Load Balanced OwnCloud Service Using AWS

Mouna SKANDER
Bachelor in Computer Science,
Hof University of Applied Sciences
Bavaria, Germany
mouna.skander@hof-university.de

Tuana GÜLER
Bachelor in Computer Science
Hof University of Applied Sciences
Bavaria, Germany
tuana.gueler@hof-university.de

This paper explores the implementation of a fault-tolerant, flexible, and highly available load-balanced OwnCloud service using Amazon Web Service (AWS). The focus is on meeting the requirement of a strong and secure system to store and access data for a regional second-hand shop. Overall, this paper demonstrates the effectiveness of AWS in building a tailored solution for a specific use case, meeting the needs of a dynamic and secure cloud computing system.

I. INTRODUCTION

Thanks to the advancement in technology at an exponential rate, the need for strong and secure systems to store and access data has been increasing rapidly. Cloud computing is one of the possible answers to meet this need. In addition, cloud computing provides organizations and customers with high availability from all around the world. Many cloud computing services offer various levels of control within their systems. This paper will describe the fault-tolerant, flexible, and highly available load-balanced OwnCloud service with a user story and discuss the reasons why Amazon Web Services (AWS) is used to build such a system.

The user scenario we are considering is about a regional second-hand shop that needs an active online platform to host continuous sales and real-time auctions at varying times during the day. The marketplace is focused on serving a specific region where the second-hand shop and most of the customers are located. Our customer only asks for a small proportion of the architecture as an example, but not a completed website. This scenario
suggests a host that requires various properties, such as being able to handle high traffic during peak auction times and staying efficient during slower periods. Therefore, a system that is capable of handling these fluctuating demands, and also flexible, fault-tolerant, and secure is needed profoundly.

We chose AWS as the platform for this project because AWS offers an extensive amount and variety of tools and services that can be utilized to customize and fit different use cases. AWS provides an exhaustive set of scalable and reliable products, bringing about a highly tailored system. AWS puts up security for risk-sensitive organizations due to its comprehensive tools and services. Besides, its infrastructure promises fault-tolerant and highly available cloud-computing systems.

In the following sections, the technical aspects of the system will be discussed in a more thorough manner. This will include a detailed description of our system in separate parts such as Web Server, Database Server, File System Server, and how we met the needs of the user-case with our solution using AWS.

II. USER STORY:

As a client, I want to have a highly available architecture for my regional second-hand/ auctionary website that can handle a large amount of traffic and provide a seamless experience to the users. The architecture should ensure high availability, low latency during high traffic times, efficient storage and retrieval of a vast collection of images and videos, and provide reliable data storage.

Acceptance Criteria:

A. User story 1: The Web Server

As a web visitor, I want the website to be available and responsive at all times, even during peak periods. I expect the website to load quickly and provide a smooth browsing experience. Plus, it has to scale seamlessly to handle increased traffic without any disruptions.

B. User story 2: The Database Server

As an administrator, I want the database to store all the second-hand pieces, auction information, and users' data to be highly reliable and available. And I expect the database to seamlessly handle node failures and maintain data consistency to ensure that all data remains intact and accessible to users at all times.

C. *User Story 3: The File System Server*

As a user, I want the website to load media assets such as images and videos quickly and without any glitches when I shop or participate in an auction. Additionally, I expect that a large volume of media will be stored securely and efficiently, ensuring a seamless viewing experience of items on the website.

D. *User story 4: Security*

As a user, I want my personal information and financial transactions to be secured while buying or bidding online. And I expect that the website will have robust security measures in place, including encrypted connections, secure storage of sensitive data, and protection against unauthorized access or breaches.

III. THE WEB SERVER

A. *Introduction and General Idea*

Our client, the owner of the second-hand shop which operates sales and auctions online, needs a small-scale working architecture for the website. The website focuses on one region where the store and its customers are. In addition, this system has a demand for an uninterrupted, high-speed, highly-available service to provide its customer with a smooth experience during auctioning and general browsing times.

The Web Server is set up considering the unique demands of our platform, to be secure, efficient, and robust. Using AWS to design this system, gives us a flexible and scalable environment.

In this design, Apache and OwnCloud are installed in three separate private EC2 Instances to function like a web server and file hosting service, handling client requests, user data, and information about items to be sold or auctioned.

B. *Design and Architecture*

Recognizing this second-hand sale and auctioning website's needs, accessibility, uptime, and efficient data exchange is crucial. That is the reason why the architecture we developed includes a dedicated VPC for the Web Server, three worker nodes to function as a web, file, and database hosting server, and an ELB Application Load Balancer to control incoming traffic.

1) *Web Server VPC*

The web Server is hosted in its own Virtual Private Cloud within AWS, enabling network isolation to enhance security through network segmentation. This allows for control over the network configuration and traffic flow, shielding the system from external threats.

