8-simulacoes profissionais

Notebook:	solutions profissional_ef8e1c0f-5d3c-42ab-ab04-9217e53e12cd		
Created:	21/10/2025 02:40	**Updated:**	21/10/2025 03:03
Author:	erikachen19@gmail.com		
URL:	https://portal.tutorialsdojo.com/courses/aws-certified-solutions-architect-professional...		

9 A multinational corporation has recently acquired a smaller company. The solutions architect was instructed to consolidate the multiple AWS accounts of both entities using AWS Organizations. The solutions architect has set up the required service control policies (SCPs) to simplify the process of controlling access permissions for each individual account and Organizational Units (OUs). However, one account is having trouble creating a new S3 bucket, and it is required to investigate the cause of this issue. The account has the following SCP attached:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloudtrail:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*"
    }
  ]
}
```

Each IAM user of the account has the following IAM policy attached:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::*"
            ]
        }
    ]
}
```

Based on the provided SCP and IAM policy, which of the following options could be the possible root cause of this problem?

- The SCP is the root cause since it does not explicitly allow the required action that would enable the account to create an S3 bucket.
- The SCP is the root cause because it does not support whitelisting actions of the AWS resources.
- The IAM policy is the root cause because you have denied user permissions to execute any S3-related actions.
- Both the IAM policy and the SCP are the problem. The SCP should explicitly allow S3 bucket creation in its policy and the IAM policy should exactly match the permissions of the SCP.

13 A global enterprise oversees at least six hundred AWS accounts via AWS Organizations. Recently, the governance team granted product owners the ability to configure Amazon S3 Access Points within the assigned accounts. These access points must be accessible only from within VPCs (Virtual Private Clouds), and any access from the Internet must be restricted. What is the most operationally scalable and manageable approach to enforce this control efficiently across the organization?

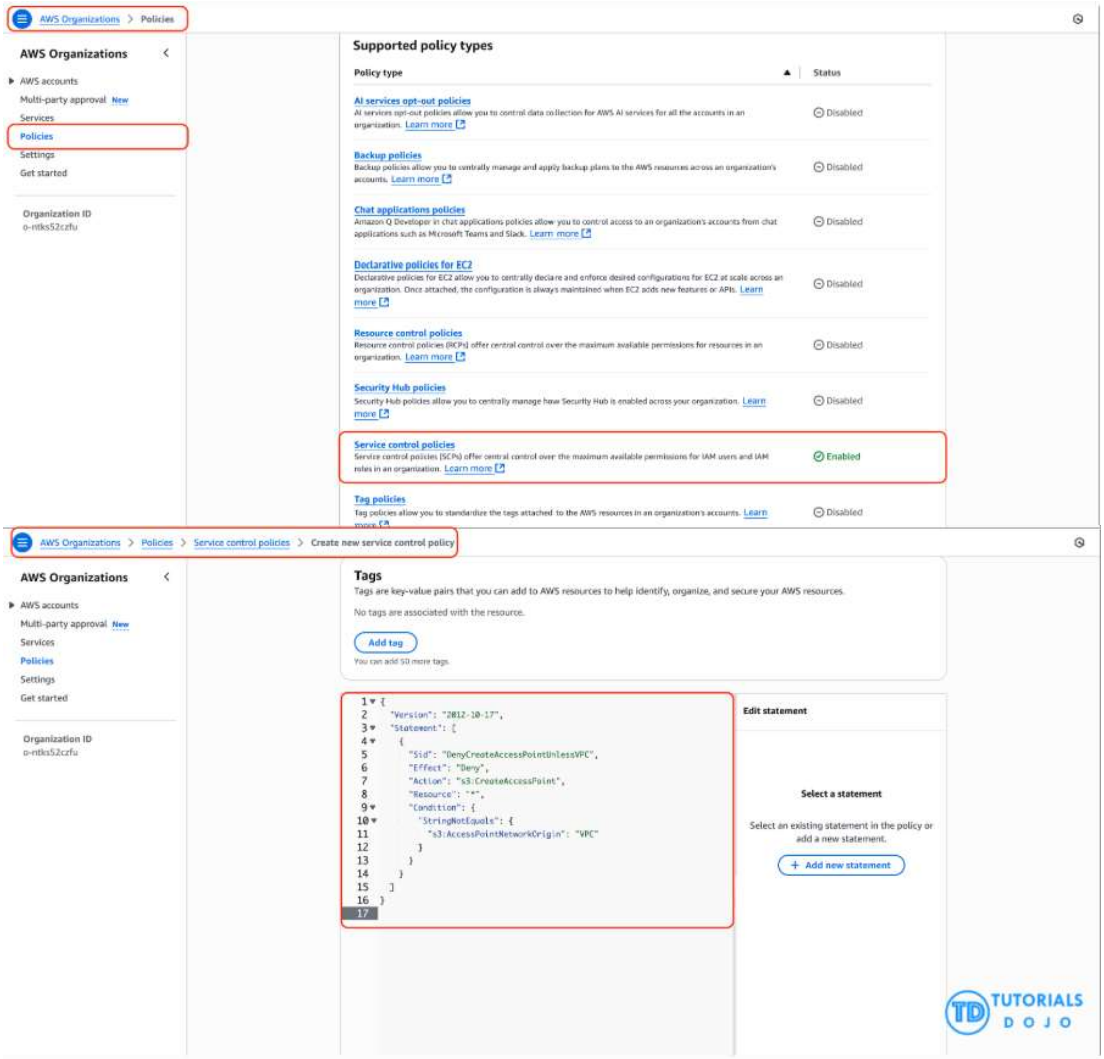
(view)

