

MLOps

Introduction to Cloud Computing

Course Plan



Storage



Compute



Networking



Databases



Cloud
Operations



Security



Serverless

<https://www.awseducate.com/>

Computer Architecture:

① Parallel / HPC

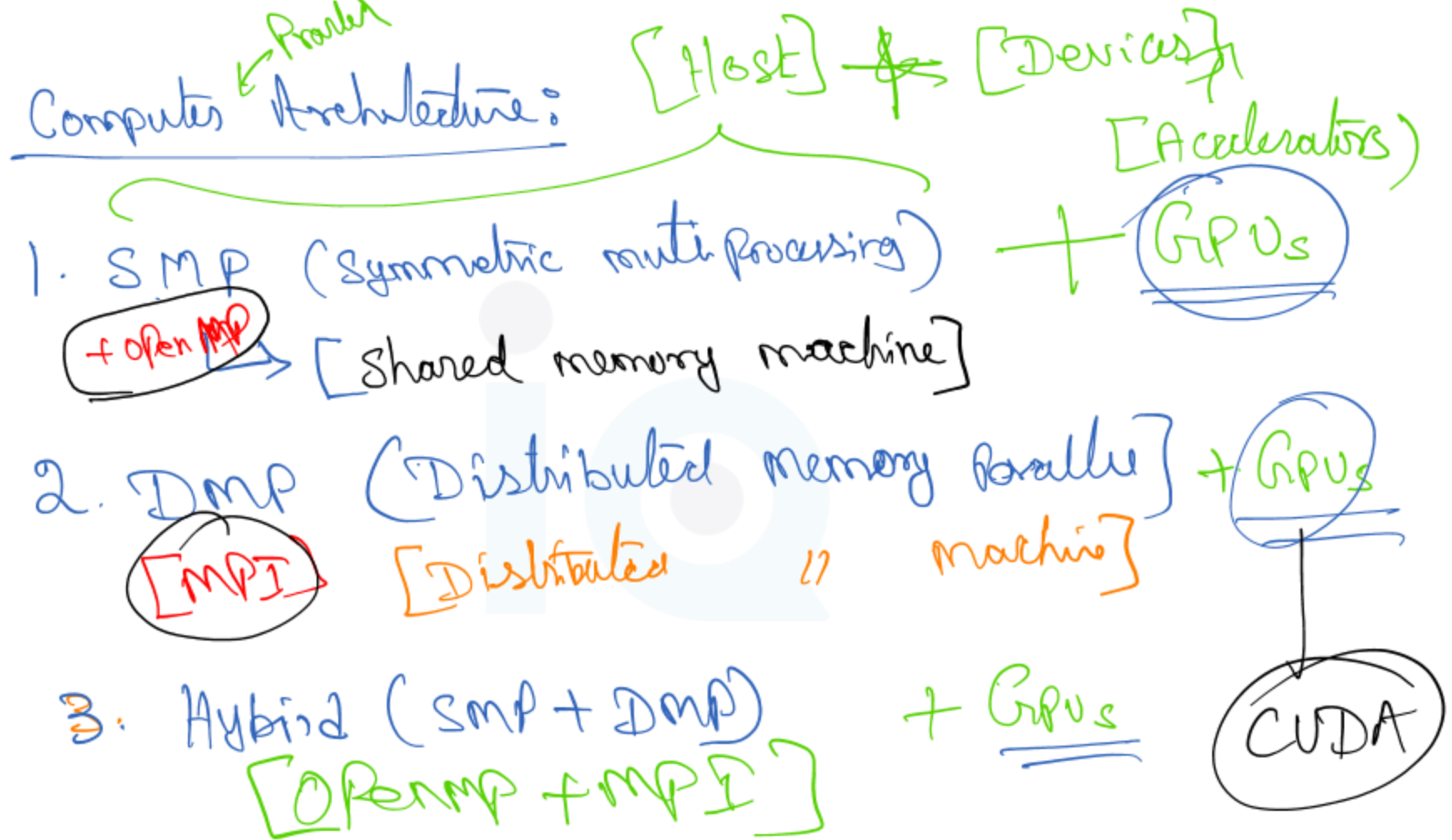
② GPU Compute / CI

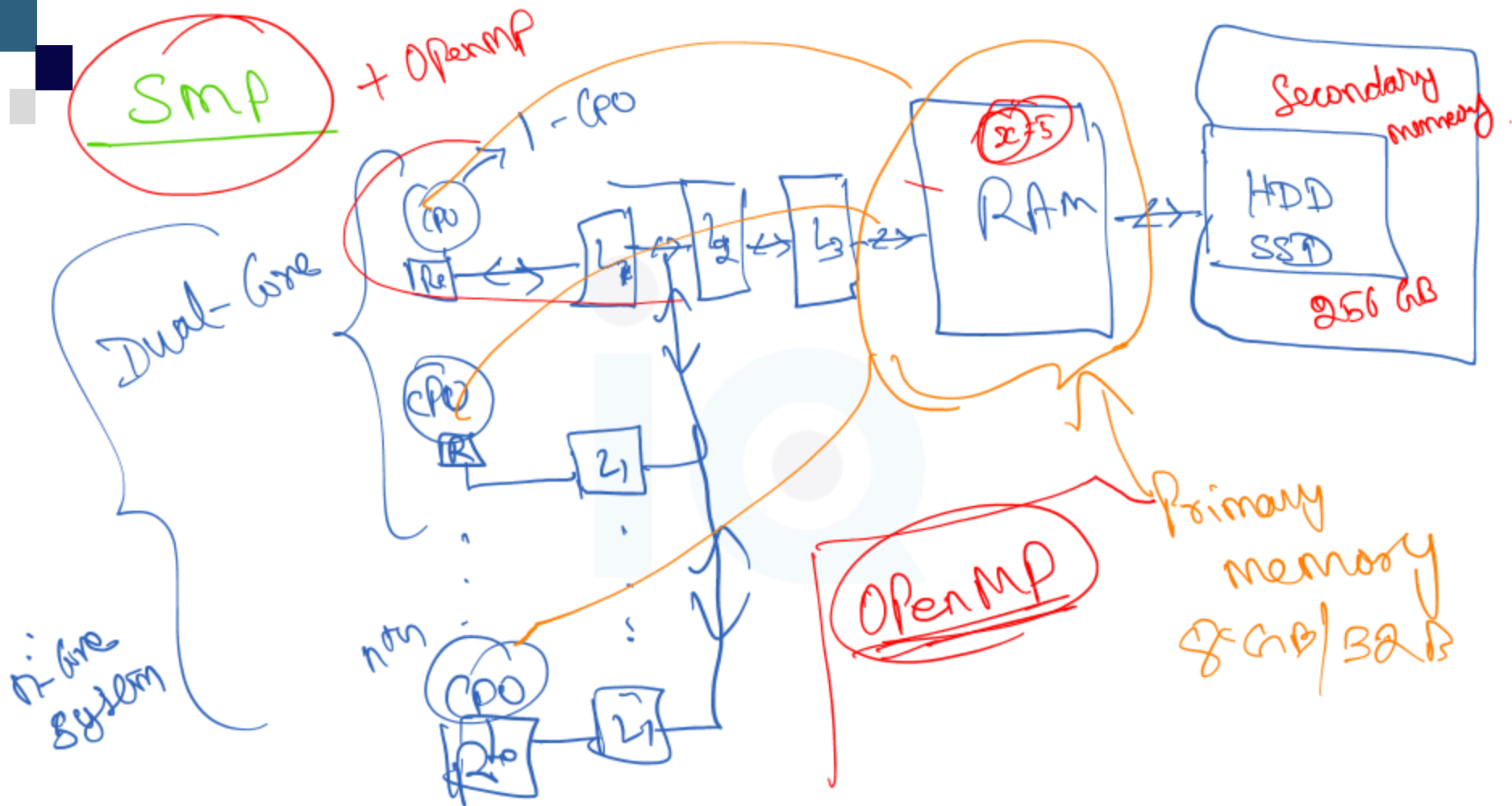
③ CPU - Compute

④ Edge -

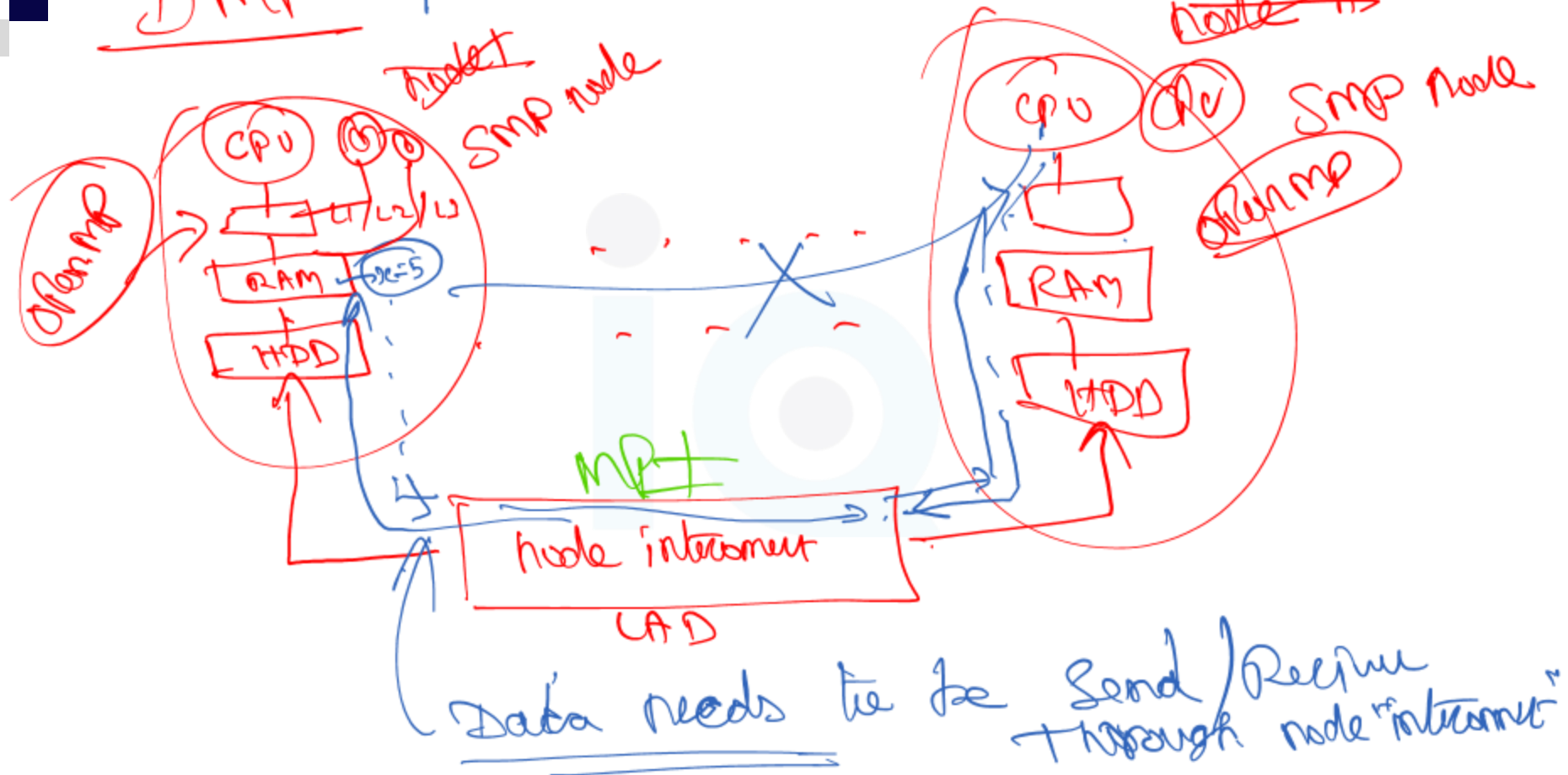
⑤ Distributed Computers

⇒ Can we choose
the right compute
infra on cloud?





DMA + MPI (message passing interface) ← standard



“Getting Started with Cloud Compute

CLOUD COMPUTE



Overview

In today's digital era, businesses are increasingly relying on cloud computing to power their applications and services.

AWS Compute services play a vital role in enabling organizations to leverage the power of the cloud for their computing needs.

In this presentation, we will explore the key concepts, benefits, and services offered by AWS Compute.

By the end of this session, you will have a solid understanding of AWS Compute and be well-equipped to start utilizing these services in your own projects and applications.

“Cloud Compute Services

Amazon EC2

Secure and resizable compute capacity for virtually any workload

750 hours per month

for 12 months with the [AWS Free Tier](#)

Get Started with Amazon EC2

Connect with an Amazon EC2 specialist

Access reliable, scalable infrastructure on demand. Scale capacity within minutes with SLA commitment of 99.99% availability.

