**Jaykumar Arvindbhai Suthar**

Project Report

**“Designing and implementing a secure web application architecture using AWS”**

Table of Contents

[**Introduction** 3](#_Toc152150295)

[**Problem Statement** 4](#_Toc152150296)

[**Scope** 5](#_Toc152150297)

[**Significant of Study** 6](#_Toc152150298)

[**Setup an AWS Account** 8](#_Toc152150299)

[**Create Virtual Private Cloud (VPC)** 11](#_Toc152150300)

[**Launch Website** 15](#_Toc152150301)

[**Configure Load Balancing** 16](#_Toc152150302)

[**Implement Network Security** 18](#_Toc152150303)

[**Enable HTTPS with ACM** 19](#_Toc152150304)

[**Implement Access Control** 23](#_Toc152150305)

[**Data Encryption** 29](#_Toc152150306)

[**Monitoring and Logging** 30](#_Toc152150307)

[**Backup and Disaster Recovery** 35](#_Toc152150308)

[**Continuous Security Testing** 37](#_Toc152150309)

[**Structure for Compliance** 38](#_Toc152150310)

[**Reference** 40](#_Toc152150311)

# **Introduction**

In today’s dynamic, ever-changing digital world, it is of the utmost importance to secure our web applications. A lot of sensitive data flows through a website, and it becomes our responsibility to protect this user data. For this, we need to take proactive steps. In this case, we would be using a combination of various security-related services offered by Amazon Web Services (AWS) to demonstrate how to properly secure a website. We would be using various offerings by AWS, like the EC2 instances, the Elastic Load Balancer, CloudWatch, Key Management Services, and Identity and Access Management. (AWS, 2019)

In this project, we shall demonstrate how to use EC2 to build scalable computing resources and then use the load balancer to make sure there is always high availability and correct distribution of incoming traffic to the webpage, making it more resilient. CloudWatch is another very important part of this puzzle as it helps us by providing real-time monitoring of the website as well as raising alerts and alarms when something goes wrong so that we can respond immediately to a threat.

Another extremely important facet of security is encryption, and, in this case, that would be taken care of by KMS. It provides us with robust security for data at rest as well as in transit by giving cryptographic keys, making sure the data is intact and integrity is not compromised. Another important factor is IAM, which acts as the gatekeeper who controls the access to AWS resources, making sure those who need access only get the access and not anyone else.

In this interconnected world, another invaluable piece of the security landscape is the VPN. We use the AWS VPN for this purpose. It provides us with secure communication channels, making data transmission safer and keeping the network private and out of sight of prying eyes. We can also create subnets and security groups using the VPN, enhancing security exponentially. (AWS, 2023)

As the world we live in today keeps getting more and more interconnected and evolves further, this interplay of various AWS tools can be a very powerful way to secure our web presence. Later in this document, we shall see exactly how to use all these tools in an efficient manner to secure a website in the best way possible.

# **Problem Statement**

Our project focuses on enhancing web application security using AWS (Amazon Web Services) security services. This approach streamlines security by integrating AWS tools, including encryption, access management, monitoring, and threat detection. Our goal is to bolster security without the typical administrative complexities, making it easier to protect user data and comply with regulations. Ultimately, we aim to leverage AWS's expertise in cloud security to ensure data confidentiality, integrity, and availability while minimizing resource-intensive manual security efforts.

# **Scope**

The project encompasses a wide range of aspects related to the design and implementation of a secure web application architecture on the Amazon Web Services (AWS) platform as follows:

* Network security entails building and configuring a secure Virtual Private Cloud (VPC), Network Access Control Lists (NACLs), Security Groups, and the deployment of security services to protect against potential external assaults.
* IAM will be configured to enable precise access control and role management within the Amazon Web Services (AWS) environment.
* Data security refers to the implementation of measures such as data encryption during storage and transmission, as well as the establishment of access control mechanisms for data kept within the Amazon S3 platform.
* The primary emphasis of the project will be on application security, specifically targeting the application layer. This will involve implementing a Web Application Firewall and adhering to secure coding principles.
* The establishment of real-time monitoring, threat identification, and incident response protocols will be undertaken.
* The implementation of automated backup systems and the development of a comprehensive disaster recovery plan will be incorporated.
* The consideration of adherence to industry-specific compliance standards and governance regulations will be given in relation to compliance and governance.
* The project will address the aspects of scalability and performance improvement in relation to web applications.
* The topic of cost management will be discussed to ensure the efficient monitoring and management of costs in operational activities.

# **Significant of Study**

A crucial and current problem in today's digital environment is designing and executing a secure web application architecture utilizing AWS. For several reasons, this study is significant.