The Web Server VPC is peered with both database VPC and file system VPCs. This peering connection ensures seamless communication between the web server nodes and the other servers, all while maintaining a high level of security.

The VPC hosts a public subnet for a jumper server, and three private subnets each hosting a web server node. These private subnets are strategically placed across different Availability zones, to promote high availability and fault tolerance, ensuring our customer that the server remains unaffected even during maintenance periods or in the unlikely event of a zone blackout.

2) *EC2 Instances*

EC2 instances are separated into two categories: a Jumper node and the Web Server nodes.

a) *Jumper Node*

A pivot for secure administrative access, the jump server (or jumper node), is an EC2 instance located in the public subnet. It acts as a secure access point for the administrative tasks required to manage the web server nodes in the private subnets.

Because of the limited requirement for computing power in the jumper node and budget constraints, we have chosen the "t2.micro" instance type for this node. It offers an excellent balance of modest computing capacity at a minimal cost. And an EBS volume of 8 GB is sufficient for handling the operation system and related software.

b) *The Web Server Nodes*

Our system contains three nodes in different private subnets to work as a web server, reducing latency and improving the general user experience. The nodes operate a reliable open-source web server software, Apache, to ensure consistent service delivery.

OwnCloud, a storage and file-sharing system, is configured in these nodes and connected to GlusterFs and the database nodes for efficient file management and supporting the website's needs for quick information.

In the context of affordability, these nodes are t2.micro instance type and have 20GB of EBS volume to picture a general and small-scale architecture for our customer. In the future, it can be expanded easily to adapt to the platform's growth.

3) *ELB Load Balancer*

The Elastic Load Balancer (ELB) is incorporated into our architecture, which is a pivotal element of the AWS Ecosystem. The ELB oversees the distribution of incoming traffic across the web server nodes. Balancing the load ensures a seamless user experience, maintaining the site's responsiveness, especially and most important during peak auction times.

IV. DATABASE SERVER:

A. Introduction and General Idea

The implementation of a highly available and scalable database solution is paramount for our customer to ensure uninterrupted operations and meet the demands of a growing user base. In this context, the MariaDB Galeria Cluster combined with the power of AWS services presents an ideal solution.

The core objective of MariaDB Galeria Cluster is to deliver a resilient and reliable database infrastructure capable of handling high volumes of traffic. By leveraging AWS services, the cluster will be designed to accommodate increasing data loads and user demands.And it will provide efficient data storage and retrieval, ensuring seamless operations even during peak usage periods.

B. Designing a Secure Infrastructure

1) Virtual Private Network

The VPC will serve as an isolated virtual network for the database cluster, ensuring secure communication between resources. And to maximize the availability three availability zones will be used within the region. Each one will have both public and private subnets to facilitate secure access and isolate database instances from the public internet.

2) Subnets

Subnets are going to provide a logical separation of resources. Three public subnets will be created, each associated with a corresponding availability zone. These subnets will be used for resources that require public internet access, such as the jumper instance. Also, three private subnets will be established in each availability zone. Private subnets ensure that the database instances remain inaccessible from the public internet while allowing communication within the cluster.

3) NAT Gateway

Nat Gateway will be implemented to enable outbound internet connectivity for resources in the private subnets. It is associated with one of the public subnets, and will act as an intermediary for outgoing traffic from private instances. This setup will allow instances to communicate with external services and retrieve necessary updates while maintaining their security and isolation from the internet.

4) Security Group

Security group acts as a virtual firewall for EC2 instances, controlling inbound and outbound traffic. By configuring security groups to allow access only to necessary ports (such as SSH and MySQL).

5) Key pair

A key pair will be generated to establish secure access to the EC2 instances within the cluster. The key pair will consist of a public key for encrypting data and a private key for decrypting the data. This key pair will be associated with the instance, allowing authorized personnel to securely access and manage the resources.

C. EC2 Instances:

The Cluster will include multiple EC2 instances deployed in the VPC, serving various roles.

1) Jump Server:

Deploying an EC2 instance in a public subnet as a jumper server enables secure access to resources in the private subnets.It acts as an intermediary, allowing administrators to connect securely to the private instances where the database is stored without exposing them to the internet.

2) MariaDB Galera Cluster Nodes:

Implementing a MariaDB Galera Cluster across the private subnets facilitates database replication and ensures high availability. With a primary node that is going to mount the cluster, and two secondary nodes, the cluster provides fault tolerance and load balancing capabilities, allowing for uninterrupted database operations.

V. FILE SYSTEM:

A. Introduction and General Idea

A robust online auctioning and selling system requires flexibility, high availability, and fault tolerance for the file storage system to function correctly. To support these needs like high volume and high availability, a unique File System Server is designed by using AWS because of its reliable and scalable cloud solutions.