Provide secure compute for your applications. Security is built into the foundation of Amazon EC2 with the AWS Nitro System.

Optimize performance and cost with flexible options like AWS Graviton-based instances, Amazon EC2 Spot instances, and AWS

Migrate and build apps with ease using AWS Migration Tools, AWS Managed Services, or Amazon Lightsail. Learn more about AWS.

- A web service that provides resizable compute capacity in the cloud.
- It allows you to quickly scale compute resources up or down based on your application requirements.
- EC2 instances can be used for various workloads such as web servers, databases, and big data processing.
- With EC2, you have full control over your virtual machines, including the choice of operating system, networking, and security settings.

Amazon Elastic Container Service

Run highly secure, reliable, and scalable containers

Create an AWS Account

Sign up for an Amazon ECS workshop

Launch containers on AWS at scale without worrying about the underlying infrastructure.

Reduce costs with automatic scaling and pay-as-you-go pricing across multiple AWS compute

Deploy faster and focus on your applications by using Amazon ECS with AWS Fargate serverless compute for containers

Build on Amazon ECS with confidence, knowing that the security, compliance, and architecture meet regulatory standards.

- Amazon ECS is a fully managed container orchestration service.
- It simplifies the deployment and management of containers at scale.
- You can run Docker containers on a cluster of EC2 instances without worrying about the underlying infrastructure.
- ECS provides features like service scheduling, load balancing, and auto scaling for containers. It integrates well with other AWS services such as Elastic Load Balancer, IAM, and CloudWatch.

Amazon Elastic Kubernetes Service

The most trusted way to start, run, and scale Kubernetes

Create an AWS Account

Leverage built-in integrations with AWS services such as EC2, VPC, IAM, EBS and more

Reduce costs with efficient compute resource provisioning and automatic Kubernetes application scaling.

Ensure a more secure Kubernetes environment with security patches automatically applied to your cluster control plane.

- Amazon EKS is a managed Kubernetes service that simplifies the deployment and operation of Kubernetes clusters.
- It eliminates the need to install and manage the Kubernetes control plane infrastructure.
- With EKS, you can easily run and scale containerized applications using Kubernetes orchestration capabilities.
- EKS integrates with other AWS services such as Elastic Load Balancer, IAM, and CloudTrail for enhanced security and observability.
- It provides automatic upgrades and high availability, ensuring your applications are always running smoothly.

AWS Lambda

Run code without thinking about servers or clusters

Create an AWS Account

Connect with an AWS Lambda specialist

1 million requests free
per month with the AWS Free Tier

Run code without provisioning or managing infrastructure. Simply write and upload code as a .zip file or container image.

Automatically respond to code execution requests at any scale, from a dozen events per day to hundreds

Save costs by paying only for the compute time you use—by per-millisecond—instead of provisioning infrastructure upfront for

Optimize code execution time and performance with the right function memory size. Respond to high demand in double-digit

AWS Lambda is a serverless computing service that lets you run code without provisioning or managing servers.

- It enables you to focus on writing code and automating tasks without worrying about infrastructure.
- Lambda functions are triggered by events and can be written in various programming languages.
- You only pay for the compute time consumed by your functions, making it highly cost-effective.
- Lambda integrates seamlessly with other AWS services, allowing you to build event-driven

“EC2 (Elastic Compute Cloud)



Amazon EC2 (Elastic Compute Cloud)



- What is Amazon EC2?
 - Amazon EC2 is a web service that provides scalable compute capacity in the cloud.
 - It allows you to quickly provision virtual servers, known as instances, and easily scale them up or down based on your needs.
 - EC2 offers a wide selection of instance types to accommodate various workloads and applications.
- EC2 instance types:
 - General-purpose instances: Balanced CPU, memory, and networking resources. Suitable for a wide range of applications.
 - Memory-optimized instances: High memory capacity for memory-intensive workloads such as databases and in-memory caching.



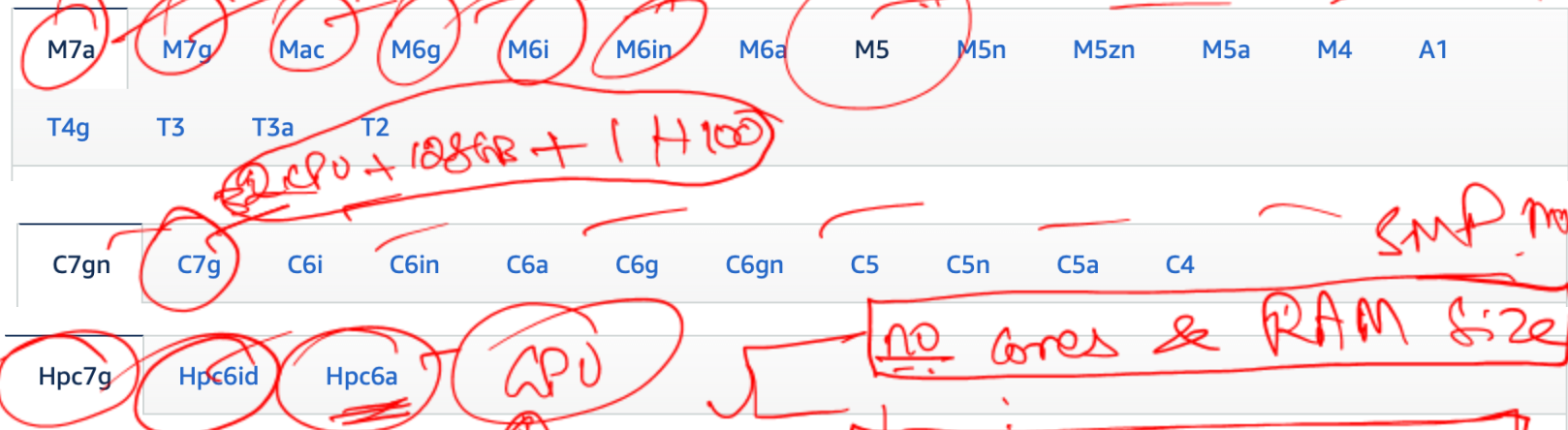
Amazon EC2 (Elastic Compute Cloud)



- GPU instances: Equipped with powerful GPUs for accelerated computing tasks like machine learning and graphics processing.
- Storage-optimized instances: Designed for workloads that require high disk throughput and low latency storage access.
- Steps to launch an EC2 instance:
 - Select the desired instance type and configuration parameters.
 - Choose the Amazon Machine Image (AMI), which contains the operating system and software for the instance.
 - Configure networking options, including virtual private clouds (VPCs) and security groups. Specify storage options such as Amazon Elastic Block Store (EBS) volumes or instance store volumes.

