Level: Advanced

Microsoft Azure Exam AZ-104 Certification

← Back to the **Course**

Azure App Service & Containers - **Practice Mode**

Completed on Sun, 16 Nov 2025

1st Attempt	1/5 Marks Obtained	20.00% Your Score

FAIL
Result

Download Report

Domain wise Quiz Performance Report

No.	Domain	Total Question	Correct	Incorrect	Unattempted
1	Deploy and manage Azure compute resources	5	1	4	0
Total	All Domains	5	1	4	0

Review the Answers

Filter By All Questions

Question 1 Incorrect

Domain: Deploy and manage Azure compute resources

Your organization requires all container images stored in Azure Container Registry (ACR) to be scanned for vulnerabilities before they are deployed. The solution must ensure that only approved images can proceed to production and provide reports on vulnerabilities.Which three configuration steps should you implement? (Select three)

Your Answer	Correct Answer
D. Configure a policy in Azure Policy to deny unscanned images	A. Enable Microsoft Defender for Containers to automatically scan ACR images
E. Set up a webhook in ACR to trigger vulnerability scans on image pushes	B. Configure ACR Tasks to run image scanning as part of the build process
B. Configure ACR Tasks to run image scanning as part of the build process	D. Configure a policy in Azure Policy to deny unscanned images

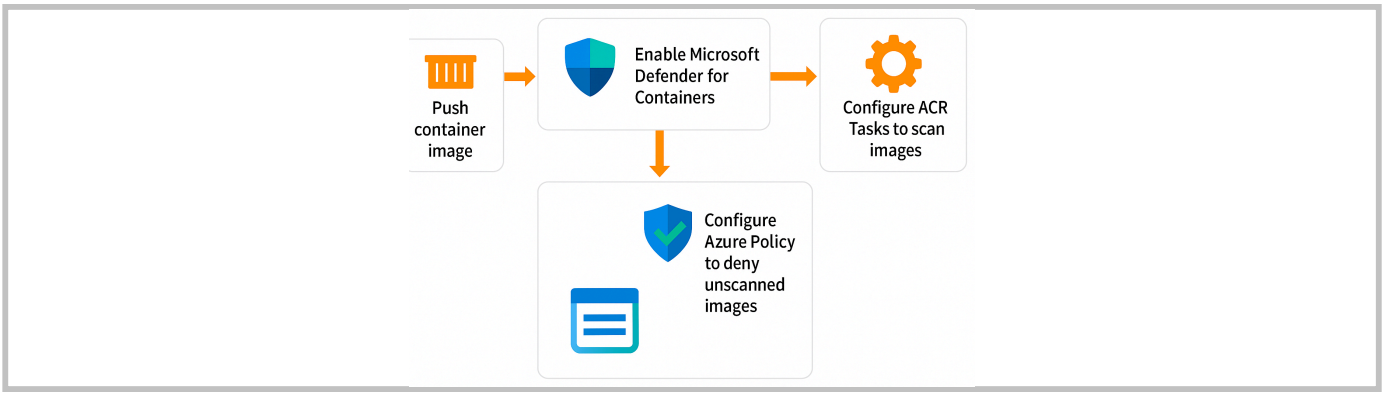
Explanation:

Correct Answers: A, B and D

Option A: Enable Microsoft Defender for Containers to automatically scan ACR images is correct because Microsoft Defender for Containers is the native Azure solution that provides automated vulnerability scanning for images in ACR. It delivers detailed reports and is integrated with Microsoft Defender for Cloud, making it the most efficient way to detect vulnerabilities.

Option B: Configure ACR Tasks to run image scanning as part of the build process is correct because ACR Tasks allow you to automate workflows, including image scanning, as part of your CI/CD pipeline. By scanning images during the build process, you catch vulnerabilities early, which is a key principle of "shift-left security."

Option D: Configure a policy in Azure Policy to deny unscanned images is correct because Azure Policy is the governance tool that provides the enforcement mechanism. By setting up a policy to deny deployments of images that have not been scanned or have critical vulnerabilities, you ensure that only compliant images are used in production environments.



Option C: Use a third-party scanning tool and integrate it with Azure Pipelines for image scanning is incorrect because while a third-party tool could be used, it is redundant when Microsoft Defender for Containers provides a native, integrated solution. Using third-party tools can add unnecessary complexity and cost.

