KMS) for your CloudTrail log files.

If you have created an organization in **AWS Organizations**, you can create a trail that will log all events for all AWS accounts in that organization. This is sometimes referred to as an **organization trail**. You can also choose to edit an existing trail in the management account and apply it to an organization, making it an organization trail. Organization trails log events for the management account, and all member accounts in the organization.

Organization trails are similar to regular trails in many ways. You can create multiple trails for your organization and choose whether to create an organization trail in all regions or a single region, and what kinds of events you want logged in your organization trail, just as in any other trail.

To create an organization trail, ensure that the "Enable for all accounts in my organization" option is checked when you create a new CloudTrail trail.



To protect your logs, you can encrypt the S3 bucket and add MFA Delete to protect your trail logs from accidental deletions. In this scenario, the following option is the best answer as it provides all of the things mentioned above:

1. **Launch a new CloudTrail trail using the AWS console with one new S3 bucket to store the logs and with the "Enable for all accounts in my organization" checkbox enabled.**
2. **Enable MFA Delete and Log Encryption on the S3 bucket.**

Organization trails are similar to regular trails in many ways. You can create multiple trails for your organization and choose whether to create an organization trail in all regions or a single region, and what kinds of events you want logged in your organization trail, just as in any other trail.组织跟踪在许多方面与常规跟踪类似。您可以为您的组织创建多个跟踪，并选择是创建覆盖所有区域还是单个区域的组织跟踪，以及选择要在组织跟踪中记录哪些类

型的事件，就像在任何其他跟踪中一样。

The following option is incorrect because although CloudTrail encrypts the data by default using SSE-S3, it is still more secure if you enabled log encryption and use SSE-KMS. Take note that the scenario asked for the most durable and secure logging solution:

1. **Launch a new CloudTrail trail using the AWS console with an existing S3 bucket to store the logs and with the "Apply trail to all regions" checkbox**
2. **Enable MFA Delete on the S3 bucket.**

The following option is incorrect because the multi-region option is not enabled which is needed to fetch all CloudTrail trail from all AWS regions:

1. **Launch a new CloudTrail with one new S3 bucket to store the logs.**
2. **Configure SNS to send log file delivery notifications to your management system.**
3. **Enable MFA Delete and Log Encryption on the S3 bucket.**

The following option is incorrect because this option creates too many S3 buckets that are unnecessary, whereas all of the events can be easily logged in just a single S3 bucket:

1. **Launch three new CloudTrail trails using three new S3 buckets to store the logs for the AWS Management console, for AWS SDKs, and for the AWS CLI.**
2. **Enable MFA Delete and Log Encryption on the S3 bucket.**

16 A company stores confidential files on an Amazon S3 bucket. There was a recent production incident in the company in which the files that are stored in an S3 bucket were accidentally made public. This has caused data leakage that affected the company revenue. The management has instructed the solutions architect to come up with a solution to safeguard the S3 bucket. The solution should only allow private files to be uploaded to the S3 bucket and no file should have a public read or public write access.

Which of the following options should the solutions architect implement to meet the above requirements with MINIMAL effort?

(view) 1

○ Set up a policy that restricts all `s3:PutObject` actions of the user to have a `private` canned ACL only which prohibits any public access to the uploaded objects.

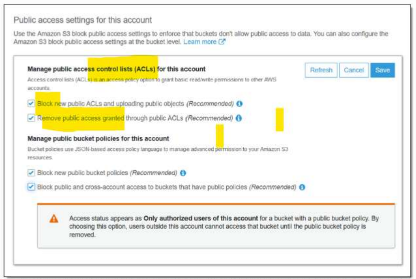
○ Use the `s3-bucket-public-read-prohibited` and `s3-bucket-public-write-prohibited` managed rules in AWS Config to restrict all users from uploading publicly accessible and writable files to the S3 bucket.

○ Set up AWS Organizations and create a new Service Control Policy (SCP) that will deny public objects from being uploaded to the Amazon S3 bucket. Attach the SCP to the AWS account.

● Enable Amazon S3 Block Public Access in the S3 bucket.

Amazon S3 provides Block Public Access settings for buckets and accounts to help you manage public access to Amazon S3 resources. By default, new buckets and objects don't allow public access, but users can modify bucket policies or object permissions to allow public access. Amazon S3 Block Public Access provides settings that override these policies and permissions so that you can limit public access to these resources.

With Amazon S3 Block Public Access, account administrators and bucket owners can easily set up centralized controls to limit public access to their Amazon S3 resources that are enforced regardless of how the resources are created.



Public access settings for this account

Use the Amazon S3 block public access settings to enforce that buckets don't allow public access to data. You can also configure the Amazon S3 block public access settings at the bucket level. Learn more

Manage public access control lists (ACLs) for this account

Access control lists (ACLs) is an access policy option to grant basic read/write permissions to other AWS accounts.