Amazon EC2 (Elastic Compute Cloud)

- Define user data scripts or metadata for customizing the instance.
- Review and launch the instance. Elastic IP addresses and security groups: Elastic IP addresses allow you to associate a static IP address with your instance, enabling you to maintain a consistent IP even if the instance is stopped and started. Security groups act as virtual firewalls, controlling inbound and outbound traffic to your EC2 instances. They define access rules based on protocols, ports, and IP ranges.

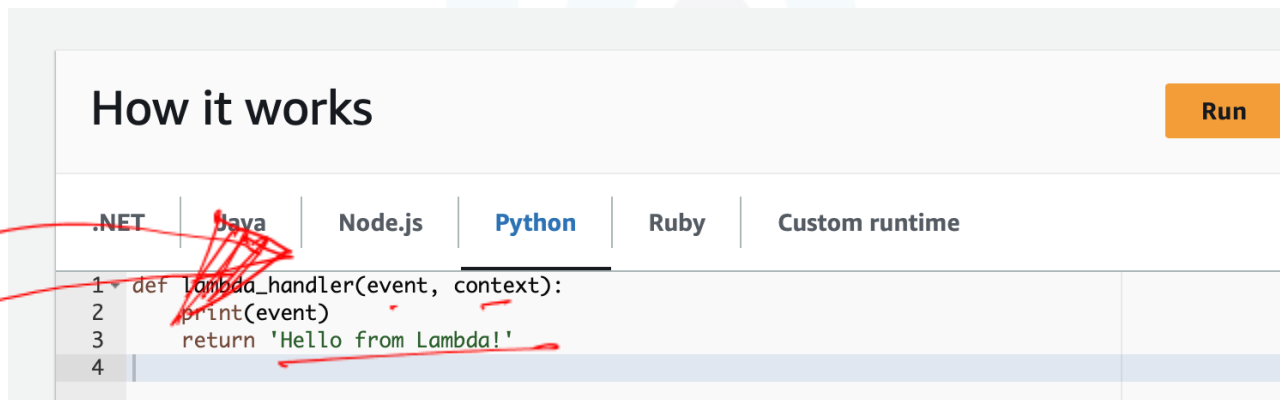


“AWS Lambda

... run code without thinking about servers.

AWS Lambda

- What is serverless computing?
 - Serverless computing is a cloud computing model where the cloud provider manages the infrastructure and automatically provisions resources to execute code without the need for server management.
 - With serverless computing, developers can focus solely on writing and deploying their code, while the cloud provider takes care of scaling, availability, and maintenance.

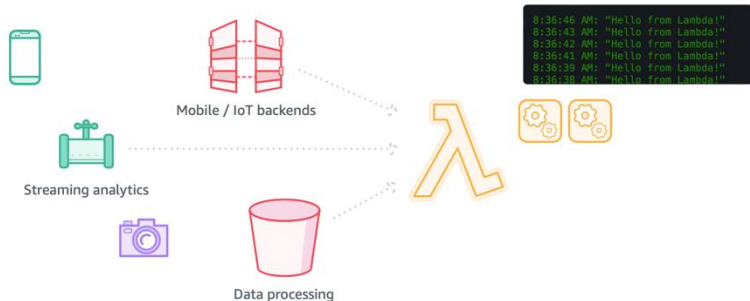


AWS Lambda

How it works

Previous

Next: Scale seamlessly



- Advantages of AWS Lambda:

- **Event-driven architecture:** Lambda functions can be triggered by various events, such as changes to data in an Amazon S3 bucket, updates to a DynamoDB table, or HTTP requests via Amazon API Gateway.
- This enables developers to build highly scalable and responsive applications.
- **Pay-per-use billing:** With Lambda, you are billed based on the number of requests and the duration of the function's execution.

AWS Lambda

- Only pay for the actual compute time consumed, with no charges when the function is idle. This provides cost efficiency and eliminates the need for upfront provisioning or over-provisioning of resources.
- **Automatic scaling:**
 - AWS Lambda automatically scales your functions in response to incoming requests.
 - It provisions and manages the necessary compute resources to handle concurrent invocations, ensuring that your applications can handle high traffic loads without manual intervention.
- **Easy integration with other AWS services:**
 - Lambda seamlessly integrates with other AWS services, allowing you to create powerful and modular applications.
 - You can easily combine Lambda with services like Amazon S3, DynamoDB, SNS, and more, to build event-driven architectures and implement complex workflows.



AWS Lambda

- Creating and deploying a Lambda function:
 - Using the AWS Management Console, AWS CLI, or AWS SDKs, you can create a Lambda function by specifying the runtime environment, code, and any required dependencies.
 - You can write Lambda functions in languages such as Python, Node.js, Java, C#, and Go.
 - Once the function is created, you can define triggers to specify the events that will invoke the function. This could be an upload to an S3 bucket, a change in a DynamoDB table, or an API Gateway request.
 - After deploying the Lambda function, AWS takes care of managing the underlying infrastructure, scaling it up or down based on demand.
 - You can monitor the function's performance and configure logging and error handling to ensure smooth operation.



AWS Lambda

- Integration with other AWS services:
 - Lambda can be seamlessly integrated with other AWS services.
 - For example, you can configure a Lambda function to process data uploaded to an S3 bucket, analyze it using machine learning services like Amazon Rekognition or Amazon Comprehend, and store the results in DynamoDB or trigger notifications through Amazon SNS.
 - You can also use Lambda in conjunction with AWS Step Functions to orchestrate complex workflows, where each step of the workflow is executed by a separate Lambda function.
 - Furthermore, AWS Lambda functions can be used as custom code for serverless application architectures built using services like AWS AppSync, AWS Amplify, or AWS SAM (Serverless Application Model).

