CloudLamb E-Learning

Capacitación E-Learning

AWS ASSOCIATE
Certified Solutions ArchitectAssociate
SAA-C03

CloudLamb E-Learning

Actualmente el mundo de las telecomunicaciones ha crecido vertiginosamente con lo cual la Certificación de Amazon Web Services (AWS) se está convirtiendo cada vez más imprescindible para el profesional de TI

La plataforma de servicios de nube proporciona una variedad de servicios de Infraestructura tales como: almacenamiento, redes, bases de datos, servicios de aplicaciones, potencia de cómputo, inteligencia artificial, servicios móviles, seguridad entre otros beneficios.

Este curso ha sido diseñado para ayudar al participante y profesionales de TI a tener dominio en la preparación del examen **AWS SOLUTIONS ARCHITECT. El contenido se encuentra actualizado a la última versión.**

Duración: 12 semanas de clases en vivo, en las cuales aprenderás las bases de AWS y estará capacitad@ hasta obtener los conocimientos para la certificación Solutions Architect examen SAA C03

CloudLamb E-Learning

Puntos importantes que se deben dominar para el examen de certificación

Domain	% of Exam
Domain 1: Design Secure Architectures	30%
Domain 2: Design Resilient Architectures	26%
Domain 3: Design High-Performing Architectures	24%
Domain 4: Design Cost-Optimized Architectures	20%
TOTAL	**100%**


CloudLamb
E-Learning
CLASES Online
EN TIEMPO REAL
Cloud.Elearn
Cloud_Elearn
Cloud Lamb
Cloud-Lamb.com

Domain 1: Design Secure Architectures

Task Statement 1: Design secure access to AWS resources.

- Access controls and management across multiple accounts
- AWS federated access and identity services (for example, AWS Identity and Access Management [IAM], AWS Single Sign-On [AWS SSO])
- AWS global infrastructure (for example, Availability Zones, AWS Regions)
- AWS security best practices (for example, the principle of least privilege)
- The AWS shared responsibility model

Task Statement 2: Design secure workloads and applications.

- Application configuration and credentials security
- AWS service endpoints
- Control ports, protocols, and network traffic on AWS
- Secure application access
- Security services with appropriate use cases (for example, Amazon Cognito, Amazon GuardDuty, Amazon Macie)
- Threat vectors external to AWS (for example, DDoS, SQL injection)

Task Statement 3: Determine appropriate data security controls

- Aligning AWS technologies to meet compliance requirements
- Encrypting data at rest (for example, AWS Key Management Service [AWS KMS])
- Encrypting data in transit (for example, AWS Certificate Manager [ACM] using TLS)
- Implementing access policies for encryption keys
- Implementing data backups and replications
- Implementing policies for data access, lifecycle, and protection
- Rotating encryption keys and renewing certificates

Domain 2: Design Resilient Architectures

Task Statement 1: Design scalable and loosely coupled architectures

- API creation and management (for example, Amazon API Gateway, REST API)
- AWS managed services with appropriate use cases (for example, AWS Transfer Family, Amazon Simple Queue Service [Amazon SQS], Secrets Manager)
- Caching strategies
- Design principles for microservices (for example, stateless workloads compared with stateful workloads)
- Event-driven architectures
- Horizontal scaling and vertical scaling
- How to appropriately use edge accelerators (for example, content delivery network [CDN])
- How to migrate applications into containers
- Load balancing concepts (for example, Application Load Balancer)
- Multi-tier architecture
- Queuing and messaging concepts (for example, publish/subscribe)
- Serverless technologies and patterns (for example, AWS Fargate, AWS Lambda)
- Storage types with associated characteristics (for example, object, file, block)
- The orchestration of containers (for example, Amazon Elastic Container Service [Amazon ECS], Amazon Elastic Kubernetes Service [Amazon EKS])
- When to use read replicas
- Workflow orchestration (for example, AWS Step Functions)

Task Statement 2: Design highly available and/or fault-tolerant architectures

- AWS global infrastructure (for example, Availability Zones, AWS Regions, Amazon Route 53)
- AWS managed services with appropriate use cases (for example, Amazon Comprehend, Amazon Polly)
- Basic networking concepts (for example, route tables)
- Disaster recovery (DR) strategies (for example, backup and restore, pilot light, warm standby, active-active failover, recovery point objective [RPO], recovery time objective [RTO])
- Distributed design patterns
- Failover strategies
- Immutable infrastructure
- Load balancing concepts (for example, Application Load Balancer)
- Proxy concepts (for example, Amazon RDS Proxy)
- Service quotas and throttling (for example, how to configure the service quotas for a workload in a standby environment)
- Storage options and characteristics (for example, durability, replication)
- Workload visibility (for example, AWS X-Ray)

Task Statement 3: Determine high-performing database solutions

- AWS global infrastructure (for example, Availability Zones, AWS Regions)
- Caching strategies and services (for example, Amazon ElastiCache)
- Data access patterns (for example, read-intensive compared with write-intensive)
- Database capacity planning (for example, capacity units, instance types, Provisioned IOPS)
- Database connections and proxies
- Database engines with appropriate use cases (for example, heterogeneous migrations, homogeneous migrations)
- Database replication (for example, read replicas)
- Database types and services (for example, serverless, relational compared with non-relational, in-memory)

Task Statement 4: Determine high-performing and/or scalable network architectures.

- Edge networking services with appropriate use cases (for example, Amazon CloudFront, AWS Global Accelerator)
- How to design network architecture (for example, subnet tiers, routing, IP addressing)
- Load balancing concepts (for example, Application Load Balancer)
- Network connection options (for example, AWS VPN, Direct Connect, AWS PrivateLink)

Task Statement 5: Determine high-performing data ingestion and transformation solutions.

- Data analytics and visualization services with appropriate use cases (for example, Amazon Athena, AWS Lake Formation, Amazon QuickSight)
- Data ingestion patterns (for example, frequency)
- Data transfer services with appropriate use cases (for example, AWS DataSync, AWS Storage Gateway)
- Data transformation services with appropriate use cases (for example, AWS Glue)
- Secure access to ingestion access points
- Sizes and speeds needed to meet business requirements
- Streaming data services with appropriate use cases (for example, Amazon Kinesis)

Domain 4: Design Cost-Optimized Architectures

- Access options (for example, an S3 bucket with Requester Pays object storage)
- AWS cost management service features (for example, cost allocation tags, multi-account billing)
- AWS cost management tools with appropriate use cases (for example, AWS Cost Explorer, AWS Budgets, AWS Cost and Usage Report)
- AWS storage services with appropriate use cases (for example, Amazon FSx, Amazon EFS, Amazon S3, Amazon EBS)
- Backup strategies
- Block storage options (for example, hard disk drive [HDD] volume types, solid state drive [SSD] volume types)
- Data lifecycles
- Hybrid storage options (for example, DataSync, Transfer Family, Storage Gateway)
- Storage access patterns
- Storage tiering (for example, cold tiering for object storage)
- Storage types with associated characteristics (for example, object, file, block)

CloudLamb
E-Learning