☑ Block new public ACLs and uploading public objects *(Recommended)*

☑ Remove public access granted through public ACLs *(Recommended)*

Manage public bucket policies for this account

Bucket policies use JSON-based access policy language to manage advanced permission to your Amazon S3 resources.

☑ Block new public bucket policies *(Recommended)*

☑ Block public and cross-account access to buckets that have public policies *(Recommended)*

Access status appears as **Only authorized users of this account** for a bucket with a public bucket policy. By choosing this option, users outside this account cannot access that bucket until the public bucket policy is removed.

When Amazon S3 receives a request to access a bucket or an object, it determines whether the bucket or the bucket owner's account has a Block Public Access setting. If there is an existing Block Public Access setting that prohibits the requested access, then Amazon S3 rejects the request. Amazon S3 Block Public Access provides four settings. These settings are independent and can be used in any combination, and each setting can be applied to a bucket or to an entire AWS account.

If a bucket has Block Public Access settings that are different from its owner's account, Amazon S3 applies the most restrictive combination of the bucket-level and account-level settings. Thus, when Amazon S3 evaluates whether an operation is prohibited by a Block Public Access setting, it rejects any request that would violate either a bucket-level or an account-level setting.

The option that says: **Set up a policy that restricts all `s3:PutObject` actions of the user to have a `private` canned ACL only which prohibits any public access to the uploaded objects** is incorrect. Although this solution is possible, it entails a lot of effort to set up an IAM policy that restricts the user from uploading public objects. Using the Amazon Block Public Access is a more suitable solution for this scenario.

The option that says: **Use the `s3-bucket-public-read-prohibited` and `s3-bucket-public-write-prohibited` managed rules in AWS Config to restrict all users from uploading publicly accessible and writable files to the S3 bucket** is incorrect. This solution with AWS Config will only notify you and your team of public objects in the S3 bucket. It would not be able to restrict any user from uploading public objects.

The option that says: **Set up AWS Organizations and create a new Service Control Policy (SCP) that will deny public objects from being uploaded to the Amazon S3 bucket, then attaching the SCP to the AWS account** is incorrect. Although you can satisfy the requirement using a service control policy (SCP), it still entails a lot of effort to implement. Remember that the scenario asks you to meet the requirements with minimal effort. Enabling the Amazon S3 Block Public Access in the S3 bucket is still the easiest one to implement. An SCP is primarily used to determine what services and actions can be delegated by administrators to the users and roles in the accounts that the SCP is applied to.

18 A big fast-food chain in Asia is planning to implement a location-based alert on their existing mobile app. If a user is in proximity to one of its restaurants, an alert will be shown on the user's mobile phone. The notification needs to happen in less than a minute while the user is still in the vicinity. Currently, the mobile app has 10 million users in the Philippines, China, Korea, and other Asian countries.

Which one of the following AWS architecture is the most suitable option for this scenario?

(view) 1 1 0 00:00:00

○ Set up an API that uses an Application Load Balancer and an Auto Scaling group of EC2 instances. The mobile app will send the user's location data to the API web service. Use DynamoDB to store and retrieve relevant offers on the nearest restaurant. Configure the EC2 instances to push alerts to the mobile app.

○ Establish connectivity with mobile carriers using AWS Direct Connect. Set up an API on all EC2 instances to receive the location data from the mobile app via the carrier's GPS connection. Use RDS to store the data and fetch relevant offers from the restaurant. The EC2 instances will communicate with mobile carriers to send alerts to the mobile app.

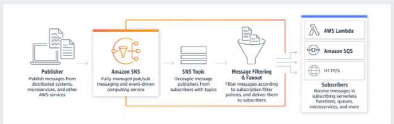
● The mobile app will send device location to an SQS endpoint. Set up an API that utilizes an Application Load Balancer and an Auto Scaling group of EC2 instances, which will retrieve the relevant offers from DynamoDB. Use Amazon SNS to send offers to the mobile app.

○ The mobile app will send the real-time location data using Amazon Kinesis. Set up an API which uses an Application Load Balancer and an Auto Scaling group of EC2 instances to retrieve the relevant offers from a DynamoDB table. Use Amazon Lambda and SES to push the notification to the mobile app.

Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. Using SQS, you can send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available.

Amazon Simple Notification Service (SNS) is a highly available, durable, secure, fully managed pub/sub messaging service that enables you to decouple microservices, distributed systems, and serverless applications. Additionally, SNS can be used to fan out notifications to end users using mobile push, SMS, and email.

With the **Mobile Push feature of Amazon SNS**, you have the ability to send push notification messages directly to apps on mobile devices. Push notification messages sent to a mobile endpoint can appear in the mobile app as message alerts, badge updates, or even sound alerts.


