DALHOUSIE UNIVERSITY

CSCI 5411

Advanced Cloud Architecting

Assignment 2

Name: Jay Sanjaybhai Patel
Banner ID: B00982253

Table Of Contents

Instance Selection and Auto Scaling: Evaluate the requirements of the application and determine the type and size of EC2 instances that would be most appropriate for your needs. Consider factors like the application's CPU, memory, storage, and network performance requirements.......... 3

Multi-AZ and Multi-Region Deployment: Implement a Multi-AZ and Multi-Region deployment to ensure high availability and fault tolerance.......... 4

EBS & EFS: Plan a strategy for data storage using Amazon EBS and/or EFS..........5

Spot and Reserved Instances: Incorporate spot instances and reserved instances in your architecture.......... 6

AMI: Explain how you would use Amazon Machine Images (AMIs) to quickly deploy and replicate your application.......... 7

Cost Management: Provide a rough estimate of the costs of running this infrastructure and discuss the strategies you would use to manage these costs..........8

EC2 Placement Groups: Design an architecture where the application has a need for low-latency, high throughput communication between instances..........9

Instance Store: Design a scenario where temporary, high-IOPS storage is required and an instance store would be used.......... 9

Dedicated Hosts/Instances: Plan for scenarios where the application has to comply with strict licensing terms (BYOL) or meet dedicated hardware requirements.......... 10

EC2 Metadata and User Data: Describe how you would use EC2 metadata and user data to handle configuration tasks and pass information to instances at launch time.......... 11

Optimization and Performance: Describe how to use tools like AWS Compute Optimizer and Trusted Advisor for identifying optimal EC2 instance types and for maintaining cost efficiency.......... 12

References.......... 13

Instance Selection and Auto Scaling: Evaluate the requirements of the application and determine the type and size of EC2 instances that would be most appropriate for your needs. Consider factors like the application's CPU, memory, storage, and network performance requirements.

For Spacetech Galactic's Holodeck application, the EC2 instance selection**[1]** and auto-scaling considerations are as follows:

EC2 Instance Selection:

1. **CPU Requirement**: Real-time physics simulation and 3D rendering require **Compute-optimized (C5)** instances. For the task, **C5.2xLarge** is a suitable choice.
2. **Memory Requirement**: For handling 3D models and real-time physics engines, 8GB-16GB RAM is adequate. **C5.2xLarge** provides 16GB memory.
3. **Storage Requirement**: **EBS volumes** backed by NVMe-based SSDs will handle the data storage needs effectively.
4. **Network Performance**: Low latency and high throughput are necessary for multiplayer mode. **C5 instances** support high network bandwidth.

Auto Scaling:

- **Variable Traffic**: Auto-scaling groups should dynamically scale instances based on CPU, memory, and network metrics, adjusting automatically for spikes during weekends or new releases. This ensures resource availability and cost efficiency.

Multi-AZ and Multi-Region Deployment: Implement a Multi-AZ and Multi-Region deployment to ensure high availability and fault tolerance.

Multi-AZ Deployment:

- Ensure high availability by deploying EC2 instances across multiple Availability Zones (AZs) within a single AWS region**[2]**. If one AZ fails, the others can handle the load, minimizing downtime.
- Deploy the same application in multiple AZs. For databases like Amazon RDS, use **read replicas** in different AZs to distribute read traffic and enhance availability.
- Utilize **Elastic Load Balancer (ELB)** to distribute traffic across AZs, ensuring automatic failover and balanced performance.
- Implement Auto Scaling groups that span AZs, automatically launching new instances in healthy zones when needed.

Multi-Region Deployment:

- Provide fault tolerance and improved performance globally by replicating applications across multiple AWS regions.
- Use **RDS with Multi-Region Read Replicas** or **DynamoDB Global Tables** to keep data in sync across regions**[3]**. S3 **Cross-Region Replication** will handle static content like 3D models and assets.
- Leverage **Amazon Route 53** for latency-based or geolocation routing, ensuring users are directed to the nearest region for low-latency access. In case of a region failure, Route 53 automatically redirects traffic to healthy regions.

