Chapter 1

Key Important Points:

1. Core Definition

- Cloud computing is on-demand delivery of IT resources via internet with pay-as-you-go pricing

- Fundamental shift from "infrastructure as hardware" to "infrastructure as software"

2. Cloud Service Models (in order of control level)

- IaaS (Infrastructure as a Service) - Most control

- PaaS (Platform as a Service) - Medium control

- SaaS (Software as a Service) - Least control

3. Cloud Deployment Models with Key Characteristics

- Public Cloud: Easiest to use, most cost-effective, lowest security

- Private Cloud: Highest security and control, most expensive

- Hybrid Cloud: Balance of private/public benefits

- Community Cloud: Shared costs and security

4. Major Advantages of Cloud Computing

- Pay-as-you-go pricing instead of capital investment

- Economies of scale benefits

- Dynamic capacity scaling

- Rapid resource deployment (minutes vs weeks)

- Elimination of data center maintenance costs

- Global deployment capability

5. Traditional IT vs Cloud Computing Equivalents

- Security: Traditional firewalls → Security groups/ACLs

- Compute: On-premises servers → EC2 instances

- Storage: DAS/SAN/NAS → EBS/EFS/S3

- Database: Traditional RDBMS → Amazon RDS

Chapter2

1. Historical Evolution

- Cloud computing evolved from earlier technologies: mainframes, parallel computing, supercomputers, distributed computing, grid computing

- Key distinction: Cloud computing introduced the utility computing business model (pay-as-you-use)

2. Virtualization Fundamentals

Critical Types:

- Server Virtualization: Uses hypervisors to create virtual machines on physical servers

- Network Virtualization: Abstracts network resources to software (e.g., VLANs)

- Storage Virtualization: Consolidates physical storage into virtual storage clusters

- Application Virtualization: Delivers apps from server to end-user devices (e.g., Office 365)

- Desktop Virtualization: Separates desktop environment from physical device (VDI)

3. Hypervisor Types

- Type 1 ("Bare-metal"): Runs directly on hardware

- Type 2 ("Hosted"): Runs on top of an operating system

4. AWS Global Infrastructure

Key Components:

- Regions: Geographical areas with full redundancy

- Availability Zones: Isolated data centers within regions

- Edge Locations: Points of Presence for content delivery

- Data Centers: Physical facilities (50,000-80,000 servers each)

5. Region Selection Criteria

- Data governance and legal requirements

- Proximity to customers (latency)

- Available services

- Cost variations

I'll extract the key important points from Chapter 3, focusing on the core concepts while filtering out less essential information:

Key Important Points:

1. Systems Development Lifecycle (SDLC)

- Software delivery requires balancing stability and rapid development
- Core Development Phases:
  - Plan -> Define -> Design -> Develop -> Deploy -> Maintain
  - Two methodologies: Waterfall (sequential) vs Agile (iterative)

2. AWS Development Setup Process

- Four Essential Steps:
  1. Create AWS Account
  2. Set up AWS IAM permissions
  3. Install development environment
  4. Interact with AWS services (via Console/CLI/SDK)

3. AWS Cloud9 Development Environment

- Key Features:
  - Browser-based coding environment
  - Collaborative coding capabilities
  - Integrated debugger and terminal
  - Supports multiple programming languages
  - Direct AWS service integration

4. AWS Service Categories and Core Services

Foundation Services:

- Compute: EC2, Lambda, ECS

- Storage: S3, EBS, EFS

- Database: RDS, DynamoDB, Aurora

- Networking: VPC, Route 53, CloudFront

Security Services:

- IAM (Identity and Access Management)

- AWS Organizations

- AWS Shield

- AWS Key Management Service

I'll extract the key important points from Chapter 4, focusing on the core concepts of security and access management in AWS:

Key Important Points:

1. Shared Responsibility Model

- Cloud Service Provider (AWS) responsible for "security OF the cloud"
  - Hardware, software, networking, facilities
  - Global infrastructure security
- Customer responsible for "security IN the cloud"
  - Access management
  - Data security
  - Operating system configuration
  - Network and firewall settings

2. Identity and Access Management (IAM) Core Components

- Users: Entities that need AWS access (people, applications, services)
- Groups: Collections of users with shared permissions
- Roles: Temporary identities for services or cross-account access
- Policies: JSON documents that define permissions
  - Identity-based policies: Attached to users, groups, roles
  - Resource-based policies: Attached to AWS resources

