

# Design and Implementation of a Scalable Web Application Using AWS Cloud Infrastructure

Abhay Kumar Verma

*Department of Computer Science and Engineering, Sagar Institute of Research & Technology, Bhopal (M.P.), India*

**Abstract**—Cloud computing provides on-demand computing, storage, networking, and database services that can be configured according to application workload. This paper presents the design and implementation of a scalable web application deployed on Amazon Web Services (AWS). The proposed architecture integrates Amazon Elastic Compute Cloud (EC2) for application hosting, Amazon Simple Storage Service (S3) for static content, Amazon Relational Database Service (RDS) for structured data, Elastic Load Balancing for request distribution, Auto Scaling for resource adjustment, Identity and Access Management (IAM) for access control, and Amazon CloudWatch for monitoring. The implementation follows a practical deployment methodology in which application traffic is distributed across multiple computing instances while storage and database responsibilities are separated from application processing. The reported evaluation compares the application before and after load balancing. Response time decreased from 2.4 seconds to 1.2 seconds, CPU utilization changed from approximately 80–90% to 40–60%, and the reported request-handling capacity increased from 50–70 users to more than 200 users. The results indicate that distributed cloud infrastructure can improve application responsiveness, resource utilization, and availability, although cost, configuration complexity, and dependency on network connectivity remain important considerations.

**Index Terms**—AWS, cloud computing, EC2, elastic load balancing, scalable web application, S3, RDS.

## I. INTRODUCTION

Web applications are increasingly required to support changing user demand, continuous availability, and rapid deployment. Traditional on-premise hosting generally depends on fixed hardware capacity and requires organizations to manage servers, storage,

networking, power, cooling, upgrades, and security. When demand increases beyond the available capacity, applications may experience increased response time, resource saturation, or service interruption. Cloud computing addresses these challenges by providing computing resources through the internet and allowing infrastructure to be provisioned according to workload requirements.

Amazon Web Services provides a broad collection of services for application hosting, storage, databases, networking, security, and monitoring. This work focuses on a practical architecture that combines these services to deploy a web application in a scalable environment. The central objective is to demonstrate how workload distribution, managed storage, database services, access control, and monitoring can be integrated into one deployment model.

The contributions of this work are: (1) design of an AWS-based architecture for scalable web hosting; (2) implementation of EC2, S3, RDS, Elastic Load Balancing, Auto Scaling, IAM, and CloudWatch; and (3) performance comparison using response time, CPU utilization, request-handling capacity, and availability.

## II. RELATED WORK

Prior research describes cloud computing as a utility-oriented model that provides on-demand resources and reduces the need for large upfront infrastructure investment. Armbrust et al. discussed the ability of cloud platforms to provision resources dynamically according to workload. Buyya et al. described cloud computing as the delivery of computing capabilities as services. Zhang et al. emphasized workload

distribution and dynamic resource allocation as important mechanisms for scalable cloud systems.

Research on distributed systems also highlights the role of redundancy, fault tolerance, and workload distribution. Fox et al. examined the role of cloud infrastructure in supporting reliable internet services, while Bernstein et al. discussed interoperability and fault tolerance in cloud environments. Marston et al. analyzed the economic implications of cloud adoption. These studies provide the conceptual foundation for the architecture implemented in this work.

### III. PROPOSED SYSTEM ARCHITECTURE

The proposed system uses a layered architecture. Incoming requests reach an Elastic Load Balancer through the internet. The load balancer distributes requests among EC2 instances that host the web application. An Auto Scaling Group manages the number of instances according to workload conditions. Amazon RDS stores structured application data, while Amazon S3 stores static files such as images, documents, and other media. IAM defines permissions for users and services. CloudWatch collects operational metrics including CPU utilization, network activity, and instance health.

The separation of application processing, structured data, and static content reduces the responsibility placed on a single server. Load balancing prevents all requests from being directed to one instance, while multiple instances provide an opportunity for continued operation when one instance becomes unavailable. The architecture also supports controlled access through security groups and IAM policies.

The workflow is as follows: (1) a user sends a request through the internet; (2) the load balancer selects a healthy application instance; (3) the EC2 instance processes the request; (4) structured data is retrieved from RDS when required; (5) static resources are served from S3; and (6) CloudWatch records system metrics for analysis and operational monitoring.

### IV. IMPLEMENTATION METHODOLOGY

The implementation followed a practical, deployment-oriented methodology. First, the AWS environment and network configuration were

prepared. EC2 instances were created and configured as virtual application servers. Required server software and application files were installed, and security group rules were defined to control inbound and outbound communication.

Next, an S3 bucket was configured for static files and an RDS database instance was prepared for structured application data. The application was then connected to the database and deployed on the EC2 environment. An Elastic Load Balancer was configured as the entry point for requests, and an Auto Scaling Group was used to manage application instances. IAM roles and permissions were applied to limit access to cloud resources.