By utilizing both Multi-AZ and Multi-Region deployments, Spacetech Galactic can ensure seamless global user experiences, even in the event of failure, and maintain robust high availability.

EBS & EFS: Plan a strategy for data storage using Amazon EBS and/or EFS.

Amazon EBS (Elastic Block Store) [4]:

- Best suited for performance-critical components such as the real-time physics simulation engine and other low-latency workloads.
- After comparing EBS types, the **EBS Provisioned IOPS (io2)** is the best fit due to its:
  - **High throughput** and **low-latency** performance, supporting more than 64,000 IOPS for intensive workloads.
  - **Sub-millisecond latency**, which is crucial for transactional workloads and the physics engine.
  - **High durability** (99.99%), ensuring data persistence during critical computations.
- **EBS Snapshot**: Use snapshots for regular backups to ensure data recovery and durability.

Amazon EFS (Elastic File System) [5]:

- Ideal for shared storage across multiple instances, especially for multiplayer mode where shared data (like 3D assets) needs to be accessed by multiple EC2 instances across Availability Zones.
- EFS automatically scales as storage needs grow, making it perfect for handling dynamic content.
- For data that is not frequently accessed, enable **EFS Infrequent Access** to lower storage costs, as it automatically moves less-used data to a more cost-effective tier.

Combined Strategy:

- **EBS Provisioned IOPS (io2)** for low-latency, performance-critical operations (e.g., real-time physics engine, high I/O workloads).
- **EFS** is for shared, scalable storage across instances (e.g., multiplayer mode data, 3D assets), and **EFS IA** is for cost optimization.

This strategy balances high performance and cost efficiency, ensuring seamless operations for the Holodeck application.

Spot and Reserved Instances: Incorporate spot instances and reserved instances in your architecture.

Spot and Reserved Instances Strategy:

Reserved Instances [6]:

- **Discount**: Provide up to **72% savings** compared to On-Demand pricing.
- **Use Case**: Ideal for **always-on critical workloads** like the **real-time physics engine**, which needs to run 24/7.
- **Cost Efficiency**: Opt for a **3-year upfront payment** for the physics engine to optimize long-term costs.
- **Capacity Reservation**: Offers **capacity guarantees** in a specific AZ, ensuring availability for critical applications.

Spot Instances [7]:

- **Discount**: Provide up to **90% savings**.
- **Use Case**: Suitable for **non-critical workloads** like **batch processing**, **analytical reports**, and **predictive models**, where occasional interruptions are tolerable.
- **Cost Optimization**: Leverage **Spot Fleets** or **Auto Scaling** to dynamically use these for background tasks.

This approach balances cost efficiency with performance by using Reserved Instances for critical services and Spot Instances for flexible, interruptible tasks.

AMI: Explain how you would use Amazon Machine Images (AMIs) to quickly deploy and replicate your application.

Amazon Machine Images (AMIs) [8] contain the common software configuration required to launch pre-configured EC2 instances. They serve as templates that can be reused to maintain consistency across deployments.

AMIs can be utilized to quickly replicate and deploy EC2 infrastructure for the Holodeck application, ensuring that all instances run the same tech stack.

1. Start by launching a base EC2 instance and installing the necessary software for the real-time physics engine and interactive 3D model display.
2. Once the EC2 instance is fully configured and tested, deploy it.
3. Create a custom AMI from the configured instance by navigating to “Images and templates” and selecting the option to create an AMI.
4. After successfully creating the AMI, you can launch new EC2 instances from this AMI. Each new instance will inherit the configurations and software from the original base instance.

Every time you create a new instance from the AMI, all the configurations will be set during the initial setup, ensuring uniformity.

AMIs can be integrated with Auto Scaling Groups to dynamically launch instances based on user demand, facilitating rapid scaling during peak usage times.

You can replicate the AMI across multiple AWS regions for enhanced availability and reduced latency for global users.