AWS Lambda



“Amazon ECS (Elastic Container Service)

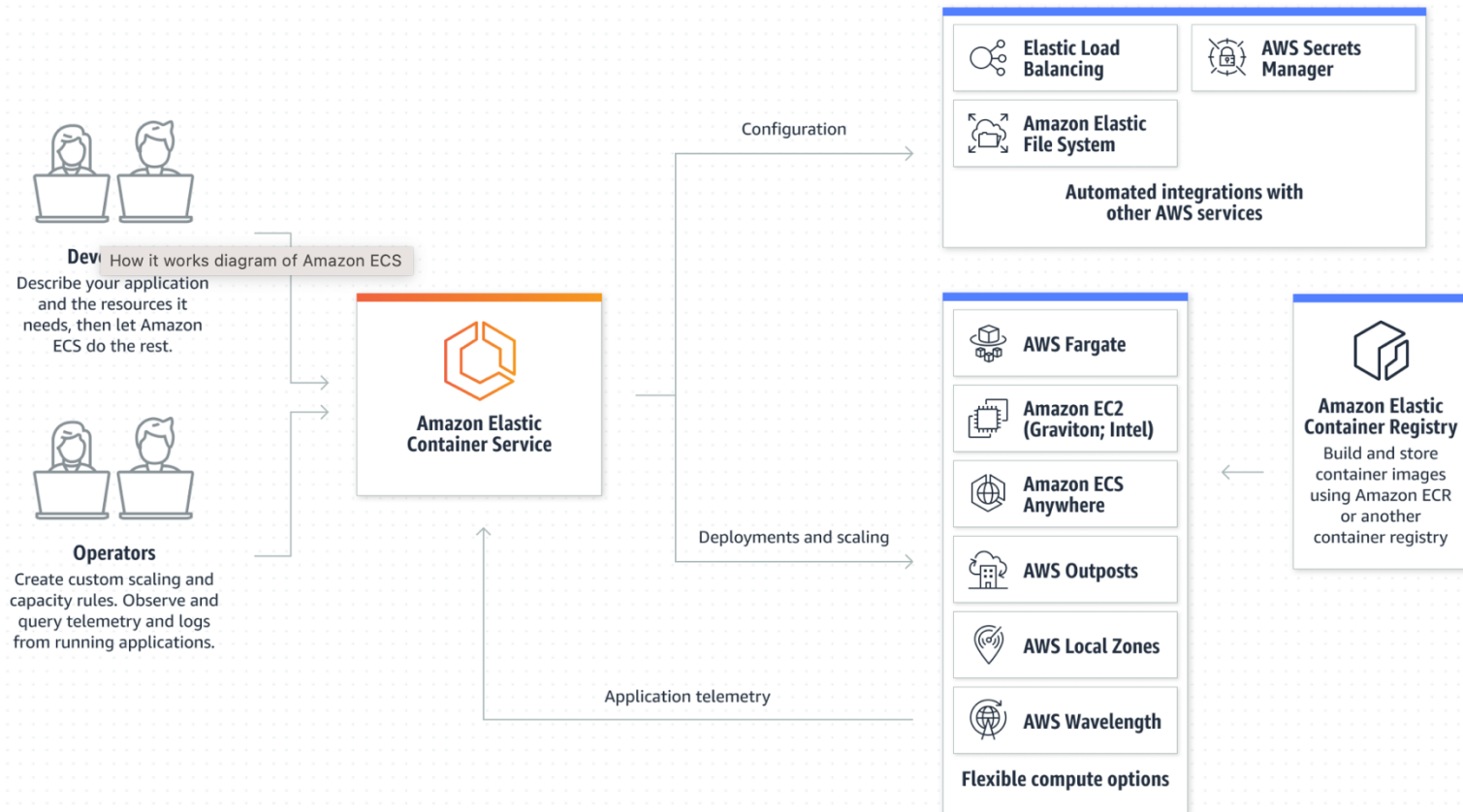
Amazon ECS (Elastic Container Service)

- Introduction to containerization and Docker:
 - Containerization is a lightweight virtualization technology that allows you to run applications in isolated environments called containers.
 - Docker is a popular containerization platform that simplifies the packaging and deployment of applications.
- Features and benefits of Amazon ECS:
 - Amazon ECS is a highly scalable and fully managed container orchestration service.
 - It provides the ability to deploy and manage containers using Docker images.
 - Benefits include simplified application deployment, automatic scaling, and built-in load balancing.

Amazon ECS (Elastic Container Service)

- Creating a task definition and a cluster:
 - A task definition is a blueprint that defines how a container should run, including container image, resource allocation, networking, and storage settings.
 - A cluster is a logical grouping of container instances that run your tasks.
 - Steps to create a task definition and a cluster, including defining container properties and configuring desired task count.
- Service scheduling and load balancing in ECS:
 - ECS allows you to create services, which define how many copies of a task should run and how they should be distributed across the cluster.
 - Load balancers can be integrated with ECS services to distribute incoming traffic across multiple containers.
 - Overview of service scheduling strategies, such as spread and binpack, for optimal resource utilization.

Amazon ECS (Elastic Container Service)



“Amazon EKS (Elastic Kubernetes Service)

Amazon EKS (Elastic Kubernetes Service)

