**Akash Lalwani**

**A035**

**Cloud Computing**

**Practical-1 Cloud Computing**

**Github Link:** [**https://github.com/akashlalwanii/Cloud\_computing**](https://github.com/akashlalwanii/Cloud_computing)

**Q1.Cloud Computing Architecture?**

Cloud computing architecture refers to the components and subcomponents required for cloud computing. It combines hardware, software, storage, and networking elements, organized to deliver cloud services. Here are the key elements of cloud computing architecture:

**1. Front-End Layer**

1. **Client Devices**: End-user devices (like computers, tablets, smartphones) used to access the cloud.
2. **User Interface**: The interface through which users interact with cloud applications, often through web browsers or custom applications.

**2. Back-End Layer**

1. **Application**: Software or platforms provided as services (SaaS, PaaS) running on the cloud.
2. **Service Models**:
   1. **Software as a Service (SaaS)**: Provides end-user applications over the internet (e.g., Google Workspace, Salesforce).
   2. **Platform as a Service (PaaS)**: Offers a development platform, tools, and environment for developers (e.g., AWS Elastic Beanstalk, Google App Engine).
   3. **Infrastructure as a Service (IaaS)**: Supplies essential computing resources like storage, servers, and networks (e.g., Amazon EC2, Microsoft Azure).
3. **Storage**: Databases, data lakes, and storage solutions to hold data for applications (e.g., Amazon S3, Azure Blob Storage).
4. **Management and Security**: Tools and systems to manage resources, monitor performance, enforce policies, and provide security.
5. **Virtualization and Hypervisors**: Virtualization creates virtual versions of resources, allowing multiple applications to share the same physical resources.

**3. Cloud Deployment Models**

1. **Public Cloud**: Services are provided over the internet by third-party providers, offering scalability and cost-effectiveness (e.g., AWS, Google Cloud).
2. **Private Cloud**: Dedicated infrastructure for a single organization, offering better control over security and customization.
3. **Hybrid Cloud**: Combines public and private clouds, enabling data and application sharing between them, which offers flexibility and optimized resource usage.
4. **Multi-Cloud**: Involves using services from multiple cloud providers to avoid vendor lock-in and improve redundancy.

**4. Network Components**

1. Connects front-end and back-end layers, ensuring data flow and communication between client devices and cloud resources through high-speed internet connections, APIs, and protocols.

**5. Middleware**

1. Software layer that helps integrate cloud services with existing applications and connects different applications, enabling communication and data exchange.

A diagram of a cloud computing structure

Description automatically generated

**Q2.What is IAAS?**

Infrastructure as a Service (IaaS) is a cloud computing model that provides virtualized computing resources over the internet. In an IaaS model, a third-party provider hosts essential infrastructure components such as servers, storage, and networking hardware, along with the virtualization or hypervisor layer. IaaS allows organizations to rent IT infrastructure, on-demand, instead of buying and maintaining physical hardware.

**Key Features of IaaS**

1. **Scalability**: Resources can be scaled up or down based on demand, which is ideal for applications with fluctuating workloads.
2. **Cost Efficiency**: Pay-as-you-go pricing allows users to pay only for what they use, avoiding the costs of buying and maintaining hardware.
3. **Access and Control**: Users have direct control over their infrastructure through a management dashboard or API, allowing customization and integration with their applications.
4. **Flexibility**: Users can run any operating system or applications they need, as they have control over the virtual machines.

**Common Components of IaaS**

1. **Compute**: Virtual machines (VMs) that can be configured with various CPU, memory, and storage specifications.
2. **Storage**: Persistent storage for VMs, databases, and other data needs (e.g., block storage, file storage, and object storage).
3. **Networking**: Virtual networks, load balancers, and firewalls to manage traffic and secure the cloud environment.
4. **Data Backup and Recovery**: Ensures data safety with options for backup, disaster recovery, and redundancy.