- Modify the bucket's resource policy to block any attempt to perform the `s3:CreateAccessPoint` operation, except when the `s3:ResourceAccount` condition is explicitly matched to VPC
- Use AWS Config rules combined with AWS Lambda to detect and revert S3 Access Points that are not restricted to VPC-only access
- Use an S3 Object ACL to block the creation of access points unless the request originates from a VPC by checking the `s3:AccessPointNetworkOrigin` condition
- At the root level of the organization, implement a Service Control Policy (SCP) that blocks `s3:CreateAccessPoint` except if the `s3:AccessPointNetworkOrigin` is equal to VPC

一家跨国企业通过 AWS Organizations 管理至少六百个 AWS 账户。最近，治理团队授权产品所有者在指定账户内配置 Amazon S3 接入点。这些接入点必须只能从 VPC（虚拟私有云）内部访问，并且任何来自互联网的访问都必须受

到限制。在整个组织内有效实施此控制措施，最具运营可扩展性和可管理性的方法是什么？
在组织的根级别，实施服务控制策略 (SCP)，阻止 s3:CreateAccessPoint，除非 s3:AccessPointNetworkOrigin 等于 VPC

Service control policies (SCPs) are a type of organization policy that you can use to manage permissions in your organization. SCPs offer central control over the maximum available permissions for the IAM users and IAM roles in your organization. SCPs help ensure your accounts stay within your organization's access control guidelines.
SCPs do not grant permissions to your organization's IAM users and IAM roles. An SCP defines a **permission guardrail** or limits the actions your organization's IAM users and IAM roles can perform.



By applying this SCP at the root level, the enterprise ensures that no user or role in any account, regardless of their IAM policies, can create an S3 Access Point unless it is explicitly configured to be VPC-only. This method is highly scalable and manageable because it requires a single policy update at the organizational root to enforce the rule across hundreds of accounts. It also aligns perfectly with the requirement to restrict access to S3 Access Points from the internet, thereby ensuring compliance and reducing the risk of misconfiguration by individual product teams.

Hence, the correct answer is: **At the root level of the organization, implement a Service Control Policy (SCP) that blocks `s3:CreateAccessPoint` except if the `s3:AccessPointNetworkOrigin` is equal to VPC.**

The option that says: **Modify the bucket's resource policy to block any attempt to perform the `s3:CreateAccessPoint` operation, except when the `s3:ResourceAccount` condition is explicitly matched to VPC** is incorrect. First, S3 bucket policies only control access to bucket-level and object-level operations (such as `s3:GetObject`, `s3:PutObject`, or `s3:ListBucket`). They do not govern actions related to creating or managing S3 Access Points, such as `s3:CreateAccessPoint`. That action is handled at the account level, not the bucket level, so attempting to restrict it through a bucket policy is fundamentally ineffective. Second, using the `s3:ResourceAccount` condition key is also inappropriate. `s3:ResourceAccount` refers to the AWS account that owns the resource, not the network origin or access type. The correct condition key for enforcing network-based restrictions during access point creation is `s3:AccessPointNetworkOrigin`, not `s3:ResourceAccount`. Therefore, this option fails on both policy scope and condition logic levels, making it an invalid approach for enforcing VPC-only access point creation across an organization.

The option that says: **Use AWS Config rules combined with AWS Lambda to detect and revert S3 Access Points that are not restricted to VPC-only access** is incorrect. While this method is technically feasible and can enforce compliance after the fact, it is only reactive and not preventive. AWS Config rules monitor resource configurations and can trigger AWS Lambda functions for remediation, but they do not block the creation of non-compliant access points in real-time. This means an S3 Access Point that allows internet access could exist, even briefly, before being detected and corrected, which introduces operational and security risks. Additionally, deploying and maintaining AWS Config rules and remediating Lambda functions across hundreds of accounts requires significant effort. It involves ensuring that Config is enabled and configured correctly in each account, managing the lifecycle of custom remediation functions, handling errors or exceptions, and maintaining consistency. Compared to a centralized and proactive mechanism like a Service Control Policy (SCP), this method introduces more moving parts, complexity, and operational overhead. Therefore, while it may serve as a supplementary compliance check, it is not the most efficient or scalable approach to enforce VPC-only S3 Access Point creation across an enterprise-wide AWS Organizations structure.