Option E: Set up a webhook in ACR to trigger vulnerability scans on image pushes is incorrect because a webhook only notifies an external system about an event. It does not perform the vulnerability scan itself. You would still need to build and maintain an external scanning solution, which is less efficient than using the native Azure services.

References:

https://learn.microsoft.com/en-us/azure/defender-for-cloud/agentless-vulnerability-assessment-azure

https://learn.microsoft.com/en-us/azure/container-registry/container-registry-tasks-overview

https://learn.microsoft.com/en-us/azure/governance/policy/concepts/effect-deny

Ask our Experts

Did you like this **Question?**

Question 2 Incorrect

Ask Whizzy AI

Domain: Deploy and manage Azure compute resources

An e-commerce application is deployed using Azure Container Instances (ACI). During a flash sale, the traffic surges significantly, causing delayed responses from the application. You must implement a solution to handle such traffic spikes without redesigning the application architecture. What should you configure to meet this requirement?

A. Use the Azure portal to add a new container instance to the existing container group and manually distribute traffic

B. Deploy an Azure Kubernetes Service (AKS) cluster with auto-scaling enabled and migrate the application from ACI to AKS wrong

C. Set up an Azure Load Balancer to manage traffic and enable a custom health probe for the ACI container group

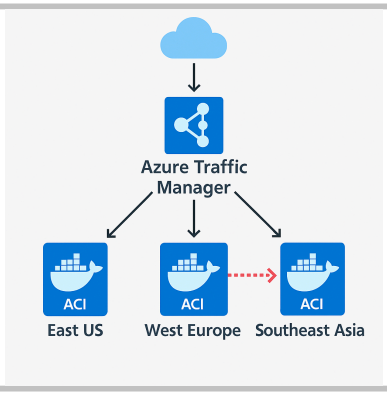
D. Configure Azure Traffic Manager to distribute incoming requests across multiple ACI deployments in different regions right

---

Explanation:

Correct Answer: D

Option D: Configure Azure Traffic Manager to distribute incoming requests across multiple ACI deployments in different regions is correct because Azure Traffic Manager is a DNS-based traffic routing service that can distribute traffic across multiple endpoints, including ACI deployments, in different regions. This approach effectively handles high traffic, provides fault tolerance by routing to healthy endpoints, and improves responsiveness without requiring a redesign of the core application architecture.



Option A: Use the Azure portal to add a new container instance to the existing container group and manually distribute traffic is incorrect because adding instances manually is not a scalable or efficient way to handle sudden traffic surges. It requires manual intervention and does not provide real-time responsiveness, which is essential during a flash sale.

Option B: Deploy an Azure Kubernetes Service (AKS) cluster with auto-scaling enabled and migrate the application from ACI to AKS is incorrect because while AKS is a robust solution for handling high traffic, it requires significant architectural changes and introduces operational complexity. The

question explicitly states that the solution must be implemented without redesigning the application architecture.

Option C: Set up an Azure Load Balancer to manage traffic and enable a custom health probe for the ACI container group is incorrect because Azure Load Balancer is not designed for direct integration with Azure Container Instances (ACI) and typically works within a virtual network. ACI operates outside a virtual network by default, making this solution incompatible with the basic ACI deployment model.

Reference:

https://learn.microsoft.com/en-us/azure/traffic-manager/traffic-manager-overview

Ask our Experts

Did you like this **Question?**

Question 3 Correct

Ask Whizzy AI

Domain: Deploy and manage Azure compute resources

An App Service is configured to support multiple custom domains. You are tasked with ensuring secure communication across all domains using Transport Layer Security (TLS) to secure all subdomains under “example.com”.

Proposed Solution: You upload a wildcard SSL certificate issued for “*.example.com”. You bind the wildcard certificate to the custom domains in the App Service. Is this proposed solution correct? (Select Yes or No)