Publisher
Publish messages from distributed systems, microservices, and other AWS services
Amazon SNS
Fully-managed pub/sub messaging and event-driven computing service
SNS Topic
Decouple message publishers from subscribers with topics
Message Filtering & Fanout
Filter messages according to subscription filter policies, and deliver them to subscribers
AWS Lambda
Amazon SQS
HTTP/S
Subscribers
Receive messages in subscribing serverless functions, queues, microservices, and more

The option that says: **Set up an API that uses an Application Load Balancer and an Auto Scaling group of EC2 instances. The mobile app will send the user's location data to the API web service. Use DynamoDB to store and retrieve relevant offers on the nearest restaurant. Configure the EC2 instances to push alerts to the mobile app** is incorrect. Using EC2 instances to push alerts to the mobile app is not an appropriate solution. You have to use the AWS Mobile Push feature of SNS.

The option that says: **Establish connectivity with mobile carriers using AWS Direct Connect. Set up an API on all EC2 instances to receive the location data from the mobile app via the carrier's GPS connection. Use RDS to store the data and fetch relevant offers from the restaurant. The EC2 instances will communicate with mobile carriers to send alerts to the mobile app** is incorrect. AWS Direct Connect is primarily used to establish a dedicated network connection from your premises to AWS.

The option that says: **The mobile app will send the real-time location data using Amazon Kinesis. Set up an API which uses an Application Load Balancer and an Auto Scaling group of EC2 instances to retrieve the relevant offers from a DynamoDB table. Use Amazon Lambda and SES to push the notification to the mobile app** is incorrect. You can't use SES to send push notifications to mobile phones. You have to use SNS instead.

19 A company manually runs its custom scripts when deploying a new version of its application that is hosted on a fleet of Amazon EC2 instances. This method is prone to human errors, such as accidentally running the wrong script or deploying the wrong artifact. The company wants to automate its deployment procedure.

If errors are encountered after the deployment, the company wants to be able to roll back to the older application version as fast as possible.

Which of the following options should the Solutions Architect implement to meet the requirements?

(view) 1 0 1 00:00:00

● Create two identical environments of the application on AWS Elastic Beanstalk. Use a blue/green deployment strategy by swapping the environment's URL. Deploy the custom scripts using Elastic Beanstalk platform hooks.

○ Create a new pipeline on AWS CodePipeline and add a stage that will deploy the application on the EC2 instances. Choose a "rolling update with an additional batch" deployment strategy, to allow a quick rollback to the older version in case of errors.

○ Utilize AWS CodeBuild and add a job with Chef recipes for the new application version. Use a "canary" deployment strategy to the new version on a new instance. Delete the canary instance if errors are found on the new version.

○ Create an AWS System Manager automation runbook to manage the deployment process. Set up the runbook to first deploy the new application version to a staging environment. Include automated tests and, upon successful completion, use the runbook to deploy the application to the production environment

Create two identical environments of the application on AWS Elastic Beanstalk. Use a blue/green deployment strategy by swapping the environment's URL. Deploy the custom scripts using Elastic Beanstalk platform hooks钩子

.在 AWS Elastic Beanstalk 上为应用程序创建两个相同的环境。通过交换环境的 URL 来使用蓝绿部署策略。使用 Elastic Beanstalk 平台钩子部署自定义脚本

AWS Elastic Beanstalk。
AWS Elastic Beanstalk is a platform-as-a-service (PaaS) offering from Amazon Web Services (AWS) that makes it easy to **deploy, manage, and scale web applications and APIs — without needing to manage the underlying infrastructure (like EC2 instances, load balancers, or auto scaling groups) directly.**
Elastic Beanstalk automatically handles:

1. Provisioning servers (EC2 instances)
2. Load balancing
3. Auto scaling
4. Monitoring and health checks
5. Deployment of your code

You just upload your application code — Elastic Beanstalk does the rest.

⚙ How it works:
You upload your code (ZIP file or from GitHub, CLI, etc.).
Elastic Beanstalk creates an environment that includes:

1. EC2 instances to run your app
2. Load balancer (if needed)
3. Auto Scaling Group
4. CloudWatch for monitoring

Your **app** is deployed automatically — you can focus on coding, not infrastructure.

🧩 Supported platforms:
Elastic Beanstalk supports multiple languages and frameworks, such as:
Python
Node.js
Java
.NET
Go
PHP
Ruby
Docker
You can even deploy containerized apps using Docker.