**Examples of IaaS Providers**

1. **Amazon Web Services (AWS)** - EC2 (Elastic Compute Cloud), S3 (Simple Storage Service), and VPC (Virtual Private Cloud).
2. **Microsoft Azure** - Virtual Machines, Blob Storage, and Virtual Network.
3. **Google Cloud Platform (GCP)** - Compute Engine, Cloud Storage, and Virtual Private Cloud.

**Q3.What is AWS?**

Amazon Web Services (AWS) is a comprehensive and widely adopted cloud platform offered by Amazon. It provides a suite of cloud-based products and services that enable businesses and developers to build, deploy, and scale applications in the cloud. AWS offers solutions across multiple service models, including **Infrastructure as a Service (IaaS)**, **Platform as a Service (PaaS)**, and **Software as a Service (SaaS)**, making it versatile for various needs and workloads.

**Key Components of AWS**

**1. Compute Services**

1. **Amazon EC2 (Elastic Compute Cloud)**: Provides virtual servers (instances) for hosting applications, with options for different CPU, memory, and storage configurations.
2. **AWS Lambda**: Serverless computing service that lets you run code in response to events without managing servers.
3. **Amazon Elastic Beanstalk**: PaaS that automatically handles application deployment and scaling while developers focus on code.

**2. Storage Services**

* **Amazon S3 (Simple Storage Service)**: Scalable object storage for data backup, archival, and application data, known for its durability and high availability.
* **Amazon EBS (Elastic Block Store)**: Persistent block storage volumes for use with EC2 instances.
* **Amazon Glacier**: Low-cost archival storage for data that is accessed infrequently but needs long-term retention.

**3. Database Services**

* **Amazon RDS (Relational Database Service)**: Managed relational database service supporting databases like MySQL, PostgreSQL, Oracle, and SQL Server.
* **Amazon DynamoDB**: Fully managed NoSQL database service for fast and flexible applications.
* **Amazon Redshift**: Data warehousing solution optimized for large-scale data analytics.

**4. Networking Services**

* **Amazon VPC (Virtual Private Cloud)**: Enables users to define a private network within AWS, with control over IP ranges, subnets, and security groups.
* **Amazon Route 53**: Scalable Domain Name System (DNS) web service for domain registration and routing.
* **AWS Direct Connect**: Establishes a dedicated network connection from on-premises data centers to AWS for enhanced performance and security.

**5. Security and Identity**

1. **AWS IAM (Identity and Access Management)**: Manages user access and permissions to AWS resources.
2. **AWS Key Management Service (KMS)**: Provides encryption key management for securing data.
3. **AWS Shield**: Managed DDoS protection service for AWS-hosted applications.

**6. Machine Learning and AI Services**

1. **Amazon SageMaker**: Complete machine learning platform to build, train, and deploy ML models at scale.
2. **AWS Rekognition**: Image and video analysis service that can recognize objects, people, text, and more.
3. **Amazon Polly**: Text-to-speech service that converts written content into spoken language.

**7. Developer Tools**

1. **AWS CodePipeline**: Automates application delivery by building, testing, and deploying code updates.
2. **AWS CodeCommit**: Fully managed source control service that hosts Git repositories.
3. **AWS CodeBuild**: Compiles source code, runs tests, and produces deployable packages.

**8. Analytics**

1. **Amazon EMR (Elastic MapReduce)**: Managed big data processing using Apache Hadoop, Spark, and other tools.
2. **Amazon Kinesis**: Real-time data streaming and analytics.
3. **AWS Glue**: Managed ETL (extract, transform, load) service for data integration and preparation.

**Key Benefits of AWS**

1. **Scalability**: Instantly scale up or down based on demand with a pay-as-you-go pricing model.
2. **Reliability**: AWS has a robust infrastructure with data centers in regions worldwide, ensuring availability and redundancy.
3. **Security**: AWS provides advanced security tools and compliance certifications, making it suitable for sensitive and regulated workloads.
4. **Global Reach**: AWS spans multiple geographic regions and availability zones, enabling organizations to deploy applications closer to users around the world.
5. **Cost Efficiency**: AWS offers flexible pricing models, including on-demand, reserved, and spot instances, to fit various budget needs.