A. Yes right

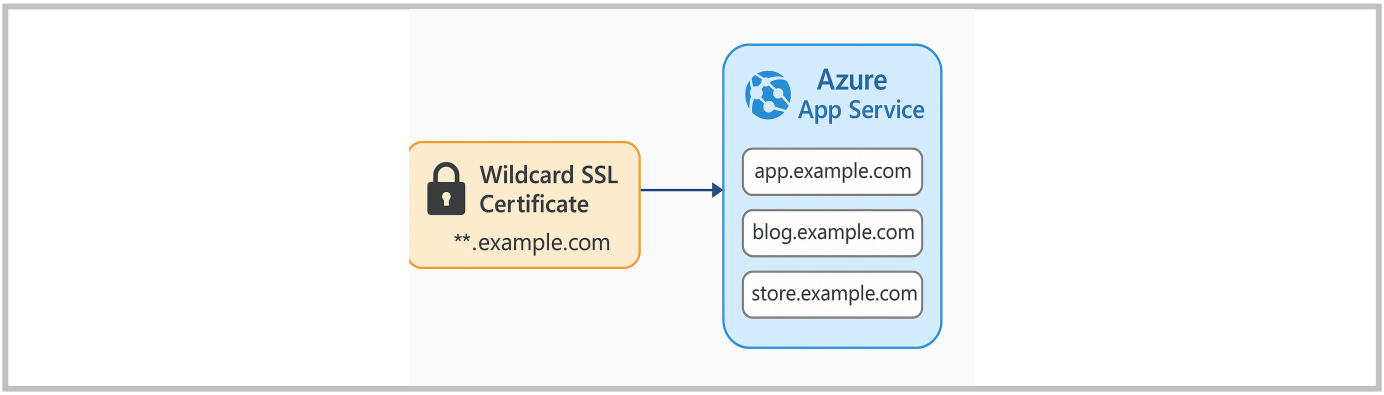
B. No

Explanation:

Correct Answer: A

The proposed solution is correct because a wildcard SSL certificate is specifically designed to secure a domain and all its subdomains. For instance, a certificate issued for “*.example.com” can secure “app.example.com”, “blog.example.com”, and other subdomains under “example.com”. By uploading

this wildcard certificate to Azure App Service and binding it to the custom domains, you enable secure communication over HTTPS for all these subdomains.



This approach simplifies management since a single certificate eliminates the need to upload and manage separate certificates for each subdomain. Additionally, the App Service platform supports binding wildcard SSL certificates to custom domains, ensuring Transport Layer Security (TLS) is enforced. By enabling HTTPS traffic with a wildcard certificate, you achieve end-to-end encryption, protecting data integrity and ensuring the authenticity of your website. It's important to note that wildcard certificates do not cover nested subdomains (e.g., sub.blog.example.com). However, this limitation does not affect the solution in the given scenario as it pertains to first-level subdomains. By following this setup, you adhere to best practices for securing applications hosted on Azure App Service.

References:

https://learn.microsoft.com/en-us/azure/cloud-services/cloud-services-custom-domain-name-portal#a-record

https://learn.microsoft.com/en-us/azure/app-service/configure-ssl-bindings#bind-a-certificate

Ask our Experts

Did you like this **Question?**

Question 4 Incorrect

Domain: Deploy and manage Azure compute resources

An App Service is deployed in a production environment and must connect securely to an Azure SQL Database using service endpoints. Public access to the database should be disabled without

affecting the application's connectivity. Arrange the following steps in the correct order to configure this setup.

Your Answer	Correct Answer
1. A. Enable the "Service Endpoint" feature on the subnet associated with the App Service	1. C. Integrate the App Service with the subnet using VNet Integration
2. B. Disable public access to the Azure SQL Database	2. A. Enable the "Service Endpoint" feature on the subnet associated with the App Service
3. C. Integrate the App Service with the subnet using VNet Integration	3. D. Update the Azure SQL Database firewall rules to allow the subnet
4. D. Update the Azure SQL Database firewall rules to allow the subnet	4. B. Disable public access to the Azure SQL Database
5. E. Test the App Service to confirm connectivity	5. E. Test the App Service to confirm connectivity

Explanation:

Correct Answers: C, A, D, B and E

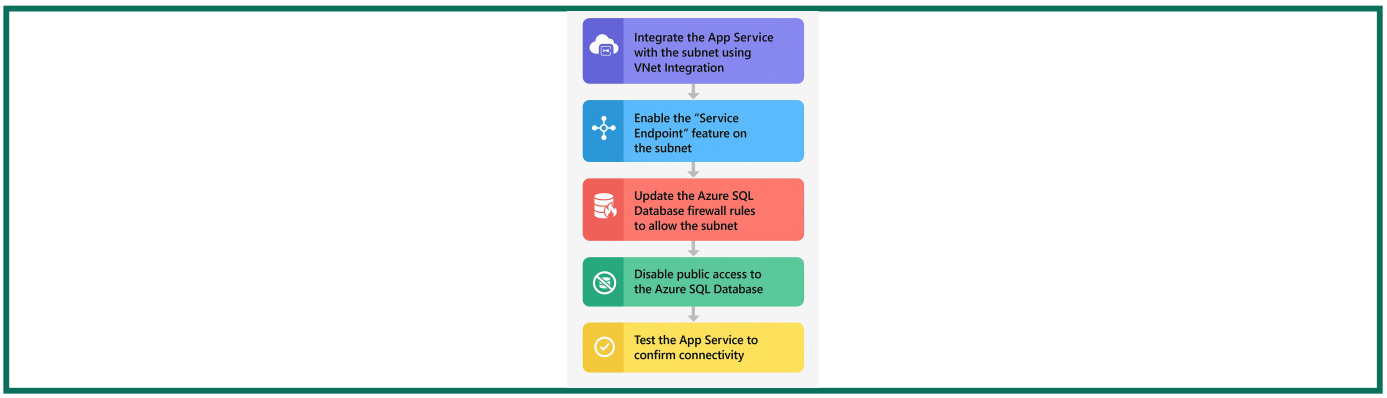
C - Integrate the App Service with the subnet using VNet Integration.

A - Enable the "Service Endpoint" feature on the subnet associated with the App Service.

D - Update the Azure SQL Database firewall rules to allow the subnet.

B - Disable public access to the Azure SQL Database.

E - Test the App Service to confirm connectivity.



1. **Integrate the App Service with the subnet using VNet Integration**: The first step is to enable Virtual Network (VNet) Integration for the App Service. This allows the App Service to communicate with resources in a virtual network, such as the Azure SQL Database. The VNet Integration ensures that traffic between the App Service and the database travels through a private network rather than the public internet, improving security. This setup requires selecting the appropriate subnet in the VNet where the App Service will be integrated. By establishing this connectivity first, you lay the groundwork for the App Service to access the database securely.
2. **Enable the "Service Endpoint" feature on the subnet associated with the App Service:** After integrating the App Service with the VNet, the next step is to enable service endpoints on the subnet. Service endpoints extend the identity of the VNet to the Azure SQL Database, allowing secure access. When a service endpoint is enabled, the subnet is explicitly permitted to access the Azure SQL Database, providing enhanced security and reducing reliance on public IP access. This step ensures that the App Service traffic to the database is recognized as originating from a trusted VNet.
3. **Update the Azure SQL Database firewall rules to allow the subnet:** Once the service endpoint is enabled, you must configure the Azure SQL Database to allow access from the subnet. This involves updating the database's firewall settings to include the subnet address range. This step ensures that only traffic from the specified subnet (and by extension, the App Service) is allowed to access the database, blocking any unauthorized access. This rule explicitly secures the database from external threats while maintaining necessary connectivity.
4. **Disable public access to the Azure SQL Database**: After securing connectivity via the service endpoint, you can disable public access to the Azure SQL Database. This step ensures that the database is no longer accessible from any public IPs, significantly reducing the attack surface. By doing this only after verifying private connectivity through the previous steps, you avoid any interruptions in the application's database connectivity. This step reinforces the security of the production environment.
5. **Test the App Service to confirm connectivity:** Finally, you must validate that the App Service can connect to the Azure SQL Database. This involves running end-to-end tests to confirm that the application functions correctly and securely communicates with the database. This step ensures that all configurations are implemented correctly and no disruptions occur. Testing helps identify any misconfigurations or connectivity issues before moving forward with the production workload.

References:

https://learn.microsoft.com/en-us/security/benchmark/azure/baselines/app-service-security-baseline

https://learn.microsoft.com/en-us/azure/app-service/overview-vnet-integration

https://learn.microsoft.com/en-us/azure/virtual-network/virtual-network-service-endpoints-overview

https://learn.microsoft.com/en-us/azure/azure-sql/database/firewall-configure?view=azuresql

Ask our Experts

Did you like this **Question?**

Question 5 Incorrect

Domain: Deploy and manage Azure compute resources

A company needs to configure deployment slots for its Azure App Service to achieve zero-downtime updates and environment isolation. Which three tasks are necessary to ensure seamless deployment, environment-specific settings, and proper traffic management? (Select Three)