3. IAM Security Principles

- Authentication: Verifies who is requesting access

- Authorization: Determines what actions are allowed

- Principle of Least Privilege: Grant minimum necessary permissions

- Default Deny: All permissions denied unless explicitly allowed

- Explicit Deny overrides any Allow permissions

4. IAM Best Practices

- Never use root account for daily tasks

- Use MFA for privileged accounts

- Rotate credentials regularly

- Use IAM roles for applications instead of stored credentials

- Organize users into groups for easier permission management

- Use temporary credentials when possible

1. AWS Elastic Beanstalk:

- A managed service that automatically handles infrastructure provisioning, deployment, load balancing, auto-scaling, health monitoring

- No additional charge (pay only for underlying resources)

- Supports multiple platforms: Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker

- Benefits: Fast to start, developer productivity, difficult to outgrow, complete resource control

2. Key Components of Amazon EC2:

- AMI (Amazon Machine Image):

  - Template used to create EC2 instances

  - Contains Windows/Linux OS and pre-installed software

  - Types: Quick Start, My AMIs, AWS Marketplace, Community AMIs

- Instance Types:

  - General purpose

  - Compute optimized

  - Memory optimized

  - Storage optimized

  - Accelerated computing

3. Security & Network Features:

- Security Groups: Set of firewall rules controlling traffic to instances

- Network Settings: VPC and subnet configuration

- Elastic IP addresses: Persistent public IP addresses

4. Storage Options:

- Amazon EBS (Elastic Block Store): Persistent storage

- EC2 Instance Store: Temporary storage

- Additional options: Amazon EFS, Amazon S3

5. Instance Lifecycle:

- States: pending, running, stopping, stopped, shutting-down, terminated

- Understanding difference between Stop/Start vs Terminate

6. Monitoring:

- Amazon CloudWatch integration

- Basic monitoring (5-minute intervals, free)

- Detailed monitoring (1-minute intervals, additional cost)

7. EC2 Instance Metadata:

- Data about your instance

- Accessible via http://169.254.169.254/latest/meta-data/

- Used for instance configuration and management

These concepts appear to be emphasized throughout the slides and represent core knowledge about AWS Elastic Beanstalk and EC2 services.

Chapter 7

Key Database Concepts:

1. Types of Cloud Databases:

- Relational (SQL) Databases

- Non-relational (NoSQL) Databases

2. Cloud Database Deployment Options:

- Managed database-as-a-service (DBaaS)

- Self-managed on cloud VM by in-house IT team

3. Relational vs NoSQL Databases Comparison:

- Relational: Fixed schema, rows/columns, SQL queries, vertical scaling

- NoSQL: Dynamic schema, flexible data models, document-based queries, horizontal scaling

4. Types of NoSQL Databases:

- Key-value pair

- Document-oriented

- Column-oriented

- Graph-based

5. DynamoDB Specific Concepts:

- Components: Tables, Items, Attributes

- Primary Keys: Simple (partition key) vs Composite (partition + sort key)

- Secondary Indexes: Global vs Local

- Read Consistency: Eventually Consistent vs Strongly Consistent

- Read/Write Capacity Units (RCU/WCU)

6. Important Calculations:

- RCU calculation (for 4KB reads)

- WCU calculation (for 1KB writes)

7. Use Cases:

- When to use Relational Databases:

  * Complex transactions

  * Complex queries

  * High durability requirements

- When to use NoSQL:

  * Flexible schemas needed

  * High scalability required

  * Semi-structured/unstructured data

  * High performance priority over consistency

The review questions shown in the slides about AWS architectures also suggest that understanding real-world implementation scenarios could be important for the exam.

Chapter 8, 9

I'll analyze these slides and extract the key exam-worthy concepts, focusing on core definitions, features, and comparisons between different storage solutions. Here are the most important points:

1. Cloud Storage Fundamentals:

- Cloud storage is a method for storing data in the cloud that provides servers and applications access through shared file systems

- Three main types: Object Storage, Block Storage, and File Storage

- Key benefits: Scalability, accessibility, cost-effectiveness, durability

2. Amazon S3 (Object Storage):

- Used for storing unstructured data like images, videos, files

- Features: unlimited storage, 11 9s durability, access from anywhere via URL

- Storage classes: Standard, Intelligent-Tiering, Standard-IA, One Zone-IA, Glacier

- Use cases: Backup/archiving, media hosting, content delivery, application hosting

3. Amazon EBS (Block Storage):

- Provides persistent block-level storage for EC2 instances

- Volume Types:

  * SSD: General Purpose (up to 16,000 IOPS) and Provisioned IOPS (up to 64,000 IOPS)

  * HDD: Throughput Optimized and Cold Storage

- Features: Snapshots, encryption, elastic volumes

4. Key Comparisons:

- Block vs Object Storage:

  * Block: Better for frequent small changes (like databases)

  * Object: Better for large, unchanging files (like media)

5. Amazon S3 Glacier:

- Used for long-term archival storage

- Lowest cost storage option

- Retrieval options: Expedited (1-5 mins), Standard (3-5 hrs), Bulk (5-12 hrs)

- Features 11 9s of durability

- Use cases: Media archiving, healthcare records, compliance data

6. Security Considerations:

- Encryption at rest and in transit

- Access control through IAM

- Vault Lock feature for compliance

- Data residency considerations

These points are particularly important as they cover fundamental concepts, key differences between services, and practical applications - all common themes in exams.