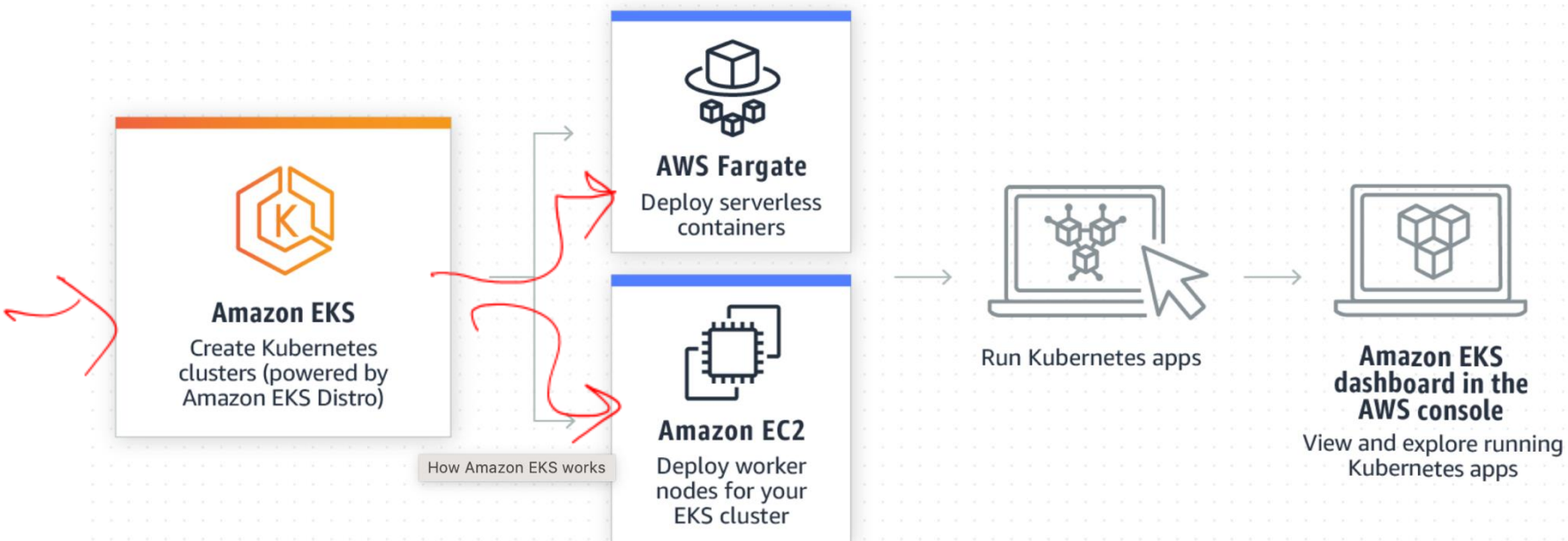
- Overview of Kubernetes and its importance:
 - Kubernetes is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications.
 - It provides a highly scalable and resilient environment for running containerized workloads.
- Benefits of using Amazon EKS:
 - Managed Kubernetes service: Amazon EKS eliminates the need to install and operate Kubernetes clusters, allowing developers to focus on application development rather than managing the underlying infrastructure.
 - Scalability and availability: EKS automatically scales the Kubernetes control plane and worker nodes, ensuring high availability and accommodating workload demands.

Amazon EKS (Elastic Kubernetes Service)

- Security and compliance: EKS integrates with AWS Identity and Access Management (IAM) and provides granular access control policies, enabling secure container deployments.
- Interoperability: EKS is compatible with standard Kubernetes tools and APIs, making it easy to migrate existing Kubernetes workloads to the AWS platform.
- Creating an EKS cluster:
 - Define a cluster configuration: Specify the desired number and type of worker nodes, networking settings, and security policies.
 - Launch the EKS cluster: Use the AWS Management Console, AWS CLI, or AWS CloudFormation to create the EKS cluster based on the defined configuration.
 - Connect to the cluster: Configure the Kubernetes command-line tool (kubectl) to connect to the EKS cluster and interact with the containerized applications.

Amazon EKS (Elastic Kubernetes Service)

- Managing and scaling containers with EKS:
 - Deploying applications: Use Kubernetes manifests or Helm charts to define and deploy containerized applications onto the EKS cluster.
 - Autoscaling: Configure horizontal pod autoscaling (HPA) to automatically scale the number of pods based on CPU or custom metrics.
 - Rolling updates and rollbacks: Perform seamless updates and rollbacks of application deployments without downtime, ensuring smooth application maintenance.
 - Monitoring and logging: Utilize Amazon CloudWatch and Kubernetes-native tools to monitor the health and performance of EKS clusters and applications.



“Choosing the Right Compute Service

Amazon EKS (Elastic Kubernetes Service)

- Workload Characteristics:
 - EC2: Ideal for workloads that require dedicated resources and customizable configurations, such as applications with specific hardware requirements.
 - ECS: Suitable for containerized applications that can benefit from automatic scaling, load balancing, and simplified management.
 - Lambda: Best for event-driven workloads and small, stateless functions that require quick execution and scalability.
 - EKS: Recommended for managing large-scale, containerized applications with Kubernetes orchestration and advanced management capabilities.

Amazon EKS (Elastic Kubernetes Service)

- Compute Flexibility:
 - EC2: Provides full control over the underlying infrastructure and allows for various operating systems, software installations, and networking configurations.
 - ECS: Offers a managed container service that abstracts the underlying infrastructure, simplifying deployment and scaling of containerized applications.
 - Lambda: Completely serverless, allowing developers to focus solely on writing code without worrying about provisioning or managing servers.
 - EKS: Enables running Kubernetes workloads with the flexibility to customize the cluster configuration and utilize Kubernetes ecosystem tools.

Amazon EKS (Elastic Kubernetes Service)

- Scaling Requirements:
 - EC2: Auto Scaling groups allow for scaling instances based on predefined policies, ensuring the desired capacity is maintained.
 - ECS: Supports automatic scaling of container instances and services, adjusting resources based on workload demands.
 - Lambda: Scales automatically and instantly in response to incoming requests or events, without the need for manual intervention.
 - EKS: Provides built-in scaling capabilities for managing the number of pods and nodes within the Kubernetes cluster.

Amazon EKS (Elastic Kubernetes Service)

- Cost Considerations:
 - EC2: Offers various pricing options (On-Demand, Reserved, and Spot Instances) to optimize costs based on workload characteristics and usage patterns.
 - ECS: Pricing is based on the underlying EC2 instances used by the containers, with similar cost optimization options available.
 - Lambda: Billed per invocation and the duration of function execution, making it cost-efficient for sporadic or low-traffic workloads.
 - EKS: Pricing is based on the EC2 instances running the EKS cluster, with similar cost optimization options as EC2.

Amazon EKS (Elastic Kubernetes Service)

- Hybrid and Multi-Cloud Deployments:
 - EC2: Enables seamless integration with on-premises infrastructure and hybrid cloud deployments through VPN or Direct Connect.
 - ECS: Supports hybrid deployments by integrating with on-premises container orchestrators or other cloud providers' container services.
 - Lambda: Can be integrated into hybrid architectures through API Gateway or AWS Step Functions, allowing execution of serverless functions alongside on-premises resources.
 - EKS: Offers multi-region and multi-cloud deployments by creating EKS clusters in different AWS regions or connecting to other Kubernetes clusters.

“Compute Service Integration

meets @
11 am

Compute Service Integration

- Amazon EC2 Integration:
 - Integration with Amazon RDS: Deploying databases on EC2 instances and establishing connections.
 - Integration with Amazon S3: Storing and retrieving data from S3 buckets using EC2 instances.
 - Integration with Elastic Load Balancer: Distributing traffic across multiple EC2 instances for high availability and scalability.
- Amazon ECS Integration:
 - Integration with Amazon S3: Storing container images in S3 buckets and accessing them during task definition creation.
 - Integration with Elastic Load Balancer: Load balancing traffic across containers in an ECS cluster.
 - Integration with AWS IAM: Managing access control to ECS resources.