The option that says: **Use an S3 Object ACL to block the creation of access points unless the request originates from a VPC by checking the `s3:AccessPointNetworkOrigin` condition** is incorrect. This approach reflects a misunderstanding of both S3 Object ACLs and the mechanics of S3 Access Point creation. S3 Object ACLs manage access control at the object level, allowing or denying access to specific users or AWS accounts for actions such as `s3:GetObject` or `s3:PutObject`. They cannot control IAM-level operations like `s3:CreateAccessPoint`, an account-level API call not associated with individual objects or even directly with the bucket. Furthermore, ACLs do not support conditional logic based on context keys like `s3:AccessPointNetworkOrigin`. That condition key applies only to IAM policies, SCPs, or resource-based policies on access points, not to ACLs. Therefore, this option is incorrect in its proposed enforcement mechanism and invalid in how it misuses a condition key in a context where it cannot be applied. It would not work at all to restrict access point creation, regardless of network origin, making it entirely ineffective for meeting the stated requirement.

12 A North American-based tech startup plans to expand to Europe due to the unexpected, overwhelming success of its software-as-a-service (SaaS) solution. A solutions architect consultant was hired to design and execute the expansion.

The company uses the `us-east-1` Region for its North American infrastructure. The system uses Amazon S3, which is accessed through Amazon CloudFront, as its front-end component. It uses Amazon API Gateway for the API endpoints. AWS Lambda functions connected to an Amazon RDS for MySQL serve as its backend and persistence layer. Amazon Route 53 manages the domain via a simple routing policy pointing to the API Gateway.

In anticipation of the upcoming expansion, the engineering team proactively provisioned `eu-west-1` Region infrastructure, including API Gateways and Lambda functions in the mentioned Region.

Initial measurements of the consultant suggest that around 95% of the database traffic is read-only.

What approach can the consultant take to expand on the existing `eu-west-1` Region and ensure minimal latency for users in the Europe Region?