Finally, CloudWatch was used to observe CPU utilization, network activity, and instance health. The system was evaluated under varying request loads, and the observed behavior was compared with a configuration without load balancing. The thesis implementation also included VPC configuration, security groups, Elastic IP configuration, and monitoring dashboards.

### V. AWS SERVICES USED

| AWS Service            | Purpose                                         |
|------------------------|-------------------------------------------------|
| EC2                    | Hosts and executes the web application.         |
| S3                     | Stores static files and application assets.     |
| RDS                    | Manages structured relational application data. |
| Elastic Load Balancing | Distributes incoming requests across instances. |
| Auto Scaling           | Adjusts the number of application instances.    |
| IAM                    | Controls identities, roles, and permissions.    |
| CloudWatch             | Monitors CPU, network, and instance health.     |
| VPC/Security Groups    | Provides network isolation and traffic control. |

## VI. RESULTS AND DISCUSSION

The implemented system was evaluated using response time, CPU utilization, request-handling capacity, and availability. The comparison reported in the thesis is summarized in Table I.

Table I. Performance comparison reported in the implementation study.

| Parameter            | Without Load Balancer | With Load Balancer |
|----------------------|-----------------------|--------------------|
| Response time        | 2.4 seconds           | 1.2 seconds        |
| CPU utilization      | 80–90%                | 40–60%             |
| Requests handled     | 50–70 users           | 200+ users         |
| System availability  | Moderate              | High               |
| Server overload risk | High                  | Low                |

The reported response time decreased by approximately 50% after load balancing. CPU utilization also became more balanced, indicating that the workload was distributed across multiple instances rather than concentrated on a single server. The reported request-handling capacity increased to more than 200 users. These observations support the practical value of distributing traffic and separating application services.

CloudWatch monitoring showed that instance status checks and system status checks passed successfully during the evaluation. CPU utilization increased when repeated requests were directed toward a limited number of resources, demonstrating the importance of capacity planning and scaling policies. The results should be interpreted as deployment-specific measurements rather than universal performance guarantees because workload pattern, instance type, application complexity, region, and network conditions influence performance.

## VII. SECURITY CONSIDERATIONS

Security was addressed through IAM permissions, security groups, VPC configuration, and the planned use of encryption and HTTPS. IAM policies restrict

access to authorized users and services. Security groups control network traffic to the application instances and database. VPC configuration provides network isolation and controlled routing. Encryption can protect stored data in S3 and RDS, while HTTPS protects data transmitted between users and the application. CloudWatch monitoring and alerts can assist in identifying unusual activity and operational failures.

## VIII. LIMITATIONS

The implementation has several limitations. Cloud resources generate operational costs, particularly when multiple instances, managed databases, load balancers, and monitoring services are active. The system also depends on internet connectivity and correct configuration of networking, permissions, scaling policies, and security rules. The evaluation is based on the workload and environment used in the project; therefore, larger-scale production behavior requires additional testing. The architecture is primarily designed around AWS services, which may increase platform dependency and migration effort.

## IX. CONCLUSION AND FUTURE WORK

This paper presented the design and implementation of a scalable web application using AWS cloud infrastructure. The integration of EC2, S3, RDS, Elastic Load Balancing, Auto Scaling, IAM, VPC, and CloudWatch provided a structured environment for application hosting, data management, traffic distribution, security, and monitoring. The reported comparison showed improved response time, more balanced CPU utilization, greater request-handling capacity, and higher availability after introducing distributed cloud infrastructure.

Future work may include automated CI/CD pipelines, containerized deployment, infrastructure as code, database read replicas, content delivery through a CDN, stronger observability, automated security scanning, multi-region deployment, and more extensive benchmarking with standardized workloads.

REFERENCES

- [1] M. Armbrust et al., “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
- [2] R. Buyya, C. S. Yeo, and S. Venugopal, “Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities,” in *Proc. 10th IEEE Int. Conf. on High Performance Computing and Communications*, 2008, pp. 5–13.
- [3] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: State-of-the-art and research challenges,” *Journal of Internet Services and Applications*, vol. 1, pp. 7–18, 2010.
- [4] A. Fox et al., “Above the clouds: A Berkeley view of cloud computing,” *University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28*, 2009.
- [5] P. Mell and T. Grance, “The NIST definition of cloud computing,” *NIST Special Publication 800-145*, 2011.
- [6] M. Marston et al., “Cloud computing—The business perspective,” *Decision Support Systems*, vol. 51, no. 1, pp. 176–189, 2011.
- [7] D. Bernstein, E. Ludvigson, K. K. K. Ramakrishnan, and A. M. S. “Blueprint for the cloud,” in *Proc. 6th Int. Conf. on Internet and Web Applications and Services*, 2009.
- [8] Amazon Web Services, “AWS documentation: Amazon EC2, Amazon S3, Amazon RDS, Elastic Load Balancing, Auto Scaling, IAM, and CloudWatch,” AWS technical documentation.