**Popular Use Cases**

1. **Web and Mobile Applications**: AWS supports quick deployment of applications and auto-scaling for variable traffic.
2. **Data Warehousing and Big Data**: Tools like Amazon Redshift and AWS EMR make it easy to store, process, and analyze massive amounts of data.
3. **Machine Learning and AI**: AWS offers a suite of tools for developing, training, and deploying machine learning models.

**Q4.What is EC2 instance?**

Amazon Elastic Compute Cloud (Amazon EC2) is a service within AWS that provides scalable, on-demand compute capacity in the cloud. It allows users to rent virtual servers, known as "instances," where they can run applications, host websites, process data, and more. EC2 enables businesses to avoid the hardware investment and operational overhead of maintaining physical servers while offering flexibility and cost-effective scaling.

**Key Features of EC2**

1. **Instance Types**:
   1. EC2 offers a variety of instance types optimized for different workloads, including general-purpose, compute-optimized, memory-optimized, storage-optimized, and accelerated computing instances.
   2. Examples:
      1. **t2.micro**: A small, general-purpose instance for low-traffic websites or small applications.
      2. **c5.large**: Compute-optimized instance for high-performance computing tasks.
      3. **r5.large**: Memory-optimized for applications with high memory requirements, like databases.
2. **Scalability and Auto Scaling**:
   1. EC2 can scale up or down based on demand. AWS Auto Scaling allows EC2 instances to automatically adjust in response to traffic and load, optimizing performance and cost.
3. **Elastic Load Balancing (ELB)**:
   1. Automatically distributes incoming traffic across multiple instances, improving fault tolerance and reliability.
4. **Elastic Block Store (EBS)**:
   1. EBS provides persistent block storage for EC2 instances, allowing data to be saved and retained independently of the lifecycle of EC2 instances.
5. **Security and Access Control**:
   1. EC2 integrates with AWS Identity and Access Management (IAM) for secure access and permissions.
   2. Security groups act as virtual firewalls, controlling incoming and outgoing traffic to and from instances.
6. **Virtual Private Cloud (VPC) Integration**:
   1. EC2 instances can be launched within a VPC, giving users complete control over their virtual networking environment.
7. **Flexible Pricing Models**:
   1. **On-Demand Instances**: Pay per second/hour with no long-term commitment.
   2. **Reserved Instances**: Commit to a 1- or 3-year term for a lower hourly rate, suited for predictable workloads.
   3. **Spot Instances**: Purchase unused capacity at a reduced rate, ideal for flexible or non-time-sensitive tasks.
   4. **Savings Plans**: Flexible pricing options for users with consistent workloads across multiple instance types.
8. **Instance Lifecycle and Management**:
   1. EC2 allows starting, stopping, rebooting, and terminating instances, making it flexible for temporary or long-term use cases.

**Popular Use Cases**

1. **Web Hosting**: Host websites and web applications with customizable instances.
2. **Application Development and Testing**: Set up development and testing environments that can be quickly scaled up or down.
3. **Big Data Processing**: Process and analyze large datasets using EC2 with tools like Hadoop or Spark.
4. **Machine Learning and High-Performance Computing (HPC)**: EC2 provides powerful GPU and compute-optimized instances for AI, ML, and HPC applications.
5. **Batch Processing**: Run large-scale batch jobs at reduced costs using Spot Instances.

**Benefits of Using EC2**

1. **Flexibility**: A wide range of instance types and configurations suited to different workloads.
2. **Scalability**: Easily scale instances up or down based on demand, including automatic scaling.
3. **Cost Efficiency**: Multiple pricing options allow you to optimize costs based on workload predictability.
4. **Reliability**: High availability and fault tolerance are supported through load balancing and multi-AZ (Availability Zone) deployments.
5. **Integration with AWS Ecosystem**: Seamless integration with other AWS services like S3, RDS, and Lambda.

**Akash Lalwani**

**A035**

**Cloud Computing**

**Practical-2 SAAS**

**Github Link:** [**https://github.com/akashlalwanii/Cloud\_computing**](https://github.com/akashlalwanii/Cloud_computing)

**Q1. What is S3?**

Amazon Simple Storage Service (Amazon S3) is an object storage service provided by AWS, designed to store and retrieve large amounts of data securely, affordably, and with high availability. S3 is widely used for everything from simple file storage to data lakes, backups, and content distribution.