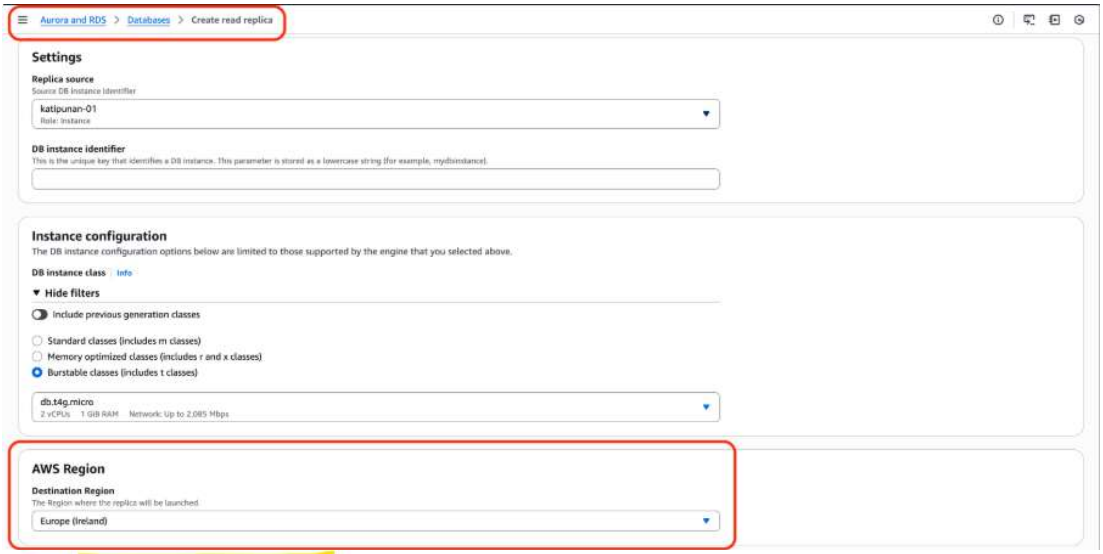
(view) 1 1 0 00:00:00

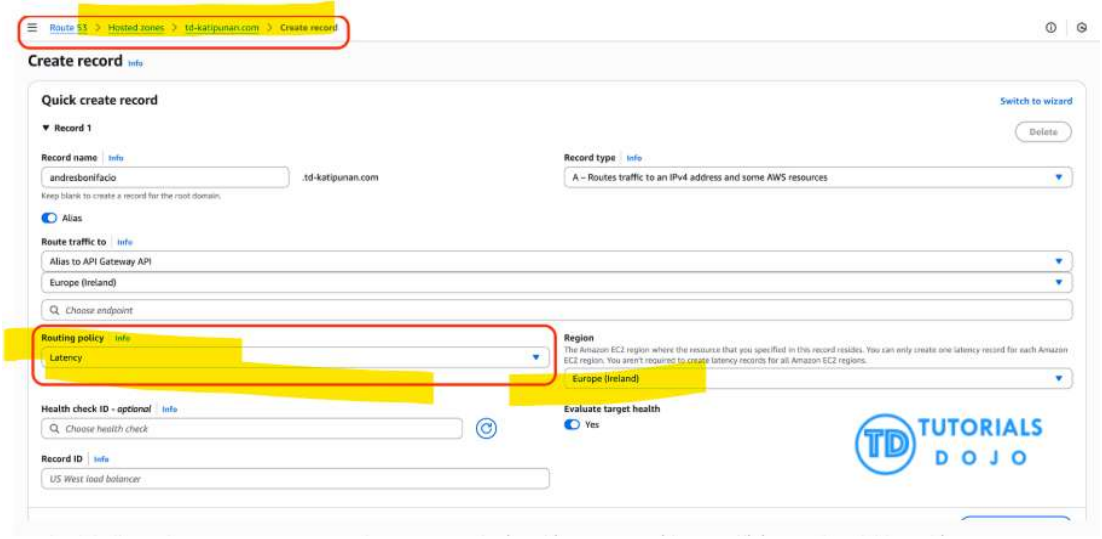
- Use latency-based routing in Route 53 to direct traffic to the appropriate API endpoint. Configure an AWS Database Migration Service (AWS DMS) task with change data capture (CDC) for replication of the main database from the `us-east-1` to `eu-west-1`.
- In Route 53, use latency-based routing. Set up a cross-Region read replica of the database in `eu-west-1`.
- Use failover routing in Route 53. Create a Multi-AZ replica of the primary database.
- Keep simple routing in Route 53 for minimal setup. Set up an AWS Database Migration Service (AWS DMS) task with full load combined with change data capture (CDC) to replicate the original database from `us-east-1` to `eu-west-1`.

To minimize latency for European users accessing a report-generation platform, the key objective is to bring data and computing closer to users in the European Region. The scenario highlights that over 95% of the database workload is read-only, indicating that read scalability through replication is a viable and preferred solution. Also, the infrastructure in `eu-west-1` (API Gateway + Lambda) has already been deployed, so we only need to ensure data consistency and fast access in that region.

We can reduce latency by the following approach:

1. **Cross-Region read replica** ensures that read-only queries (which comprise 90% of the load) are offloaded from the primary `us-east-1` database and served locally in `eu-west-1`, reducing cross-Region latency and data transfer.
2. **Latency-based routing** in Amazon Route 53 ensures that European users are routed to the fastest (often the nearest) API Gateway + Lambda stack in `eu-west-1`.





Since RDS cross-Region read replicas are designed for scaling reads globally, and Route 53 can route to the correct endpoint, this setup provides an efficient and scalable architecture.

Hence, the correct answer is: **In Route 53, use latency-based routing. Set up a cross-Region read replica of the database in `eu-west-1`.**

The option that says: **Use latency-based routing in Route53 to direct traffic to the appropriate API endpoint. Configure an AWS Database Migration Service (AWS DMS) task with change data capture (CDC) for replication of the main database from the `us-east-1` to `eu-west-1`** is incorrect. AWS DMS with CDC can replicate data across Regions, but it is not designed for real-time, low-latency read access in high-throughput applications. DMS is only meant for data migration, warehousing, or hybrid environments, not for scalable, low-latency read operations in production. Additionally, DMS adds complexity and operational overhead, including managing replication tasks and monitoring lag.

The option that says: **Use failover routing in Route 53. Create a Multi-AZ replica of the primary database** is incorrect. Failover routing in Route 53 is designed to improve availability during outages by routing traffic to a healthy endpoint if the primary fails. It does not facilitate latency-based traffic distribution between Regions. Additionally, Multi-AZ deployments are confined to a single Region. You cannot create a Multi-AZ read replica across Regions; they are designed to provide high availability within the same Region only.

The option that says: **Keep simple routing in Route 53 for minimal setup. Set up an AWS Database Migration Service (AWS DMS) task with full load combined with change data capture (CDC) to replicate the original database from `us-east-1` to `eu-west-1`** is incorrect. First, simple routing does not differentiate traffic based on Region or latency. All users are directed to a single endpoint, defeating the goal of optimizing for European users. Second, AWS DMS is not the right tool for real-time production read replicas, as with the first option. It introduces lag and does not scale efficiently for large volumes of read traffic.