In this system, the volume created in GlusterFS nodes accommodates storage for information about each item listed to be sold, including descriptions and images. This data is essential for buyers to navigate the website, participate in auctions, or make purchases from the shop.

B. Design and Architecture

The foundations of the design of the FSS are high availability and resilience. The architecture consists of a Virtual Private Cloud (VPC), which includes an intricate subnet layout, for the file system, and Elastic Compute Cloud (EC2) Instances to host GlusterFS.

1) File System VPC

The FSS is built in a dedicated VPC whose CIDR block is 10.2.0.0/16. The File System VPC contains one public and three private subnets in different availability zones, which allows fault tolerance and high availability in case of the failure of one zone.

A NAT gateway within the File System VPC, allows private subnets to access the internet for package installations and updates when needed. One NAT gateway, instead of three NAT gateways per subnet, is sufficient for these kinds of needs and also helps to reduce the cost.

Because File System Server and Web Server have different VPCs, a VPC peering connection was crucial between those VPCs. Otherwise, the Web Server would not be able to communicate with File System Server.

A security group for the VPC is configured to control inbound and outbound traffic and all necessary communication through specific ports. This security group permits communication between GlusterFS nodes. In addition, it lets the Web Server nodes and OwnCloud connect to the GlusterFs nodes so that OwnCloud can use the volume as storage.

2) *EC2 Instances*

The EC2 Instances are divided into two categories: the jumper node and the GlusterFS nodes.

a) *Jumper Node*

The jumper node, the only node launched in the public subnet, acts as a bastion host. Accessing the private GlusterFS nodes is only possible with this node providing a secure point to enter.

Its instance type is t2.micro, which is cost-effective and practical. Since our customer wants small-scale architecture and we have a limited budget, t2.micro is a proper choice for a starting point. If, in the future, our customer's needs grow, thanks to the flexibility of AWS, it can be easily changed into a more powerful instance type.

Amazon Elastic Block Store (EBS) is an on-point choice as the root device of the Jumper node because of its highly available and high-performance block-level storage.

It has a default 8GB of storage, which is enough for a jumper node in a small-scale architecture. Besides, the Jumper node is not used to store data but controls the traffic for private instances. Yet again, if it is needed, the volume can be increased easily.

b) *GlusterFS Node*

To fulfill our customer's needs, a small-scale working architecture for the second-hand sales and auction website is required. The system, which we designed to meet these needs, uses GlusterFS as a distributed file system so that it can be scalable and highly performing. Additionally, considering our budget and performance requirements, the instance type is chosen as t2.micro, providing enough computing power and cost-effectiveness.

TABLE I.

Node IP	**Brick Name**	**Volume Name**
10.2.1.20	brick_a	gfs_v_r5
10.2.3.36	brick_b	gfs_v_r5
10.2.5.12	brick_c	gfs_v_r5

Fig. 1 Nodes and bricks within them forming the volume

TABLE II.

brick_a	**brick_b**	**brick_c**
A1	A2	P(A1 XOR A2)
B1	P(B1 XOR B2)	B2
P(C1 XOR C2)	C1	C2

Fig. 2 Division of the volume into each brick

Each node utilizes an EBS of 20GB as the root device because of its high performance and durability. 20 GB volume is enough for a small scale, it can be increased in the future owing to the flexibility of AWS.

GlusterFS system consists of three private EC2 Instances in different private subnets. Each instance is configured to include a GlusterFS server and a brick which will form the volume. (Fig. 1)

A RAID-5-like volume configuration is implemented in the GlusterFS nodes for improved performance and redundancy. In this design, parity, and data information is striped across all bricks by setting up the volume with 'disperse 3 redundancy 1', allowing the system to not lose any data when one of the bricks is failed. Since performance and fault tolerance are equally important to store the item information for sales and auctions, this setup similar to RAID-5 is an excellent choice. (Fig. 2)

VI. THE SCHEMA OF THE ARCHITECTURE

In the schema of the architecture, all three servers which create the general system can be seen. (Fig. 3)

The system exists in one region, divided into three VPCs for each server. Each VPC consists of three availability zones to host public and private subnets. In addition to that, the Web Server VPC is peered with the

File System Server and the Database Server VPCs to communicate with them.