**Key Features of Amazon S3**

1. **Object Storage**:
   1. S3 is an object storage system, meaning data is stored as individual objects in a flat namespace rather than as files within a hierarchical file system.
   2. Each object consists of data, metadata, and a unique identifier within a bucket, which acts as a storage container.
2. **Buckets and Objects**:
   1. **Bucket**: A logical container for storing objects (files). Each bucket has a unique name within AWS and can hold unlimited objects.
   2. **Object**: Any data file (image, video, document, etc.) with a unique key in the bucket. Objects are accessed via their unique key.
3. **Storage Classes**:
   1. **Standard**: High durability and availability, ideal for frequently accessed data.
   2. **Intelligent-Tiering**: Automatically moves data between two access tiers (frequent and infrequent) to optimize cost.
   3. **Standard-IA (Infrequent Access)**: Lower-cost storage for less frequently accessed data.
   4. **One Zone-IA**: Lower-cost option for infrequent access, with data stored in a single availability zone.
   5. **Glacier** and **Glacier Deep Archive**: Low-cost archival storage for data that is rarely accessed, suitable for long-term retention and compliance.
4. **Scalability and Durability**:
   1. S3 offers virtually unlimited scalability, allowing you to store as much data as needed.
   2. Designed for **99.999999999% durability** (11 nines), meaning data is redundantly stored across multiple devices and locations.
5. **Data Security and Access Control**:
   1. **Encryption**: Supports both server-side and client-side encryption to protect data.
   2. **Access Control**: Uses AWS Identity and Access Management (IAM), bucket policies, and access control lists (ACLs) to define access permissions.
   3. **S3 Object Lock**: Prevents deletion or modification of objects, making it suitable for regulatory compliance.
6. **Data Management and Storage Analytics**:
   1. **Lifecycle Policies**: Automates data movement between storage classes based on rules for archiving, deletion, or transition to more cost-effective storage.
   2. **Replication**: Replicates objects across regions or accounts for data redundancy or compliance.
   3. **S3 Inventory**: Provides reports on objects and their metadata for tracking and analysis.
7. **Event Notifications**:
   1. S3 can trigger AWS services like Lambda, SNS, or SQS in response to events (e.g., new object creation or deletion), enabling automation.
8. **Data Transfer and Access**:
   1. **S3 Transfer Acceleration**: Speeds up data transfer to S3 by using Amazon CloudFront’s global network.
   2. **Public Access and Content Distribution**: Supports public access to objects and integration with Amazon CloudFront for global content distribution.

**Common Use Cases**

1. **Backup and Disaster Recovery**: S3 is often used for data backups, snapshots, and disaster recovery due to its durability and secure storage options.
2. **Data Lake and Big Data Storage**: S3’s scalability and integration with analytics tools make it ideal for data lakes and big data processing.
3. **Static Website Hosting**: S3 can host static websites, serving HTML, CSS, JavaScript, and media files directly from a bucket.
4. **Content Storage and Delivery**: Frequently used for storing and distributing media content, such as images, videos, and documents.
5. **Log Storage and Analysis**: Store application, server, and user logs for analysis with tools like Amazon Athena.

**Q2.S3 use cases?**

Amazon S3 (Simple Storage Service) has diverse use cases due to its scalable, reliable, and secure cloud storage capabilities. Here are some of the primary ways it is used:

**1. Backup and Disaster Recovery**

1. **Data Backup**: Store critical data backups, such as server snapshots, database backups, and user data, due to S3's durability (99.999999999%).
2. **Disaster Recovery**: With cross-region replication and multi-AZ storage options, S3 ensures data availability even if one location is compromised.

**2. Data Lake for Big Data and Analytics**

1. **Data Lakes**: Serve as a centralized repository for storing structured and unstructured data at any scale. S3 is compatible with data lake architectures and supports storage and processing of massive datasets.
2. **Analytics**: Integrated with services like Amazon Athena (SQL queries) and Amazon EMR (Hadoop processing) for big data processing directly from S3.