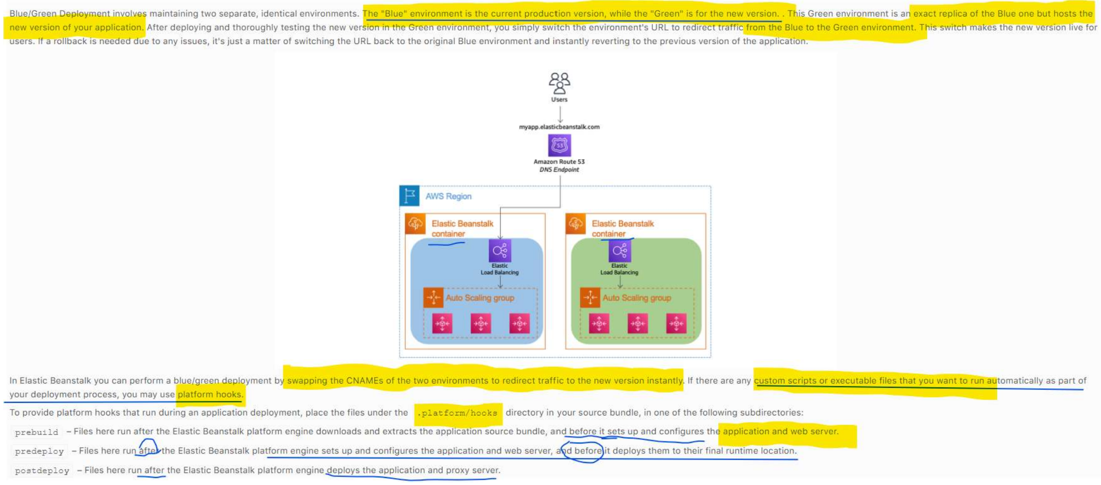
Your Code
↓
Elastic Beanstalk
↓
EC2 + Load Balancer + Auto Scaling + CloudWatch + S3 + RDS (optional)

1. No manual infrastructure management
2. Automatic scaling

3. Easy rollback & version control
4. Integration with CI/CD
5. Full control if needed (you can still access underlying resources)

Suppose you've built a Python Flask web app.
You can:

1. Zip your app files.
2. Upload them to Elastic Beanstalk.
3. It will automatically:
    1. Create an EC2 instance,
    2. Install Python and dependencies,
    3. Set up a load balancer,
    4. And expose your app's URL (like myapp.us-east-1.elasticbeanstalk.com).



Blue/Green Deployment involves maintaining two separate, identical environments. The "Blue" environment is the current production version, while the "Green" is for the new version. . This Green environment is an exact replica of the Blue one but hosts the new version of your application. After deploying and thoroughly testing the new version in the Green environment, you simply switch the environment's URL to redirect traffic from the Blue to the Green environment. This switch makes the new version live for users. If a rollback is needed due to any issues, it's just a matter of switching the URL back to the original Blue environment and instantly reverting to the previous version of the application.

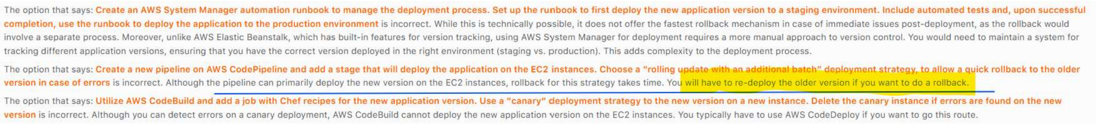
In Elastic Beanstalk you can perform a blue/green deployment by swapping the CNAMEs of the two environments to redirect traffic to the new version instantly. If there are any custom scripts or executable files that you want to run automatically as part of your deployment process, you may use platform hooks.

To provide platform hooks that run during an application deployment, place the files under the `.platform/hooks` directory in your source bundle, in one of the following subdirectories:

`prebuild` – Files here run after the Elastic Beanstalk platform engine downloads and extracts the application source bundle, and before it sets up and configures the application and web server.

`predeploy` – Files here run after the Elastic Beanstalk platform engine sets up and configures the application and web server, and before it deploys them to their final runtime location.

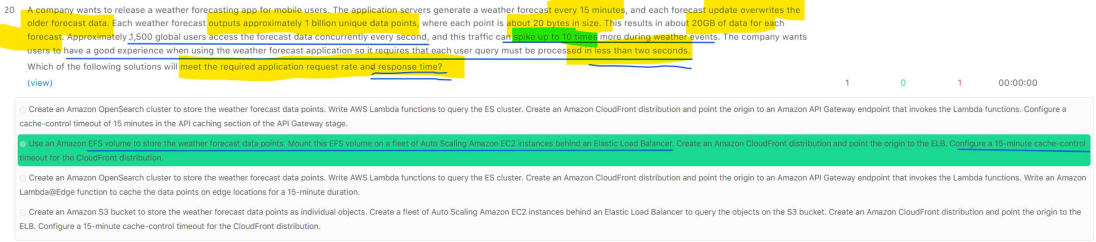
`postdeploy` – Files here run after the Elastic Beanstalk platform engine deploys the application and proxy server.



The option that says: **Create an AWS System Manager automation runbook to manage the deployment process. Set up the runbook to first deploy the new application version to a staging environment. Include automated tests and, upon successful completion, use the runbook to deploy the application to the production environment** is incorrect. While this is technically possible, it does not offer the fastest rollback mechanism in case of immediate issues post-deployment, as the rollback would involve a separate process. Moreover, unlike AWS Elastic Beanstalk, which has built-in features for version tracking, using AWS System Manager for deployment requires a more manual approach to version control. You would need to maintain a system for tracking different application versions, ensuring that you have the correct version deployed in the right environment (staging vs. production). This adds complexity to the deployment process.