Cost Management: Provide a rough estimate of the costs of running this infrastructure and discuss the strategies you would use to manage these costs.

Infrastructure Components:

- **EC2 Instances**: C5.2xLarge shared instances for compute-optimized performance.
- **EBS Storage**: io2 storage device for high-performance, low-latency requirements.
- **Monitoring**: AWS CloudWatch for resource monitoring and management.

Estimated Costs:

- The estimated annual cost of running this infrastructure is approximately **$68,000**.

Cost Management Strategies:

1. **Reserved Instances**:
   - Purchase reserved instances for 3 years to secure up to 72% savings compared to on-demand pricing.
2. **Spot Instances**:
   - Utilize spot instances for non-critical workloads, achieving savings of up to 90%.
3. **Auto Scaling**:
   - Implement auto scaling strategies to dynamically scale down instances during off-peak hours, optimizing costs based on demand.
4. **EBS Storage Optimization**:
   - Implement storage lifecycle policies to automatically move infrequently accessed data to cheaper storage tiers, reducing overall storage costs.

By leveraging these strategies, Spacetech Galactic can effectively manage infrastructure costs while maintaining high performance and availability for the Holodeck application.

EC2 Placement Groups: Design an architecture where the application has a need for low-latency, high throughput communication between instances.

- By creating a **cluster placement group**, we can position instances closer together within a **single Availability Zone**. This reduces network latency and enhances network performance, as AWS will try to place the instances as close as possible within the same rack or network switch.
- Opt for **C5.2xLarge** instances to provide the necessary computing power for the real-time physics engine and interactive 3D models.
- Enable **Elastic Network Adapter (ENA) [9]** for improved network performance, ensuring lower latency and higher throughput during communication between instances.

This approach ensures optimal performance for the application by maximizing network efficiency and minimizing latency through strategic placement of instances.

Instance Store: Design a scenario where temporary, high-IOPS storage is required and an instance store would be used.

- In the Spacetech Holodeck application, there is a need for temporary block-level storage to process real-time data for the physics engine. This data is continuously changing and requires high Input/Output Operations Per Second (IOPS) for optimal performance.
- The application must process large volumes of VR data quickly, performing compute-intensive tasks to ensure a smooth user experience.
- The application manages stateful interactions that do not require long-term data persistence, making temporary storage suitable.
- Utilize instance store volumes for temporary caching of frequently accessed data during VR sessions, allowing rapid read and write operations essential for real-time interactions **[10]**.

Dedicated Hosts/Instances: Plan for scenarios where the application has to comply with strict licensing terms (BYOL) or meet dedicated hardware requirements.

- Using AWS EC2 Dedicated Hosts **[11]** provides direct access to the underlying hardware, ensuring high performance and isolation from other AWS accounts.
- Dedicated Hosts allow Holodeck to utilize existing Bring Your Own License (BYOL) agreements, such as server software or SQL licenses that are tied to VMs, sockets, or physical cores. This not only facilitates compliance but also leads to significant cost savings on licensing.
- By deploying instances on Dedicated Hosts, Holodeck can leverage the specific hardware requirements needed for the physics engine, ensuring that resource-intensive tasks are handled efficiently without interference from other workloads.

EC2 Metadata and User Data: Describe how you would use EC2 metadata and user data to handle configuration tasks and pass information to instances at launch time.

EC2 Metadata[12]

- EC2 Metadata provides data about the instance itself, including information such as the hostname, events, security groups, and other attributes that can be used to configure running instances.
- You can retrieve instance metadata by making an HTTP request.
- For the Holodeck application, querying EC2 Metadata allows the application to adapt based on its environment. For instance, it can adjust configurations or resource allocations based on the instance type or security group settings.

EC2 User data [13]

- User Data consists of parameters or scripts that are executed when an instance is launched. This allows for automated setup and configuration tasks to be performed at the time of the instance's first boot.
- The User Data script is executed only once during the initial boot of the instance.
- In the Holodeck application, User Data can be used to install the necessary software and configure the environment automatically at launch. For example, the User Data script can install required packages, set environment variables, and start the application.