Chapter 10, 11

Based on my analysis of these slides, here are the key concepts that are likely to be tested in an exam:

1. Caching Fundamentals:

- Definition and purpose of caching in cloud computing

- When to use caching (read-heavy workloads, compute-intensive workloads)

- Key considerations for implementing caching (data staleness, query expense, access frequency)

2. Amazon ElastiCache:

- Core features: in-memory data store, sub-millisecond latency

- Two engine types: Memcached vs Redis

- Key differences between Redis and Memcached features and use cases

- ElastiCache terminology (nodes, clusters, shards)

3. Amazon CloudFront:

- Purpose and benefits of CDN (Content Delivery Network)

- Types of CloudFront distributions:

  * Web content

  * Video on demand

  * Live event streaming

- Cache key concepts and control

- CloudFront functions vs Lambda@Edge capabilities and differences

- Use cases for CloudFront (global content delivery, reduced latency)

4. Caching Strategies:

- Lazy Loading:

  * Updates cache only when necessary

  * Best for read-heavy, infrequently updated data

  * Advantages: less cache data, non-fatal node failures

  * Disadvantages: cache miss penalty, potential stale data

- Write-Through:

  * Updates cache whenever database is updated

  * Best for real-time data requirements

  * Advantage: no stale data

  * Disadvantages: write penalty, missing data on node failure, cache churn

5. Best Practices:

- Cache invalidation

- Monitoring and alerts

- Security considerations

- Cache warm-up strategies

Chapter 12

Let me extract the key exam-relevant information from these slides about Serverless Computing and AWS Lambda:

Key Concepts:

1. Serverless Computing:

- Cloud computing model where provider manages infrastructure

- Developers only write and deploy code

- Pay per actual resource usage

- Automatic scaling and maintenance

2. AWS Lambda:

- Serverless compute service

- Invokes code in response to events

- Auto-scales

- Built-in monitoring with CloudWatch

- Only pay for compute time used

3. Lambda Invocation Models:

- Synchronous (Request/Response)

- Asynchronous (Event)

- Polling (Event source mapping)

4. Lambda Permissions:

- IAM resource policies for invocation
- Execution roles for Lambda functions
- Event sources need permissions to invoke Lambda
- Lambda needs permissions to interact with other services

5. API Gateway:
- Types: REST API and HTTP API
- Supports custom domains
- Can integrate with:
  - Lambda functions
  - HTTP endpoints
  - Other AWS services
  - Mock integrations

6. API Gateway Features:
- Authentication methods:
  - IAM
  - Lambda authorizers
  - Amazon Cognito
  - JWT
- Caching capabilities
- Request/Response transformations
- CORS support
- Rate limiting (throttling)

7. API Deployment:

- Uses stages (dev, test, prod)

- Supports canary deployments

- Custom domain names

- SSL/TLS certificates through ACM

Chapter 13

Based on the slides, here are the key concepts that are likely to be important for exam purposes:

1. Message Queue Services Overview:

- Types of asynchronous processing services:
  - Message queues (Amazon SQS)
  - Pub/sub messaging (Amazon SNS)
  - Data streams (Amazon Kinesis)

2. Amazon SQS (Simple Queue Service):

- Two types of queues:
  - Standard queues: Nearly unlimited throughput, at-least-once delivery, best-effort ordering
  - FIFO queues: Limited throughput, exactly-once delivery, preserves message order
- Main operations:
  - SendMessage
  - ReceiveMessage
  - DeleteMessage
- Features:
  - Visibility timeout
  - Dead-letter queues
  - Long polling vs Short polling
  - Security features (IAM, encryption, VPC endpoints)

3. Amazon SNS (Simple Notification Service):

- Features:
  - Pub/sub functionality
  - Message filtering
  - Multiple endpoint types (A2A and A2P)
  - Supports both standard and FIFO topics
- Endpoint Types:
  - Application-to-Application (A2A): Lambda, SQS, HTTP(S), Kinesis
  - Application-to-Person (A2P): Email, SMS, mobile push
- Security features similar to SQS

4. Use Cases:

- Microservices communication
- Event-driven architectures
- Asynchronous processing
- Data streaming
- Application decoupling

5. Important Concepts:

- Message lifecycle
- Visibility timeout
- Dead-letter queues
- Message filtering
- Topic subscriptions
- Security and access control