The option that says: **Create a new pipeline on AWS CodePipeline and add a stage that will deploy the application on the EC2 instances. Choose a "rolling update with an additional batch" deployment strategy, to allow a quick rollback to the older version in case of errors** is incorrect. Although the pipeline can primarily deploy the new version on the EC2 instances, rollback for this strategy takes time. You will have to re-deploy the older version if you want to do a rollback.

The option that says: **Utilize AWS CodeBuild and add a job with Chef recipes for the new application version. Use a "canary" deployment strategy to the new version on a new instance. Delete the canary instance if errors are found on the new version** is incorrect. Although you can detect errors on a canary deployment, AWS CodeBuild cannot deploy the new application version on the EC2 instances. You typically have to use AWS CodeDeploy if you want to go this route.



20 A company wants to release a weather forecasting app for mobile users. The application servers generate a weather forecast every 15 minutes, and each forecast update overwrites the older forecast data. Each weather forecast outputs approximately 1 billion unique data points, where each point is about 20 bytes in size. This results in about 20GB of data for each forecast. Approximately 1,500 global users access the forecast data concurrently every second, and this traffic can spike up to 10 times more during weather events. The company wants users to have a good experience when using the weather forecast application so it requires that each user query must be processed in less than two seconds.

Which of the following solutions will meet the required application request rate and response time?

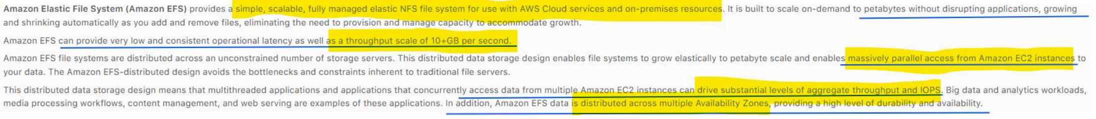
(view) 1 0 1 00:00:00

○ Create an Amazon OpenSearch cluster to store the weather forecast data points. Write AWS Lambda functions to query the ES cluster. Create an Amazon CloudFront distribution and point the origin to an Amazon API Gateway endpoint that invokes the Lambda functions. Configure a cache-control timeout of 15 minutes in the API caching section of the API Gateway stage.

● Use an Amazon EFS volume to store the weather forecast data points. Mount this EFS volume on a fleet of Auto Scaling Amazon EC2 instances behind an Elastic Load Balancer. Create an Amazon CloudFront distribution and point the origin to the ELB. Configure a 15-minute cache-control timeout for the CloudFront distribution.

○ Create an Amazon OpenSearch cluster to store the weather forecast data points. Write AWS Lambda functions to query the ES cluster. Create an Amazon CloudFront distribution and point the origin to an Amazon API Gateway endpoint that invokes the Lambda functions. Write an Amazon Lambda@Edge function to cache the data points on edge locations for a 15-minute duration.

○ Create an Amazon S3 bucket to store the weather forecast data points as individual objects. Create a fleet of Auto Scaling Amazon EC2 instances behind an Elastic Load Balancer to query the objects on the S3 bucket. Create an Amazon CloudFront distribution and point the origin to the ELB. Configure a 15-minute cache-control timeout for the CloudFront distribution.



Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on-demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.

Amazon EFS can provide very low and consistent operational latency as well as a throughput scale of 10+GB per second.

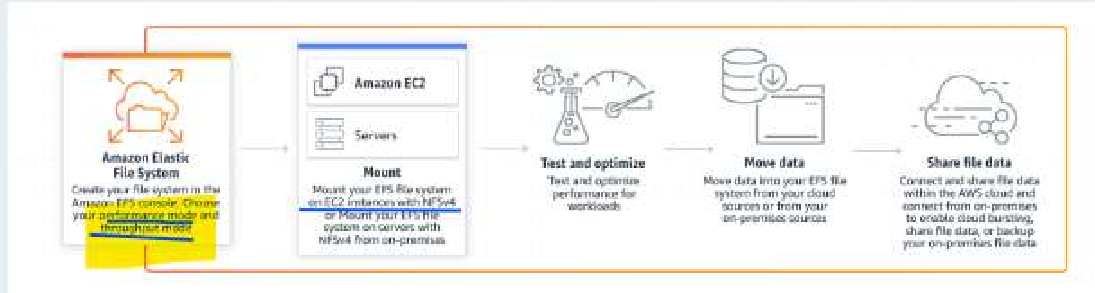
Amazon EFS file systems are distributed across an unconstrained number of storage servers. This distributed data storage design enables file systems to grow elastically to petabyte scale and enables massively parallel access from Amazon EC2 instances to your data. The Amazon EFS-distributed design avoids the bottlenecks and constraints inherent to traditional file servers.

