Amazon EBS

April 2024

Mahmoud Najmeh

Contents

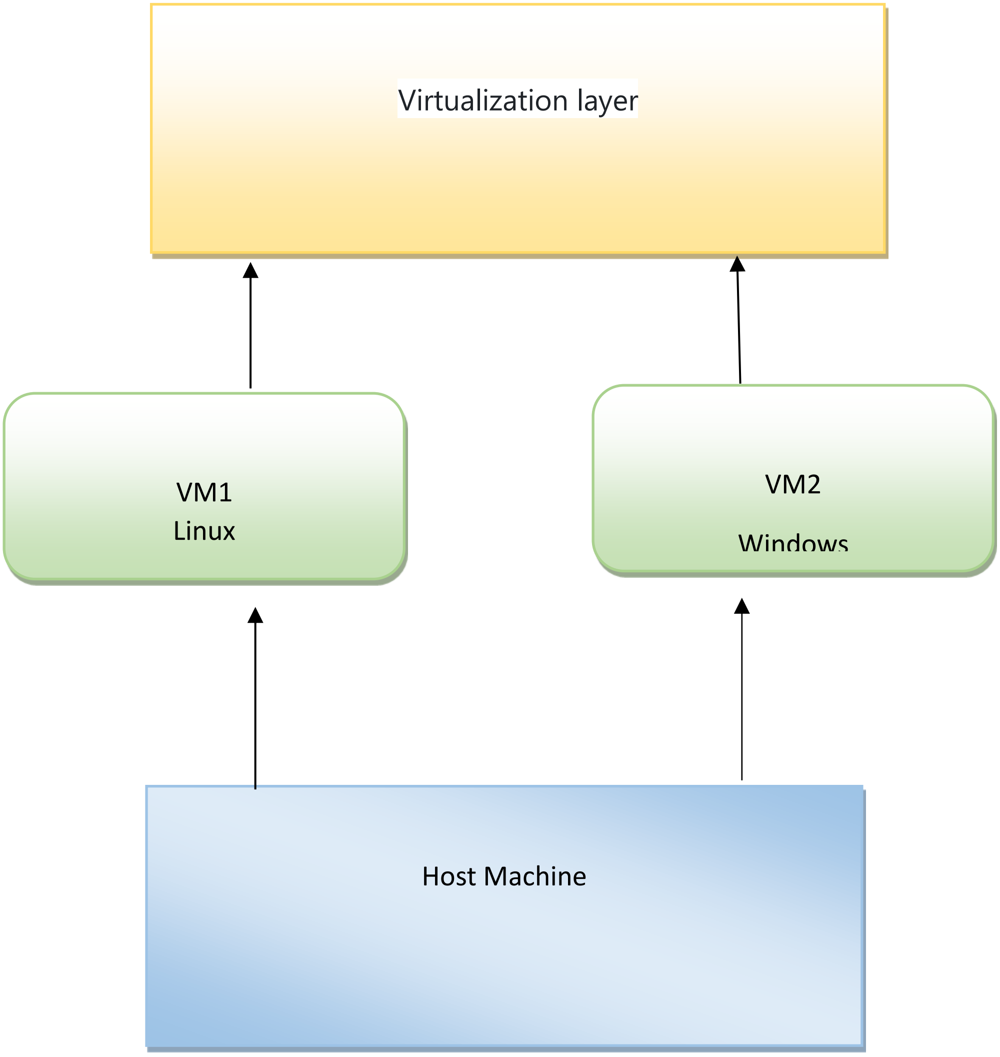
AWS

Exercise 1

Virtualization:

Virtualization is a technology that allows you to create virtual representations of servers, storage, networking, and other physical devices.

Diagram:



List of 3 different types of Virtualization:

1. Hardware Virtualization:
   A hypervisor on the physical server manages virtual machines (VMs) in complete isolation.
2. Operating System Virtualization (Containerization):
   Containers run on a single OS kernel, making them lightweight and efficient.
3. Storage Virtualization:
   Abstracts physical storage into a single pool for easier management and better resource utilization.