Your Answer	Correct Answer
A. Enable auto-swap	A. Enable auto-swap
E. Swap staging to production	D. Configure slot settings
B. Clone production settings	E. Swap staging to production

Explanation:

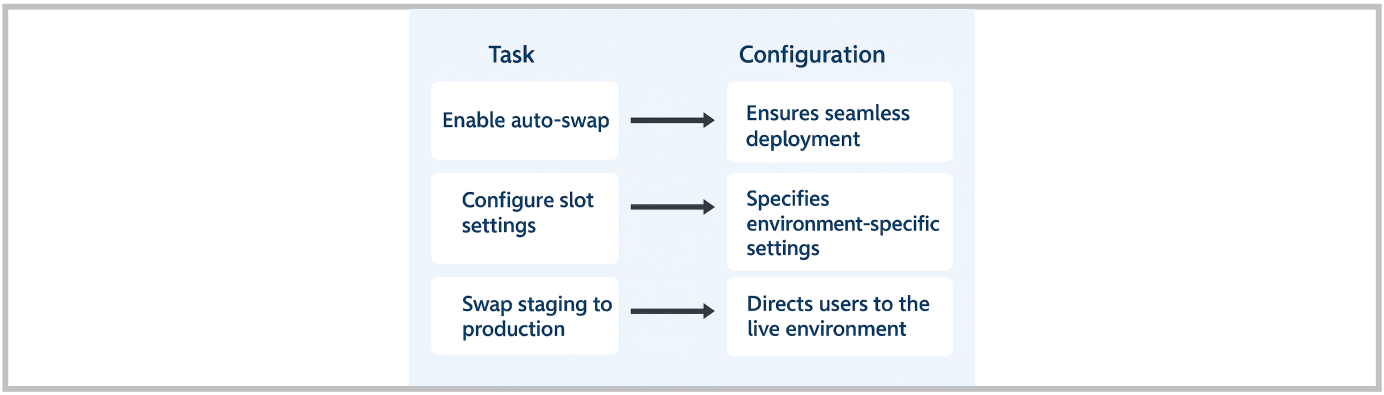
Correct Answers: A, D and E

Option A: Enable auto-swap is correct because auto-swap is a feature in Azure App Service deployment slots that automatically swaps a staging slot with the production slot once the deployment is complete and the application is verified to be ready. This ensures a seamless transition without manual intervention and significantly reduces downtime, making it ideal for zero-downtime updates. Auto-swap eliminates the need for manual slot swapping by triggering the swap after a successful deployment. The staging slot is warmed up before being swapped, ensuring performance and readiness. It supports CI/CD scenarios by making the deployment process faster and more reliable.

Option D: Configure slot settings is correct because deployment slots allow each slot to have its app settings and connection strings, which can be configured independently. This isolation ensures that

different environments, such as staging and production, do not interfere with each other. By configuring slot-specific settings, you maintain environment isolation, which is crucial for testing and compliance. Slot-specific settings allow you to define unique values for each deployment slot, such as database connection strings or API keys. Prevents production data or settings from being accidentally exposed during testing in staging environments. Supports scenarios where slots require different configurations for seamless testing and deployment.

Option E: Swap staging to production is correct because swapping a staging slot with a production slot is a core feature of Azure deployment slots. It allows you to deploy changes to the staging slot, test them thoroughly, and then promote the changes to production without downtime. The swap operation is transactional, meaning it can revert automatically if there are issues. The swap operation involves switching the content and configuration between the two slots. It minimizes risks by keeping the production slot untouched until the changes in staging are validated. The transactional nature of slot swaps ensures fail-safe deployments.



Option B: Clone production settings is incorrect because while cloning production settings might sound useful, it is not an essential task for enabling zero-downtime updates or environment isolation. Deployment slots already share the same base configurations, and additional slot-specific configurations can be applied independently. This option does not directly contribute to the objectives of the question.

Option C: Set traffic routing is incorrect because traffic routing is primarily used for A/B testing or gradual feature rollouts by directing a percentage of traffic to a specific slot. While valuable in its context, traffic routing is not a mandatory step for achieving zero-downtime updates or environment isolation as per the scenario described.

Reference:

https://learn.microsoft.com/en-us/azure/app-service/deploy-staging-slots?tabs=portal#autoswap

Ask our Experts

Did you like this **Question?**

Finish Review