**3. Content Storage and Distribution**

1. **Media Hosting**: Stores and distributes images, videos, audio files, and other media for websites and applications.
2. **Software Distribution**: Ideal for storing and distributing large software packages and updates for mobile and desktop applications.
3. **Integration with CDN**: Works seamlessly with Amazon CloudFront to accelerate content delivery globally.

**4. Static Website Hosting**

1. **Static Site Hosting**: S3 can host static websites (HTML, CSS, JavaScript) without needing a web server. Public URLs enable direct access to site assets and content.
2. **Public File Hosting**: Use S3 to share public files with simple URLs.

**5. Application Data Storage**

1. **Storage for Web Applications**: Acts as a backend for applications to store user-uploaded content, images, and documents.
2. **Logs and User Data**: Efficient for storing large quantities of log files, user preferences, session data, and more.

**6. Machine Learning and Data Processing**

1. **Training Data Storage**: Use S3 to store training data for machine learning models. It integrates with Amazon SageMaker for seamless access to data during training.
2. **Data Processing Pipelines**: S3 can act as a staging area for ETL (Extract, Transform, Load) operations, with data processed by services like AWS Glue.

**7. Compliance and Archival Storage**

* **Long-term Archival**: S3 Glacier and Glacier Deep Archive storage classes are ideal for data that must be retained for regulatory or compliance purposes, at low cost.
* **Data Compliance**: S3 Object Lock provides WORM (Write Once, Read Many) storage, ensuring data cannot be deleted or modified, making it suitable for compliance with data retention policies.

**8. Log Storage and Analysis**

* **Log Aggregation**: Centralize log storage from multiple sources, including server logs, API logs, and transaction logs, for analysis and troubleshooting.
* **Real-Time Analysis**: S3 can trigger events to AWS Lambda or Amazon Kinesis for real-time log processing and monitoring.

**9. Mobile and Web Application Hosting**

* **Resource Storage**: Mobile and web applications can store static assets like images, videos, or app data in S3, with access managed through the AWS Mobile SDK.
* **Data Caching**: S3 can serve as a temporary cache for application resources to manage high traffic and load balancing.

**10. IoT Data Storage**

1. **Sensor and Device Data**: IoT devices can upload and store data directly in S3 for processing, analysis, and machine learning applications.
2. **Time-Series Data Storage**: Store continuous streams of IoT data that can be analyzed over time for insights or trends.

**11. Media Processing**

1. **Transcoding and Compression**: Store media files for transcoding (e.g., video and audio) using AWS Elemental MediaConvert or other AWS media processing tools.
2. **Image and Video Storage for Processing Pipelines**: S3 can serve as both the input and output location for media processing workflows.

**12. Software Development and Deployment Artifacts**

1. **CI/CD Pipelines**: Store build artifacts, deployment packages, and container images as part of continuous integration and delivery (CI/CD) pipelines.
2. **Versioning and Rollback**: S3 versioning helps keep multiple versions of the same file, providing rollback options if needed.

**13. GIS and Mapping Data Storage**

1. **Geospatial Data**: Used to store GIS datasets, satellite imagery, and mapping data for geographic information systems (GIS) applications.
2. **Data Processing and Visualization**: Easily integrates with tools and services for processing and visualizing GIS data, such as Amazon EMR and Amazon SageMaker.