**1 Increasing Cybersecurity Concerns:** Organizations must put strong security measures in place to safeguard sensitive data and keep consumers' confidence in the face of cyberattacks' rising frequency and complexity. To overcome these issues, it is crucial to create a secure web application architecture.

**2. Amazon Web Services (AWS)** is one of the most extensively used cloud platforms globally. It is regarded as a leading cloud provider. AWS is a popular platform for hosting web applications, thus it's critical to comprehend how to safeguard these platforms efficiently.

**3. Strict data privacy** laws like the GDPR and CCPA impose significant penalty for data breaches. To maintain compliance with these rules and avert legal repercussions, a secure web application design is necessary.

**4. Business Reputation:** Security lapses have the potential to seriously harm a company's reputation. Businesses may show their dedication to protecting user data and upholding trust by putting into practice a secure web application architecture.

**5. Availability and Reliability:** Security concerns the availability and dependability of online applications as well as the prevention of illegal access. Downtime brought on by security issues may be avoided with a secure design.

**6. Cost-Efficiency:** Addressing security issues may be expensive. Organizations may be able to make considerable long-term resource savings by planning and putting in place security measures in a proactive manner.

**7. Learning and Skill Development:** Completing a project like this offers beneficial hands-on experience and strengthens competence in these fields, which are becoming more and more important in the job market.

**8. Scalability:** A safe architecture takes scalability into account and enables the web application to expand as user needs rise. For modern enterprises, knowing how to grow safely is crucial.

**9. Case Study and Best Practices:** This project may be used as an insightful case study by other businesses wishing to protect their web apps on AWS. Best practices and learnt lessons can be highlighted.

**10. Contributing to Research and Knowledge:** The project's conclusions and discoveries may further our understanding of web application development and cloud security, which may be of interest to the larger tech community.

The development and implementation of a secure web application architecture utilizing AWS is crucial for the immediate protection of sensitive data as well as for long-term corporate success, regulatory compliance, and the growth of knowledge in the field of cloud security. For businesses and industry experts in the field of technology, this initiative may have broad repercussions.

# **Setup an AWS Account**

Setting up an AWS account involves a series of steps to create an account on the Amazon Web Services platform.

Steps to Set Up an AWS Account:

**1. Visit the AWS Website:**

Go to the AWS website (https://aws.amazon.com/) and click on the "Create an AWS Account" button.

A screenshot of a computer

Description automatically generated

*“Figure 1: Sign-up AWS”*

**2. Provide Account Information:**

Enter your email address, choose an account name, and set a secure password for your AWS account.

**3. Contact Information:**

Fill in your contact information, such as name, address, and phone number.

**4. Payment Information:**

Provide payment details (credit/debit card information). AWS may charge a nominal fee to verify your identity, which will be reimbursed later.

**5. Choose a Support Plan:**

Select a support plan based on your preferences. AWS offers various support tiers, including basic (free) and premium (paid) support plans.

**6. Verify Your Identity:**

AWS may demand further verification steps, such as calling or texting your phone number to confirm it.

**7. Configure Billing Alerts:**

Once your account is set up, access the AWS Management Console.

Navigate to the Billing Dashboard.

Set up billing alerts to monitor your AWS usage and receive notifications when charges exceed predefined thresholds.

**8. Enable Multi-Factor Authentication (MFA):**

Consider enabling Multi-Factor Authentication (MFA) for added security. This requires an additional layer of verification when logging in to your AWS account.

**9. Explore AWS Services:**

Familiarize yourself with the AWS Management Console and explore the vast range of AWS services available.

**10. Access Documentation and Tutorials:**

Utilize AWS documentation, tutorials, and resources to understand various AWS services, best practices, and implementation guides.

**11. Start Using AWS Services:**

Begin using AWS services according to your project requirements, such as launching EC2 instances, setting up S3 buckets, or utilizing databases through RDS.

**12. Set Up Budgets and Controls:**

Establish spending budgets and controls within the AWS Billing Dashboard to manage costs and prevent unexpected charges.

# **Create Virtual Private Cloud (VPC)**

To create a Virtual Private Cloud (VPC) and implement it for a website on Amazon Web Services (AWS), we use several steps. A VPC allows you to logically isolate resources within the AWS cloud, providing a private network in which you can deploy your website.