Compute Service Integration

- AWS Lambda Integration:

- Integration with Amazon S3: Triggering Lambda functions in response to object creations, updates, or deletions in S3 buckets.
- Integration with Amazon DynamoDB: Executing Lambda functions in response to database events or table changes.
- Integration with AWS API Gateway: Building serverless APIs and exposing them as endpoints for Lambda function invocation.

- Amazon EKS Integration:

- Integration with AWS IAM: Defining fine-grained access control to EKS resources using IAM roles and policies.
- Integration with Amazon RDS: Deploying databases on EKS clusters and establishing connections.
- Integration with Elastic Load Balancer: Load balancing traffic across containers in an EKS cluster.

Compute Service Integration

- Examples of integrating compute services with other AWS services:
 - EC2 Integration Example:
 - Use case: Hosting a web application on EC2 instances with a MySQL database hosted on Amazon RDS.
 - Steps: Launch EC2 instances, configure security groups, set up IAM roles, create an RDS database, establish connectivity between the EC2 instances and the RDS database, and use Elastic Load Balancer for distributing incoming traffic.
 - ECS Integration Example:
 - Use case: Running a microservices architecture with containers in ECS, storing static assets in S3, and using Elastic Load Balancer for load balancing.
 - Steps: Create a task definition for each microservice, configure the ECS cluster, store container images in S3, set up IAM roles, and configure Elastic Load Balancer for routing traffic to containers.

Compute Service Integration

- Lambda Integration Example:
 - Use case: Processing uploaded files in an S3 bucket and storing the results in DynamoDB using a Lambda function.
 - Steps: Create an S3 bucket, create a Lambda function, configure triggers on the S3 bucket, set up IAM roles, and write code in the Lambda function to process the files and interact with DynamoDB.
- EKS Integration Example:
 - Use case: Running a scalable and resilient application on EKS, with a PostgreSQL database hosted on Amazon RDS.
 - Steps: Create an EKS cluster, deploy containers to the cluster, create an RDS PostgreSQL database, establish connectivity between the containers and the RDS database, and utilize Elastic Load Balancer for traffic distribution.

Compute Service Integration

- Leveraging integration for increased efficiency and productivity:
 - Integration with other AWS services enables seamless communication and data exchange between compute resources and other components of your application architecture.
 - By leveraging integration capabilities, you can build highly scalable, resilient, and interconnected systems on AWS.
 - Proper integration reduces manual effort, streamlines workflows, and allows you to focus on developing and deploying your applications rather than managing infrastructure connections.
 - Take advantage of the rich ecosystem of AWS services to enhance the capabilities and flexibility of your compute resources.

“

Amazon SageMaker

Amazon SageMaker

- What is Amazon SageMaker?
 - Fully managed service for building, training, and deploying machine learning models
 - Offers tools for every step of the machine learning workflow
- Key Features:
 - Data Labeling: Automated data labeling for training datasets
 - Model Training: Scalable training infrastructure
 - Model Deployment: One-click deployment of models to production
 - Model Monitoring: Continuous monitoring for accuracy and performance
- Benefits:
 - Scalability: Easily handle large datasets and complex models
 - Cost-Effective: Pay only for what you use
 - Productivity: Accelerate the ML development process

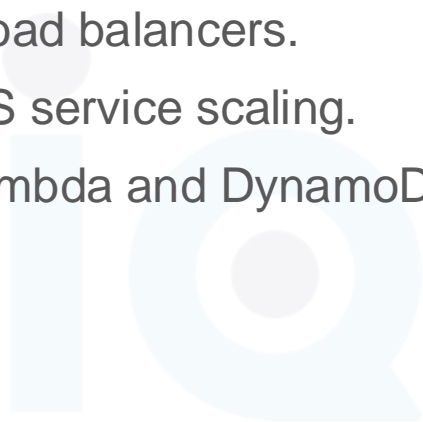
Amazon SageMaker

- Predictive Analytics:
 - Customer behavior predictions, sales forecasting
 - Natural Language Processing: Sentiment analysis, translation
 - Computer Vision: Image and video analysis
 - Personalization: Recommender systems for products and services
- Core Components:
 - SageMaker Studio: Integrated development environment for ML SageMaker
 - Ground Truth: Fully managed data labeling service
 - SageMaker Autopilot: Automatically create ML models with AutoML
 - SageMaker Neo: Optimize models to run up to 2x faster with no loss in accuracy

“High Availability and Scalability



High Availability and Scalability

- Designing highly available and scalable architectures.
 - Auto Scaling groups and load balancers.
 - EC2 Auto Scaling and ECS service scaling.
 - Serverless scaling with Lambda and DynamoDB.
- 

High Availability and Scalability

- Designing highly available and scalable architectures:
 - Ensure your applications are resilient and can handle increased traffic or workload demands.
 - Auto Scaling groups and load balancers: Utilize Auto Scaling groups to automatically adjust the number of EC2 instances or containers based on demand.
- Load balancers distribute traffic evenly across instances.
 - EC2 Auto Scaling and ECS service scaling: EC2 Auto Scaling allows you to automatically scale the number of EC2 instances based on predefined conditions.
 - ECS service scaling helps manage the number of tasks running in a service.
- Serverless scaling with Lambda and DynamoDB:
 - Lambda functions automatically scale based on the incoming request volume.
 - DynamoDB can handle massive scale and automatically adjusts capacity to match workload patterns.
 - Ensure your compute resources are designed for high availability and can scale to meet varying demands, enabling a seamless experience for your users.

“Security Best Practices



Security Best Practices

- Overview of AWS security features and best practices:
 - AWS provides a robust set of security features to protect your compute resources.
 - These include network security groups, virtual private clouds (VPCs), and encryption mechanisms.
 - Follow AWS's shared responsibility model, where AWS is responsible for the security "of" the cloud, and you are responsible for the security "in" the cloud.
- Identity and Access Management (IAM) roles and policies:
 - Implement least privilege access control by assigning IAM roles with only necessary permissions to users and services.
 - Regularly review and update IAM policies to ensure they align with your organization's security requirements.