**Q3**.How to create S3 bucket?

Creating an Amazon S3 bucket is a straightforward process. Here are the steps to create an S3 bucket using the AWS Management Console:

**Step 1: Log in to AWS**

1. Go to the [AWS Management Console](https://aws.amazon.com/console/) and log in with your credentials.

**Step 2: Navigate to Amazon S3**

1. From the AWS Console, search for "S3" in the search bar or find it in the "Storage" section.

**Step 3: Create a New Bucket**

1. **Click "Create bucket"**:
   1. This button is located at the top right of the S3 dashboard.
2. **Configure Bucket Settings**:
   1. **Bucket Name**: Enter a unique name for the bucket (bucket names must be globally unique across all AWS users and comply with specific naming rules).
   2. **Region**: Select the AWS region where you want to create the bucket. Consider choosing a region close to your users to reduce latency.
3. **Configure Options (Optional)**:
   1. **Object Ownership**: Choose between "ACLs enabled" or "ACLs disabled" (Access Control Lists). Generally, "ACLs disabled" is recommended for secure bucket access control.
   2. **Block Public Access**: Configure public access settings based on your use case. By default, S3 blocks all public access for security. You can later modify these settings if you need public access (for example, to host a static website).
   3. **Bucket Versioning**: Enable versioning if you want to keep multiple versions of objects.
   4. **Tags**: Add key-value pairs as tags for organizing and identifying resources.
   5. **Default Encryption**: Enable server-side encryption to automatically encrypt all objects stored in the bucket.
4. **Set Advanced Options** (Optional):
   1. **Object Lock**: Enable this to enforce WORM (Write Once, Read Many) restrictions on objects, which is useful for compliance.
5. **Review and Create**:
   1. Review your bucket settings and ensure everything is configured as needed.
   2. Click **Create bucket** to complete the setup.

**Step 4: Upload Data (Optional)**

1. After creating the bucket, you can start uploading objects (files) by selecting the bucket and using the **Upload** option.

**Step 5: Configure Bucket Policies and Permissions (If Needed)**

1. After creating the bucket, you can set up permissions:
   1. Use **Bucket Policy** to specify permissions for specific users or groups.
   2. Use **Access Control Lists (ACLs)** if needed to grant access to other AWS accounts.

**Step 6: Additional Configurations (Optional)**

1. **Enable Logging or Analytics**: Set up logging or analytics to track access and usage.
2. **Lifecycle Policies**: Create policies to automatically transition data to different storage classes or delete objects after a certain period.

**Step 7: Access Your Bucket**

1. Your bucket is now ready, and you can start accessing it through the console, AWS CLI, SDKs, or APIs.

Cloud Computing Practical 3

Identity Access Management (IAM)

Name: Akash Lalwani

Roll No.: A035

**Github Link:** [**https://github.com/akashlalwanii/Cloud\_computing**](https://github.com/akashlalwanii/Cloud_computing)

**Write-up:**

**Users and Groups**

**In cloud environments, Users and Groups are essential components in managing and organizing access control.**

**- Users represent individual identities that require access to resources. They can be employees, contractors, or applications that need permissions to operate in the system. Each user is given a unique identity within the organization, allowing for customized access and permissions.**

**- Groups are collections of users with similar access needs. Rather than assigning permissions to each user individually, administrators can create groups and assign specific permissions to the group, simplifying management. For example, a "Developers" group might be given permission to deploy applications, while a "Support" group could be restricted to viewing logs and system statuses.**

**Together, Users and Groups allow administrators to streamline access control policies, manage permissions more efficiently, and ensure that only authorized individuals have access to specific resources.**

**---**

**IAM (Identity and Access Management)**

**IAM (Identity and Access Management) is a critical framework in cloud security that enables organizations to define, manage, and control user access to resources. IAM provides a centralized way to create and manage identities, roles, and policies that specify what level of access each user or application has to resources.**

**The key functions of IAM include:**

**- Authentication: Verifying that users are who they claim to be, typically via passwords, multi-factor authentication, or single sign-on (SSO).**

**- Authorization: Granting the correct level of access to authenticated users based on their roles and policies.**

**- User Management: Creating, modifying, and deleting user accounts as employees join, leave, or change roles within the organization.**

**- Policy Management: Defining permissions and rules that control what resources users or groups can access and what actions they can perform.**

**IAM plays a crucial role in maintaining security and compliance, helping organizations avoid unauthorized access and safeguard sensitive data.**

**---**

**Role of IAM**

**The Role of IAM extends beyond just assigning access; it serves as the foundation of security and governance within an organization. IAM’s responsibilities are essential for:**