* **Access AWS Console:**
  + Access the AWS Management Console and go to the VPC dashboard.
* **Create a VPC**
  + Click on "Create VPC" and provide a name, IPv4 CIDR block, and other relevant details. The CIDR block determines the IP address range for the VPC.
* **Set Up Subnets:**
  + Divide the VPC into subnets across different Availability Zones for high availability. Create public and private subnets. Private subnets rely on Network Address Translation (NAT) gateways for outbound traffic, whereas public subnets have direct internet connectivity.
* **Internet Gateway (IGW):**
  + Connect an Internet Gateway to the VPC to allow connectivity between VPC instances and the internet. This is crucial for public-facing resources like web servers.
* **Route Tables:**
  + Set up route tables to control the traffic between subnets. Assign the public subnet to a route table that includes a route to the Internet Gateway, and assign the private subnets to route tables that include a NAT gateway.
* **Security Groups:**
  + Create security groups for the instances, controlling inbound and outbound traffic.
* **Launch Instances:**
  + Launch EC2 instances in the appropriate subnets.
* **Load Balancer:**
  + In the public subnet, configure an Elastic Load Balancer (ELB) to spread incoming traffic over many instances. This improves availability as well as fault tolerance.
* **Monitoring and Logging:**
  + Enable AWS CloudWatch for monitoring and logging. Set up alarms to be notified of any abnormal behavior and use AWS CloudTrail for auditing API calls.

**Screenshots:**

A screenshot of a computer

Description automatically generated

*“Figure: 2 VPC”*

A screenshot of a computer

Description automatically generated

*“Figure: 3”*

A screenshot of a computer

Description automatically generated

*“Figure: 4”*

A screenshot of a computer

Description automatically generated

*“Figure: 5”*

# **Launch Website**

To launch a website on AWS, we follow these steps after setting up the VPC:

EC2 Instances: Launch EC2 instances to host the website. Choose the appropriate Amazon Machine Image (AMI) and configure instance details like the number of instances, instance type, and storage.

* **Security Group Configuration:**
  + Adjust security group settings to allow traffic on ports 80 (HTTP) and 443 (HTTPS) for web traffic.
* **Key Pairs:** 
  + Create or import a key pair for secure SSH access the instances. This key pair is essential for securely connecting to and managing EC2 instances.
* **Elastic Load Balancer (ELB):** 
  + Set up an Elastic Load Balancer to equally distribute traffic and improve the website's availability and fault tolerance.
* **Domain Configuration:** 
  + Associate the website's domain name with the public IP address or DNS name of the Elastic Load Balancer.
* **Monitoring and Scaling:** 
  + We set up monitoring with AWS CloudWatch to track the performance of the website.

A screenshot of a web security

Description automatically generated

*“Figure 6: Live Website”*

# **Configure Load Balancing**

To configure load balancing for the website on AWS, we utilize the Elastic Load Balancer (ELB) service:

* **Access the AWS Management Console:**
  + To access the EC2 dashboard, select "Load Balancers."
* **Create a Load Balancer:**
  + Click on "Create Load Balancer" and choose the appropriate type (Application, Network, or Classic Load Balancer).
* **Configure Settings:**
  + Specify the load balancer name, listener configurations (such as ports for HTTP and HTTPS) and associate it to the VPC.
* **Configure Routing:**
  + Define target groups, associating them with the EC2 instances. This guides the load balancer on how to distribute traffic.
* **Configure Health Checks:**
  + Configure health checks to monitor instance state and automatically divert traffic away from unhealthy instances.
* **Add EC2 Instances:**
  + Register the EC2 instances with the load balancer to enable traffic distribution.
* **Review and Create:** 
  + We review the settings and create the load balancer.

# **Implement Network Security**

* **Virtual Private Cloud (VPC) Configuration:**
  + Establish a well-defined VPC with a secure CIDR block to isolate resources logically.
  + Configure subnets across multiple Availability Zones for high availability and fault tolerance.
  + Implement public and private subnets to control internet access for specific resources.
* **Security Groups and Network Access Control Lists (NACLs):**
  + Create security groups to control inbound and outbound traffic at the instance level.
  + Utilize NACLs to add an additional layer of security by controlling traffic at the subnet level.
* **Internet Gateway and Route Tables:**
  + Attach an Internet Gateway to enable communication between instances and the internet.
  + Configure route tables to control traffic flow between subnets and the internet.
* **Secure Instance Deployment:**
  + Launch instances in private subnets for enhanced security.
  + Use Elastic Load Balancer (ELB) in the public subnet to distribute traffic securely.
* **Monitoring and Logging:**
  + Enable CloudWatch for real-time monitoring of network performance.
  + Set up alarms to receive notifications for any unusual network activities.
  + Implement CloudTrail for auditing API calls related to network configuration.
* **Data Encryption in Transit:**
  + Enforce SSL/TLS encryption for secure communication between instances and users.
  + Obtain SSL/TLS certificates from ACM and configure ELB to use them.