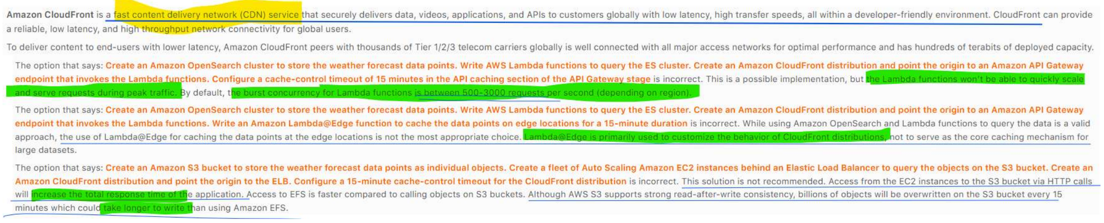
This distributed data storage design means that multithreaded applications and applications that concurrently access data from multiple Amazon EC2 instances can drive substantial levels of aggregate throughput and IOPS. Big data and analytics workloads, media processing workflows, content management, and web serving are examples of these applications. In addition, Amazon EFS data is distributed across multiple Availability Zones, providing a high level of durability and availability.

Amazon Elastic File System (Amazon EFS) 提供简单、可扩展且完全托管的弹性 NFS 文件系统，可与 AWS 云服务和本地资源配合使用。它旨在按需扩展到 PB 级，且不会中断应用程序运行，并随着文件的添加和删除而自动扩展和缩减，无需预置和管理容量以适应增长。

Amazon EFS 可以提供极低且一致的操作延迟，以及每秒 **10+GB** 的吞吐量扩展。

Amazon EFS 文件系统分布在不受数量限制的存储服务器上。这种分布式数据存储设计使文件系统能够弹性扩展到 PB 级，并支持从 Amazon **EC2 实例大规模并行访问您的数据**。Amazon EFS 分布式设计避免了传统文件服务器的瓶颈和固有限制。
这种分布式数据存储设计意味着多线程应用程序以及同时从多个 Amazon EC2 实例访问数据的应用程序可以实现显著的聚合吞吐量和 IOPS。大数据和分析工作负载、媒体处理工作流、内容管理和 Web 服务就是这些应用程序的示例。此外，**Amazon EFS 数据分布在多个可用区，**提供高水平的持久性和可用性。
mount your efs file systems挂载你的 efs 文件系统


Amazon Elastic File System
Create your file system in the Amazon EFS console. Choose your performance mode and throughput mode
Amazon EC2
Servers
Mount
Mount your EFS file system on EC2 instances with NFSv4 or Mount your EFS file system on servers with NFSv4 from on-premises
Test and optimize
Test and optimize performance for workloads
Move data
Move data into your EFS file system from your cloud sources or from your on-premises sources
Share file data
Connect and share file data within the AWS cloud and connect from on-premises to enable cloud bursting, share file data, or backup your on-premises file data



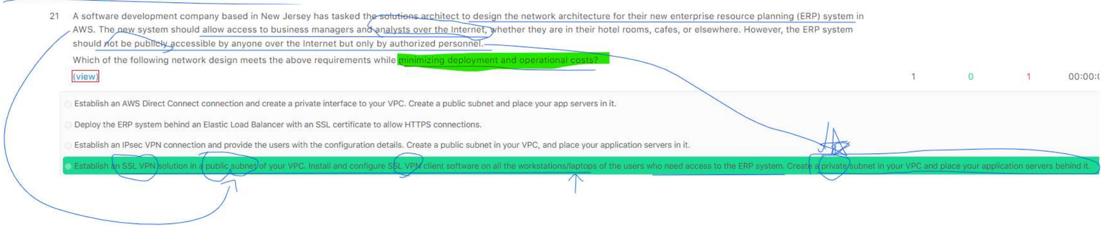
Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment. CloudFront can provide a reliable, low latency, and high throughput network connectivity for global users.

To deliver content to end-users with lower latency, Amazon CloudFront peers with thousands of Tier 1/2/3 telecom carriers globally is well connected with all major access networks for optimal performance and has hundreds of terabits of deployed capacity.

The option that says: **Create an Amazon OpenSearch cluster to store the weather forecast data points. Write AWS Lambda functions to query the ES cluster. Create an Amazon CloudFront distribution and point the origin to an Amazon API Gateway endpoint that invokes the Lambda functions. Configure a cache-control timeout of 15 minutes in the API caching section of the API Gateway stage** is incorrect. This is a possible implementation, but the Lambda functions won't be able to quickly scale and serve requests during peak traffic. By default, the burst concurrency for Lambda functions is between 500-3000 requests per second (depending on region).

The option that says: **Create an Amazon OpenSearch cluster to store the weather forecast data points. Write AWS Lambda functions to query the ES cluster. Create an Amazon CloudFront distribution and point the origin to an Amazon API Gateway endpoint that invokes the Lambda functions. Write an Amazon Lambda@Edge function to cache the data points on edge locations for a 15-minute duration** is incorrect. While using Amazon OpenSearch and Lambda functions to query the data is a valid approach, the use of Lambda@Edge for caching the data points at the edge locations is not the most appropriate choice. Lambda@Edge is primarily used to customize the behavior of CloudFront distributions, not to serve as the core caching mechanism for large datasets.