By effectively utilizing EC2 Metadata and User Data, you can automate instance configuration and streamline deployment processes. This enhances the adaptability of applications while ensuring consistency across environments, reducing manual intervention, and improving operational efficiency.

Optimization and Performance: Describe how to use tools like AWS Compute Optimizer and Trusted Advisor for identifying optimal EC2 instance types and for maintaining cost efficiency.

1. AWS Compute Optimizer [14]:

- AWS Compute Optimizer provides recommendations for optimizing EC2 instance types based on an analysis of your instances, auto-scaling groups, and workload patterns.
- By examining historical utilization metrics such as CPU, memory, and network performance, it suggests instance types that not only meet the application's requirements but also help reduce costs.

2. AWS Trusted Advisor [15]:

- AWS Trusted Advisor identifies unused or underutilized resources to help lower costs.
- It analyzes the usage and configuration of your AWS environment, including high network or CPU utilization, and offers recommendations that enhance application performance, speed, and responsiveness.

By effectively using AWS Compute Optimizer and Trusted Advisor, you can optimize instance types for performance and ensure cost efficiency in your AWS environment. This approach enhances application performance while minimizing unnecessary expenses, making it ideal for applications like Holodeck that require high performance and responsiveness.

References

[1] "Amazon EC2 Instance Types," aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html. [Accessed: Oct 22nd, 2024].

[2] "Multi-AZ Deployment for Amazon EC2 Instances," GitHub, [Online]. Available: https://github.com/rohitmourya40901/Multi-AZ-Deployment-for-Amazon-EC2-Instances. [Accessed: Oct 22nd, 2024].

[3] "Creating a Multi-Region Application with AWS Services: Part 1 - Compute and Security," aws.amazon.com [Online]. Available: https://aws.amazon.com/blogs/architecture/creating-a-multi-region-application-with-aws-services-part-1-compute-and-security/. [Accessed: Oct 22nd, 2024].

[4] "Amazon Elastic Block Store (EBS)," aws.amazon.com [Online]. Available: https://aws.amazon.com/ebs/. [Accessed: Oct 22nd, 2024].

[5] "Amazon Elastic File System (EFS)," aws.amazon.com [Online]. Available: https://aws.amazon.com/efs/. [Accessed: Oct 22nd, 2024].

[6] "Amazon EC2 Pricing - Reserved Instances," aws.amazon.com [Online]. Available: https://aws.amazon.com/ec2/pricing/reserved-instances/. [Accessed: Oct 22nd, 2024].

[7] "Amazon EC2 Spot Instances," aws.amazon.com [Online]. Available: https://aws.amazon.com/ec2/spot/. [Accessed: Oct 22nd, 2024].

[8] "Amazon Machine Images (AMIs)," aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html. [Accessed: Oct 22nd, 2024].

[9] "Elastic Network Adapter: High-Performance Network Interface for Amazon EC2," aws.amazon.com [Online]. Available: https://aws.amazon.com/blogs/aws/elastic-network-adapter-high-performance-network-interface-for-amazon-ec2/. [Accessed: Oct 22nd, 2024].

[10] "Amazon EC2 Instance Storage," aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html. [Accessed: Oct 22nd, 2024].

[11] "Dedicated Hosts Overview," aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-hosts-overview.html. [Accessed: Oct 22nd, 2024].

[12] "Instance Metadata and User Data," aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instancedata-data-retrieval.html. [Accessed: Oct 22nd, 2024].

[13] “User Data and Metadata,” aws.amazon.com [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html. [Accessed: Oct 22nd, 2024].

[14] “AWS Compute Optimizer,” aws.amazon.com [Online]. Available: https://aws.amazon.com/compute-optimizer/. [Accessed: Oct 22nd, 2024].

[15] “AWS Trusted Advisor,” aws.amazon.com [Online]. Available: https://aws.amazon.com/premiumsupport/technology/trusted-advisor/. [Accessed: Oct 22nd, 2024].