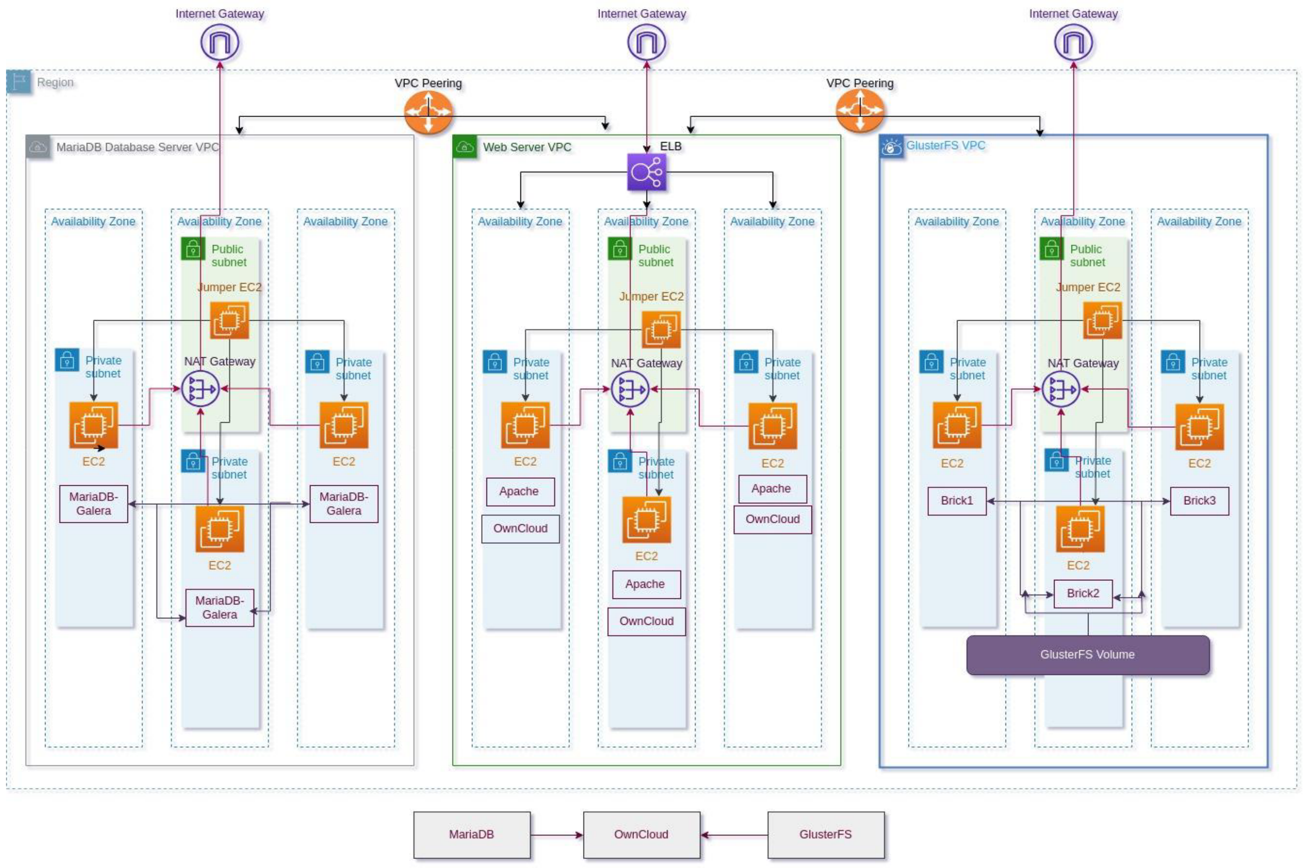

Fig. 3 General architecture of the Web Server, Database Server, and File System Server

VII. CONCLUSION:

Establishing a secure and highly available infrastructure is important for our customer, aiming to safeguard their data and ensure uninterrupted access. This paper has represented a comprehensive architectural design utilizing AWS services such as VPC with multiple availability zones, public and private subnets, EC2 Instances, ELB (Elastic Load Balancer), and Gluster File System.

The Utilization of VPC with multiple availability zones provides fault tolerance and high availability, minimizing the risk of service disruption. The segregation of resources into public and private subnets enhances security and protects sensitive data. And configuring security groups with strict inbound and outbound rules allows one to control network traffic and prevent unauthorized access.

Deploying EC2 instances as web servers, along with ELB, allows for load balancing and ensures that the website remains highly available, even during peak traffic periods. The inclusion of the Gluster File System facilitates distributed storage enabling efficient data management and redundancy.

Furthermore, with MariaDB Galeria Cluster, it ensures database replication, providing data consistency, high availability, and uninterrupted service in the event of a failure in the primary node.

By leveraging these AWS services and implementing a comprehensive infrastructure, our customer can provide its users with a secure and reliable platform. Additionally, with AWS he can benefit from scalability, in case he wants to grow his business.

In summary, the adoption of a robust and highly available infrastructure, combined with the utilization of services such as VPC, security groups, EC2 instances, MariaDB Galeria Cluster, ELB and Gluster File System, empowers our customer to offer a seamless and secure experience to its users while ensuring scalability and reliability through AWS's cloud services.