**1. Enhancing Security: By ensuring that only authorized individuals or systems can access specific resources, IAM reduces the risk of unauthorized access or data breaches.**

**2. Maintaining Compliance: Many industries require strict access control for regulatory compliance. IAM provides the necessary tools to meet these standards, often with detailed logging and auditing capabilities.**

**3. Improving Operational Efficiency: IAM simplifies access management by using roles and groups, reducing the administrative workload associated with granting and revoking access.**

**4. Supporting Scalability: As organizations grow, IAM makes it easier to manage thousands of users and their permissions across various systems, applications, and resources.**

**IAM is essential in cloud environments, where resources are highly distributed and continuously scaled, making access control crucial for effective governance and security.**

Cloud Computing Practical 7:

Creating and Running Virtual Machine on BAre-Metal Hypervisors

Name: Akash Lalwani

Roll no: A035

**Github Link:** [**https://github.com/akashlalwanii/Cloud\_computing**](https://github.com/akashlalwanii/Cloud_computing)

Write-up:

**Bare-Metal Hypervisors**

A Bare-Metal Hypervisor is a type of hypervisor that installs directly on the physical hardware, acting as the primary operating system. It manages the hardware resources and virtualizes them for the virtual machines (VMs) running on top of it. Unlike hosted hypervisors, bare-metal hypervisors do not need a separate OS to function, making them highly efficient, secure, and stable for enterprise-grade applications.

---

**Bare-Metal Hypervisors Type 0**

Type 0 Bare-Metal Hypervisors, sometimes referred to as firmware hypervisors, are embedded directly into the server hardware. This type of hypervisor is common in large, high-performance systems such as mainframes. Due to their close integration with hardware, they offer extremely low overhead and high performance, but they are generally specialized and not widely used in commercial or consumer-grade environments.

---

**Bare-Metal Hypervisors Type 1**

Type 1 Bare-Metal Hypervisors, also known as native or bare-metal hypervisors, install directly on the physical hardware without the need for an underlying operating system. These hypervisors provide an isolated environment for virtual machines and are typically used in data centers and enterprise environments. Examples of Type 1 hypervisors include VMware ESXi, Microsoft Hyper-V, and Xen. Type 1 hypervisors are known for their stability, performance, and efficient use of system resources.

---

**VMware**

VMware is a popular virtualization platform that offers both bare-metal and hosted hypervisors. VMware's Type 1 hypervisor, ESXi, is widely used in enterprise environments for server virtualization, providing robust management tools, security features, and high performance. Additionally, VMware offers VMware Workstation, a Type 2 hosted hypervisor used for running virtual machines on a host operating system. VMware allows users to create and manage virtual environments efficiently and is compatible with various guest operating systems.

---

**VirtualBox**

VirtualBox is an open-source hosted hypervisor developed by Oracle, primarily used on desktops and laptops. As a Type 2 hypervisor, VirtualBox installs on top of an existing operating system, allowing users to run multiple guest operating systems concurrently. It supports a variety of host and guest systems, including Windows, Linux, and macOS. VirtualBox is known for its ease of use, flexibility, and compatibility, making it a popular choice for personal use and development.

---

**Using VMware and VirtualBox to Implement Windows 7 and Ubuntu Operating Systems**

Both VMware and VirtualBox can be used to install and run different operating systems in virtual environments. Here’s a brief guide on how to set up Windows 7 and Ubuntu using each platform:

**1. \*\*VMware\*\*:**

- Download and install VMware Workstation for a hosted environment or VMware ESXi for a bare-metal setup.

- Create a new virtual machine and select the desired operating system (e.g., Windows 7 or Ubuntu).

- Allocate system resources such as CPU, RAM, and disk space based on the requirements of the OS.

- Install the OS using an ISO file or installation media, following the on-screen setup instructions.

**2. \*\*VirtualBox\*\*:**

- Download and install VirtualBox from Oracle.

- Open VirtualBox and create a new virtual machine, specifying the operating system (Windows 7 or Ubuntu) and version.

- Configure resources such as memory and storage, then attach the OS installation ISO.