The option that says: **Create an Amazon S3 bucket to store the weather forecast data points as individual objects. Create a fleet of Auto Scaling Amazon EC2 instances behind an Elastic Load Balancer to query the objects on the S3 bucket. Create an Amazon CloudFront distribution and point the origin to the ELB. Configure a 15-minute cache-control timeout for the CloudFront distribution** is incorrect. This solution is not recommended. Access from the EC2 instances to the S3 bucket via HTTP calls will increase the total response time of the application. Access to EFS is faster compared to calling objects on S3 buckets. Although AWS S3 supports strong read-after-write consistency, billions of objects will be overwritten on the S3 bucket every 15 minutes which could take longer to write than using Amazon EFS.



21 A software development company based in New Jersey has tasked the solutions architect to design the network architecture for their new enterprise resource planning (ERP) system in AWS. The new system should allow access to business managers and analysts over the Internet, whether they are in their hotel rooms, cafes, or elsewhere. However, the ERP system should not be publicly accessible by anyone over the Internet but only by authorized personnel.

Which of the following network design meets the above requirements while minimizing deployment and operational costs?

(view)

1 0 1 00:00:(

○ Establish an AWS Direct Connect connection and create a private interface to your VPC. Create a public subnet and place your app servers in it.

○ Deploy the ERP system behind an Elastic Load Balancer with an SSL certificate to allow HTTPS connections.

○ Establish an IPsec VPN connection and provide the users with the configuration details. Create a public subnet in your VPC, and place your application servers in it.

● Establish an SSL VPN solution in a public subnet of your VPC. Install and configure SSL VPN client software on all the workstations/laptops of the users who need access to the ERP system. Create a private subnet in your VPC and place your application servers behind it.

Conceito básico de VPC

Uma VPC (Virtual Private Cloud) é uma rede virtual isolada dentro da AWS onde você pode lançar recursos (como EC2, RDS, Lambda, etc.).
Dentro dessa VPC, você cria subnets (sub-redes) — que podem ser públicas ou privadas, dependendo da forma como elas se conectam à Internet.

🌐 1. Public Subnet

Uma public subnet é uma sub-rede com acesso direto à Internet.
☑ Requisitos para ser pública:
A subnet está associada a uma Route Table que aponta para o Internet Gateway (IGW).
As instâncias dentro dela têm endereços IP públicos ou Elastic IPs.
💡 Exemplo prático:

Imagine uma VPC 10.0.0.0/16 com duas subnets:
10.0.1.0/24 → Public Subnet
10.0.2.0/24 → Private Subnet

Você cria um Internet Gateway e o associa à VPC.
Na route table da public subnet, há uma rota:
Destination: 0.0.0.0/0

Target: igw-123abc

➡️ Isso significa que o tráfego de saída vai pra Internet via o Internet Gateway.

💻 Exemplo de uso:
Servidores web (EC2) que precisam ser acessados externamente.
Exemplo: um site hospedado em um EC2 público.
O EC2 recebe um IP público e pode ser acessado via navegador:
http://54.210.xxx.xxx

🔒 2. Private Subnet

Uma private subnet é uma sub-rede sem acesso direto à Internet.
🚫 Requisitos:

A route table não aponta para o Internet Gateway.
As instâncias não têm IP público.
Mas… elas ainda podem acessar a Internet indiretamente usando um NAT Gateway (que fica em uma public subnet).

💡 Exemplo prático:
Na private subnet (10.0.2.0/24), a route table tem:
Destination: 0.0.0.0/0
Target: nat-123abc

O NAT Gateway está em uma public subnet e faz a ponte para a Internet.

➡️ Assim, instâncias privadas podem baixar pacotes (outbound), mas não podem ser acessadas de fora (inbound).

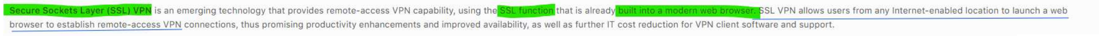
💻 Exemplo de uso:
Bancos de dados (RDS)
Servidores de aplicação internos
Processos ETL, back-end, ou APIs internas
Esses recursos não precisam (nem devem) estar acessíveis publicamente.

🧭 Cenário típico de arquitetura
VPC (10.0.0.0/16)