Advantages and disadvantages of Virtualization:

Advantages:

- Cost saving
- Increased Efficiency
- Enhanced Security

Disadvantages:

- Complexity
- Hardware Reliance
- Security Concerns

Exercise 2:

Comparative Analysis: Block-Level Storage vs. Object-Level Storage

Choosing the right storage solution depends on the specific needs of your data. This analysis dives into the key differences between block-level storage and object-level storage to help you make an informed decision.

Feature	Block-Level Storage	Object-Level Storage
Data Organization	Fixed-size blocks	Self-contained objects with metadata
Performance	High for random access, low latency	High for sequential access, variable latency
Scalability	Limited scalability, requires adding physical disks	Highly scalable, elastically sizes based on demand
Cost	Higher cost per GB	Lower cost per GB for large data sets
Management Complexity	Less complex, familiar file system structure	More complex, requires additional metadata management
Use Cases	Databases, applications requiring fast data access, transactional workloads	Backups, archives, media files, large unstructured data sets

Real-World Examples:

- Block-Level Storage:
    - Running a database server
    - Hosting a virtual machine
- Object-Level Storage:
    - Cloud storage services
    - Big data analytics

Choosing the Right Storage Solution

The choice between block-level and object-level storage hinges on your data access patterns and needs:

- For applications requiring fast, random access to data (databases, VMs), block storage is the preferred option due to its lower latency.
- For storing large, static data sets (backups, archives, media) that don't require frequent access, object storage offers superior scalability and cost-effectiveness.

Additional Considerations:

- Security: Both storage types offer security features, but object storage's built-in metadata management can enhance access control and versioning.
- Durability: Both offer data redundancy options, but object storage often replicates data across geographically dispersed locations for increased data durability.

Exercise 3:

Amazon EBS Data Model Architecture

Amazon Elastic Block Store (EBS) utilizes a block-level data storage model. This means data is organized into fixed-size blocks, typically 1 MiB in size, which are managed by EBS. These blocks are exposed to users as volumes that can be attached to EC2 instances (virtual machines on AWS).

Components of the EBS Data Model:

Volumes: The fundamental unit of storage in EBS. Volumes are virtual disks that can be formatted with various file systems and used for data storage by EC2 instances.

Snapshots: Point-in-time copies of entire volumes. They capture the complete state of a volume at a specific moment and can be used for backups, disaster recovery, or creating new volumes.

Interaction between Components:

- EC2 instances can attach volumes and access data stored within the blocks.
- Users can create snapshots of volumes for backups or to create new volumes from existing ones.
- Snapshots are stored redundantly within the chosen AZ, ensuring data availability.

Comparison with S3 and Glacier

Feature	EBS	S3	Glacier
Data Model	Block-level	Object-level	Object-level
Access Pattern	Optimized for random access	Optimized for sequential access	Optimized for infrequent access
Scalability	Scales by adding volumes	Highly scalable, elastically scales based on needs	Highly scalable, pay-per-use model
Use Cases	Databases, applications requiring fast data access	Backups, archives, large data sets	Long-term archiving, data retrieval is slower
Cost	Higher cost per GB	Lower cost per GB for large data sets	Lower cost per GB

Scalability and Performance Considerations of EBS:

- Scalability: Enables rapid scaling of compute capacity to meet fluctuating demand, ensuring a smooth user experience during peak times.
- Performance: io1 volumes provide predictable and consistent IOPS performance, making them ideal for I/O-intensive applications.

Choosing the Right AWS Storage Solution:

- For applications requiring fast, random access to data (databases, VMs): EBS is the ideal choice due to its high performance and low latency.
- For storing large, static data sets (backups, archives, media) that don't require frequent access: S3 or Glacier offer superior scalability and cost-effectiveness. S3 provides a good balance between cost and access times, while Glacier is the most cost-effective option for long-term archiving with slower retrieval times.