- Start the virtual machine, and follow the installation steps to set up Windows 7 or Ubuntu.

Both VMware and VirtualBox provide the flexibility to run multiple operating systems on a single host, making it easy to experiment with and deploy different environments without needing additional hardware.

Cloud Computing Practical

Name: Akash Lalwani

Roll no: A035

**Github Link:** [**https://github.com/akashlalwanii/Cloud\_computing**](https://github.com/akashlalwanii/Cloud_computing)

Write-up:

**1. Platform as a Service (PaaS)**

Platform as a Service (PaaS) is a cloud computing model that offers a complete development and deployment environment in the cloud. This model allows developers to focus on application creation without worrying about the underlying infrastructure, which includes OS, middleware, and runtime configurations. PaaS supports various development and testing capabilities, enabling users to create applications more efficiently while reducing the complexity of deployment and scaling.

**Key Features of PaaS:**

- Simplifies app development with built-in software stacks and services.

- Automates infrastructure provisioning, scaling, and management.

- Provides flexibility to support multiple programming languages and frameworks.

**Examples of PaaS:** AWS Elastic Beanstalk, Google App Engine, Microsoft Azure App Service.

**2. AWS Elastic Beanstalk**

AWS Elastic Beanstalk is a PaaS offering from Amazon Web Services (AWS) designed to simplify application deployment. It enables developers to deploy and manage applications without managing the infrastructure. With Elastic Beanstalk, users can quickly deploy applications in various languages (such as Java, Python, Node.js, and Ruby) by uploading the application code. Beanstalk handles the rest—provisioning the resources, load balancing, autoscaling, and monitoring the application’s health.

**Benefits of Elastic Beanstalk:**

- Automation: Manages underlying infrastructure, making deployment and scaling easier.

- Flexibility: Allows customization of AWS resources, providing control over EC2 instances, databases, and networking settings.

- Cost-Efficiency: Users only pay for the resources used, while Beanstalk itself is free of charge.

**3. Components of Elastic Beanstalk**

AWS Elastic Beanstalk consists of several key components that work together to provide a streamlined environment for application deployment:

**- Environment:** The logical construct where your application runs, containing all the necessary resources (like EC2 instances, load balancers, and databases).

**- Application:** A collection of Elastic Beanstalk environments, settings, and application versions. It represents the overall project or system you are building.

**- Environment Tiers:**

- Web Server Tier: For handling HTTP(S) requests from clients (used for web applications).

- Worker Tier: For applications with background processing tasks or queuing mechanisms.

**- Environment Configuration:** Configurations related to instance types, autoscaling policies, load balancing, and software settings (e.g., runtime, platform).

**- Application Versions:** Different iterations of your application’s code. Each version can be deployed to different environments, allowing for testing, staging, and production control.

**- Elastic Beanstalk Command Line Interface (EB CLI):** A tool for managing applications and environments from the command line, providing ease in deployment and updates.

**4. IAM (Identity and Access Management)**

AWS Identity and Access Management (IAM) is a web service that allows secure control over AWS resources. IAM enables users to create and manage AWS users and groups and assign permissions to control access to resources. It’s essential in environments like Elastic Beanstalk for managing who can access, deploy, or make changes to applications, ensuring only authorized users can manage the environment.

**Key IAM Concepts in Elastic Beanstalk:**

- IAM Users: Individual accounts for people or applications interacting with AWS. Permissions can be assigned to control their actions within Elastic Beanstalk.

- IAM Roles: Used to grant permissions to applications or services. Elastic Beanstalk itself requires an IAM role to interact with resources (e.g., creating instances, accessing S3 for storage).

- Policies: JSON documents that define permissions. Policies are attached to IAM users, groups, or roles, determining what actions they can perform on Elastic Beanstalk resources.

**Conclusion:**

AWS Elastic Beanstalk, as a PaaS, offers a powerful and automated solution for application deployment. By integrating with IAM, Beanstalk maintains secure access and resource management, simplifying the deployment process while ensuring control over user permissions. This integration allows developers to focus on application logic and performance rather than infrastructure, accelerating development cycles and improving efficiency.