Security Best Practices

- Network security groups and VPC configuration:
 - Utilize network security groups to control inbound and outbound traffic to your EC2 instances and containers.
 - Configure VPCs with appropriate subnets, route tables, and internet gateways to create isolated network environments.
- Securing containerized applications in ECS and EKS:
 - Implement container security best practices such as scanning container images for vulnerabilities, using secure container registries, and enforcing access controls.
 - Configure appropriate security group rules for containers to restrict network traffic and implement container-level isolation.

Security Best Practices

- Regularly update and patch your compute resources:
 - Stay up to date with security patches provided by AWS and regularly apply them to your EC2 instances, containers, and Lambda functions.
 - Enable automatic updates whenever possible to ensure you have the latest security fixes.
- Implement monitoring and logging:
 - Utilize Amazon CloudWatch to monitor compute resources, set up alarms, and receive notifications for suspicious activities or security breaches.
 - Enable detailed logging for your compute services and regularly review logs to identify any security incidents or anomalies.
- Implement encryption mechanisms: Encrypt data at rest using AWS Key Management Service (KMS) or customer-managed keys.
 - Implement encryption in transit using Secure Sockets Layer (SSL)/Transport Layer Security (TLS) certificates for communication between services.

Security Best Practices

- Regularly audit and review security settings:
 - Conduct regular security audits to identify any misconfigurations or vulnerabilities in your compute resources.
 - Leverage AWS Config and AWS Trusted Advisor to get insights into your security posture and receive recommendations for improvements.
- Employee training and awareness:
 - Train your employees on AWS security best practices, including the importance of ~~strong passwords~~, multi-factor authentication (MFA), and data protection.
 - Foster a culture of security awareness and ensure employees understand their roles and responsibilities in maintaining a secure environment.
 - Engage AWS security services: Explore additional AWS security services such as AWS WAF (Web Application Firewall), AWS Shield, and AWS GuardDuty to enhance the security of your compute resources.
 - Leverage AWS Security Hub for centralized security management and automated compliance checks

“Monitoring and Troubleshooting

Monitoring and Troubleshooting

- Monitoring compute resources with Amazon CloudWatch:
 - Amazon CloudWatch provides a comprehensive monitoring solution for AWS Compute services.
 - It enables you to collect and track metrics, collect and monitor log files, and set alarms.
 - You can monitor EC2 instances, containers in ECS and EKS, and Lambda functions.
- Setting up alarms and notifications:
 - Amazon CloudWatch alarms allow you to monitor metrics and trigger actions based on predefined thresholds.
 - You can set up alarms to notify you via email, SMS, or through other AWS services like SNS (Simple Notification Service) or Lambda functions.
 - Alarms help you proactively respond to potential issues and ensure the health and availability of your compute resources.

Monitoring and Troubleshooting

- Troubleshooting common issues with EC2 instances, containers, and Lambda functions:
 - EC2 instances: Monitor CPU utilization, memory usage, disk I/O, and network traffic to identify performance bottlenecks.
 - Containers: Monitor container health, resource utilization, and networking metrics to troubleshoot issues related to containerized applications.
 - Lambda functions: Monitor invocation counts, errors, and durations to identify performance issues or function failures.
- Logging and analyzing application logs:
 - AWS provides several services for logging and analyzing application logs, such as Amazon CloudWatch Logs and AWS X-Ray.
 - CloudWatch Logs enables you to centralize and store log data from EC2 instances, containers, and Lambda functions.
 - AWS X-Ray provides end-to-end visibility into distributed applications, allowing you to trace requests and identify performance bottlenecks.



Monitoring and Troubleshooting

- Regularly reviewing and analyzing monitoring data:
 - Monitoring data should be regularly reviewed to identify trends, detect anomalies, and optimize resource utilization.
 - Utilize CloudWatch dashboards, logs, and metrics to gain insights into the performance of your compute resources.
 - Use the data collected to optimize resource allocation, identify scaling needs, and improve overall application performance.

“Migration Strategies

Migration Strategies

- Migrating on-premises workloads to AWS Compute services:
 - Evaluate your existing on-premises infrastructure and workloads.
 - Identify applications and services suitable for migration to AWS Compute services.
 - Consider factors such as dependencies, resource requirements, and data transfer.
- Lift-and-shift vs. refactoring approaches:
 - Lift-and-shift: Migrate applications as-is to AWS without significant modifications.
 - Refactoring: Optimize applications for the cloud environment by rearchitecting or redesigning them.



Migration Strategies

- AWS Migration Hub and Server Migration Service:
 - AWS Migration Hub: Provides a single place to track and monitor your application migrations.
 - Server Migration Service: Simplifies the process of migrating on-premises servers to EC2 instances.



“Best Practices for Performance

Best Practices for Performance

- Optimizing performance for EC2 instances, containers, and serverless functions.
 - Utilize the appropriate instance type based on workload requirements and resource needs.
 - Implement horizontal scaling to handle increased traffic and demand.
 - Leverage auto-scaling groups to dynamically adjust resources based on workload patterns.
 - Use load balancers to distribute traffic evenly across multiple instances or containers.
 - Implement caching strategies to reduce the load on compute resources.
 - Utilizing AWS services like Amazon CloudFront and Elastic Load Balancer.

Best Practices for Performance

- Caching strategies and content delivery.
 - Implement caching mechanisms at various levels, such as CDN caching, in-memory caching, or database query caching, to reduce the response time and minimize compute resource utilization.
 - Leverage technologies like Amazon ElastiCache to store frequently accessed data in-memory, reducing the need for repetitive computations.
 - Use AWS CloudFront to cache static and dynamic content closer to end-users, reducing the latency and improving the overall performance.

Cost

Best Practices for Performance

- Performance monitoring and testing.
 - Utilize Amazon CloudWatch to monitor the performance metrics of compute resources, such as CPU utilization, network traffic, and memory usage.
 - Set up alarms and notifications to proactively identify and address performance bottlenecks or anomalies.
 - Conduct load testing and performance testing to simulate real-world traffic and identify areas for optimization.
 - Utilize AWS X-Ray for distributed tracing and performance analysis of microservices-based architectures.

“Summary

Summary

Ayush Sageman

Recap of key takeaways:

- Understand the various AWS Compute services such as EC2, ECS, Lambda, and EKS.
- Consider factors like scalability, cost, and integration when choosing the right compute service.
- Implement security best practices and monitor resources for optimal performance.
- Explore migration strategies and leverage AWS tools for smooth transitions.

Resources for further learning and documentation:

- AWS Compute Services documentation on the official AWS website.
- AWS Compute whitepapers and case studies for in-depth knowledge.