Exercise 4:

Table listing the different types of EBS volumes

Type	Performance	Cost	Durability	Use cases
Gp2	Balanced	Moderate	High	General purpose workloads, databases with moderate IOPS needs (e.g., development environments, web servers)
Io1	High	High	High	I/O-intensive workloads, databases with predictable high IOPS (e.g., real-time analytics, large-scale databases)
St1	Bursty	Low	High	Bursty workloads, applications with unpredictable spikes in IOPS (e.g., batch processing, log processing)
Sc1	Cold Storge	Very Low	High	Infrequent access workloads, archival data, backups (e.g., disaster recovery backups, long-term data storage)

Use Case Scenarios:

- gp2 Volumes: A good fit for most workloads due to their balanced performance and cost. Ideal for development and test environments, web servers with moderate database needs, and general-purpose applications.
- io1 Volumes: Suitable for applications requiring consistent high performance with predictable IOPS. Examples include real-time analytics platforms, large-scale databases like Oracle or SQL Server, and high-frequency trading applications.
- st1 Volumes: Cost-effective option for workloads with unpredictable bursts of IOPS followed by periods of low activity. Examples include batch processing functions, log processing systems, and scientific computing workloads.
- sc1 Volumes: Ideal for storing rarely accessed data or backups due to their very low cost. Suitable for disaster recovery backups, long-term archival data, and infrequently accessed logs.

Real-World Examples:

- Netflix: Likely utilizes a combination of gp2 and io1 volumes. gp2 volumes might handle user profiles, movie recommendations, and general website data. io1 volumes could power real-time analytics for customer behavior and content delivery optimization.
- Airbnb: Might use gp2 volumes for user databases, listing information, and search functionalities. st1 volumes could be used for storing historical booking data and user logs for compliance purposes.
- Scientific Research Institutions: May leverage sc1 volumes for storing massive datasets from experiments or simulations that are accessed infrequently for further analysis

Trade-offs in EBS Volume Selection:

Choosing an EBS volume type involves balancing several factors:

- Performance: gp2 offers a balance, io1 provides high IOPS, st1 is bursty, and sc1 has lower performance.
- Cost: sc1 is the cheapest, followed by st1, gp2, and io1 being the most expensive.
- Durability: All EBS volume types are highly durable with multiple redundancies.

Exercise 5:

Instructions to how create and attach an EBS volume using AWS CLI:

1. Select the volumes tab in the left navigation pane, your existing volumes are displayed on this page.
2. Now select volume in the upper right corner of the page
3. Select the volume type and input the size of the volume you want to create
4. Make sure that the availability zone selected is the same where the ec2 instance was previously launched
5. Once done scroll down and select on create volume
6. Available volumes have been created but are not attached to an ec2 instance
7. To attach the volume to an ec2 instance:
    - First select the volume then select actions to open the actions menu
    - Choose attach volume
    - Select the instance ID from the drop down list of ec2 instances and leave everything else as default
    - Select attach volume
    - To verify that your newly created volume is attached to your chosen rc2 instance
        - Select instances on the ec2 console from the left side navigation pane
        - Select the ec2 instance
        - Select the storage tag
        - Your newly created and attached EBS volume will be displayed in the attached state

🥳 🥳 🥳

Create a snapshot of an EBS volume and restore it to a new volume:

1. Go to Elastic Block Store/ Snapshots
2. Create Snapshots
3. The volume from which to create the snapshot.
4. Add a description to your snapshot.
5. Press add tags
    - Key
        - Name
    - Value
        - Snapshots_Volume_EC2
6. Press on create snapshot

🥳 🥳 🥳

The concept of lifecycle policies for managing EBS volumes

Cost Optimization:

- Auto-Delete After Inactivity: Delete EBS volumes not attached to instances for a certain period (e.g., 30 days).
- Snapshot and Delete: Create a snapshot of an EBS volume before deletion for potential future use.

Data Retention:

- Snapshot on Schedule: Create periodic snapshots of volumes for backup and disaster recovery purposes.

Troubleshooting EBS Volumes and Snapshots:

Common Issues:

- Volume not found: Ensure you're using the correct volume ID and the instance is in the same Availability Zone.
- Insufficient permissions: Verify your IAM user has the required permissions to create, attach, or modify volumes.
- Snapshot creation failure: Check if the volume is in use. Snapshots cannot be created of attached volumes.

Troubleshooting Tips:

- Use the aws ec2 describe-volumes command to get details on existing volumes.
- Utilize the aws ec2 describe-snapshots command to list snapshots.
- Refer to the AWS documentation for detailed troubleshooting steps for specific errors: https://docs.aws.amazon.com/

🤓 🖥️ 🤓