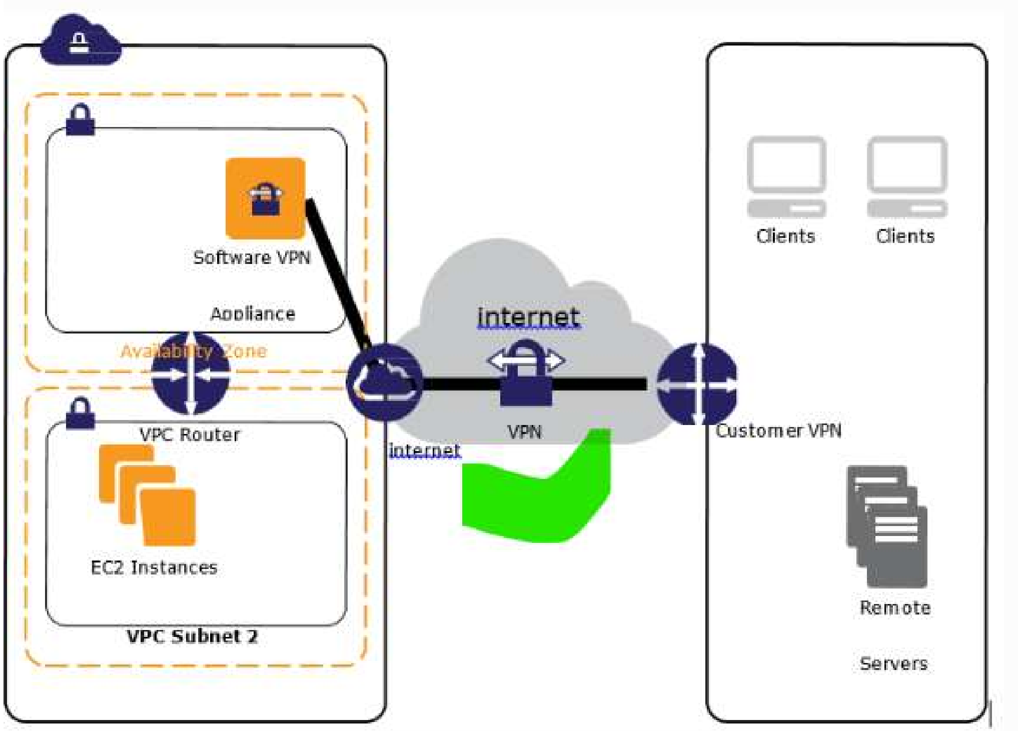
```
 │
 ├──── Public Subnet (10.0.1.0/24)
 │      ├──── EC2 Web Server (IP público)
 │      └──── NAT Gateway
 │
 └──── Private Subnet (10.0.2.0/24)
        ├──── RDS Database
        └──── EC2 App Server (sem IP público)
```

👉 O app server acessa a Internet via NAT Gateway.
👉 O usuário acessa o site via Internet Gateway → Public Subnet → EC2 Web Server.
👉 O banco de dados nunca fica exposto à Internet.

Característica	Public Subnet	Private Subnet
Acesso direto à Internet	☑ Sim (via IGW)	✖ Não
Precisa de IP público	☑ Sim	✖ Não
Pode acessar a Internet?	☑ Sim	☑ Sim (via NAT Gateway)
Pode ser acessada da Internet?	☑ Sim	✖ Não
Exemplo de uso	Servidores web, bastion host	Bancos de dados, app servers

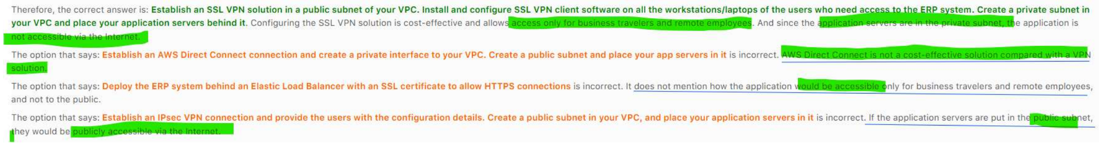


Secure Sockets Layer (SSL) VPN is an emerging technology that provides remote-access VPN capability, using the SSL function that is already built into a modern web browser. SSL VPN allows users from any Internet-enabled location to launch a web browser to establish remote-access VPN connections, thus promising productivity enhancements and improved availability, as well as further IT cost reduction for VPN client software and support.



Always keep in mind to i**nstall the SSL VPN software in your EC2 instances which are deployed in the public subnet.** This will be where your users can connect to the Internet to be able to access t**he business applications, which is deployed in the private subnet of your VPC.**

ssl vpn -> ec2 in public subnet(web site de compras onde os clientes acessam via internet) -> business applications in private subnet



Therefore, the correct answer is: **Establish an SSL VPN solution in a public subnet of your VPC. Install and configure SSL VPN client software on all the workstations/laptops of the users who need access to the ERP system. Create a private subnet in your VPC and place your application servers behind it**. Configuring the SSL VPN solution is cost-effective and allows access only for business travelers and remote employees. And since the application servers are in the private subnet, the application is not accessible via the Internet.

The option that says: **Establish an AWS Direct Connect connection and create a private interface to your VPC. Create a public subnet and place your app servers in it** is incorrect. AWS Direct Connect is not a cost-effective solution compared with a VPN solution.

The option that says: **Deploy the ERP system behind an Elastic Load Balancer with an SSL certificate to allow HTTPS connections** is incorrect. It does not mention how the application would be accessible only for business travelers and remote employees, and not to the public.

The option that says: **Establish an IPsec VPN connection and provide the users with the configuration details. Create a public subnet in your VPC, and place your application servers in it** is incorrect. If the application servers are put in the public subnet, they would be publicly accessible via the Internet.