# **Enable HTTPS with ACM**

* **Certificate Provisioning:**
  + Utilize AWS Certificate Manager (ACM) to provision SSL/TLS certificates for the website.
  + Choose the appropriate certificate validation method (e.g., email validation or DNS validation).
* **Integration with Load Balancer:**
  + Associate the ACM certificate with the Elastic Load Balancer to enable HTTPS.
  + Configure the load balancer to terminate SSL/TLS, ensuring end-to-end encryption.
* **SSL/TLS Configuration:**
  + Implement strong SSL/TLS configurations to enhance security.
  + Regularly update SSL/TLS protocols and ciphers based on industry best practices.
* **Security Group Adjustments:**
  + Modify security groups to allow HTTPS traffic (port 443) securely.
  + Restrict unnecessary ports to minimize potential attack vectors.
* **Content Delivery Network (CDN) Integration:**
  + Integrate AWS CloudFront for content delivery, enhancing performance and security.
  + Ensure that ACM certificates are correctly configured with CloudFront distributions.

**Screenshots:**

A screenshot of a computer

Description automatically generated

*“Figure 7: ACM”*

A screenshot of a computer

Description automatically generated

*“Figure: 8”*

A screenshot of a computer

Description automatically generated

*“Figure: 9”*

A screenshot of a computer

Description automatically generated

*“Figure: 10”*

# **Implement Access Control**

* **Identity and Access Management (IAM):**
  + Configure IAM to manage user access to AWS services.
  + Implement the principle of least privilege to grant minimum necessary permissions to users and roles.
* **Role-Based Access Control (RBAC):**
  + Define roles based on job responsibilities and assign permissions accordingly.
  + Regularly review and update IAM roles to align with organizational changes.
* **Multi-Factor Authentication (MFA):**
  + Enforce MFA for privileged accounts to add an extra layer of security.
  + Implement MFA for AWS Management Console access and API calls.
* **Access Key Rotation:**
  + Set up automated access key rotation to minimize the risk associated with long-lived keys.
  + Regularly audit and revoke unnecessary access keys.
* **Resource Policies:**
  + Utilize resource-based policies to control access to resources such as S3 buckets.
  + Implement conditions in policies to enforce specific criteria for access.
* **Monitoring and Logging:**
  + Monitor IAM activity using CloudWatch to detect and respond to suspicious behavior.
  + Configure CloudTrail to log IAM events for auditing and compliance purposes.
* **Regular Security Audits:**
  + Conduct regular security audits to assess the effectiveness of access controls.
  + Perform periodic reviews of IAM configurations to identify and remediate any security gaps.

**Screenshots:**

A screenshot of a computer

Description automatically generated

*“Figure: 11”*

A screenshot of a computer

Description automatically generated

*“Figure: 12”*

A screenshot of a computer

Description automatically generated

*“Figure: 13”*A screenshot of a computer

Description automatically generated

*“Figure: 14”*

A screenshot of a computer

Description automatically generated *“Figure: 15”*

A screenshot of a computer

Description automatically generated

*“Figure: 16”*

A screenshot of a computer

Description automatically generated

*“Figure: 17”*

A screenshot of a computer

Description automatically generated

*“Figure: 18”*

A screenshot of a computer

Description automatically generated

*“Figure: 19”*

# **Data Encryption**

Data encryption is a critical component of our project's security strategy, aimed at ensuring the confidentiality and integrity of sensitive information. In this section, we will outline the steps involved in implementing data encryption using AWS Key Management Service and other relevant tools.

* **At Rest Encryption:**
  + Utilizing AWS KMS: Our project will employ AWS KMS to make and manage encryption keys for securing data at rest. KMS provides a centralized and secure key management solution.
  + Integration with Amazon S3: We will implement encryption for data stored in Amazon S3 buckets, ensuring that even if unauthorized access occurs, the data remains encrypted and unreadable.
* **In Transit Encryption:**
  + SSL/TLS Encryption: To secure data during transmission, our architecture will enforce SSL/TLS encryption. This will be achieved by obtaining SSL/TLS certificates from AWS Certificate Manager and configuring Elastic Load Balancer to use these certificates for secure communication.
* **Compliance Considerations:**
  + Meeting Regulatory Standards: With the increasing emphasis on data privacy laws like GDPR and CCPA, our project's encryption measures align with these regulations, minimizing the risk of legal repercussions and ensuring compliance.
* **Benefits of Data Encryption:**
  + Enhanced Data Security: Encryption safeguards sensitive information, preventing unauthorized access and data breaches.
  + Regulatory Compliance: Adherence to data protection regulations and standards, fostering trust among users and stakeholders.
  + Data Integrity: Encryption ensures that data remains unaltered during storage and transmission.

# **Monitoring and Logging**

Monitoring and logging play a pivotal role in proactively identifying and responding to security incidents. This section outlines the implementation of real-time monitoring and logging using Amazon CloudWatch and AWS CloudTrail.

* **Amazon CloudWatch:**
  + Real-time Resource Monitoring: Our project will leverage Amazon CloudWatch to monitor various AWS resources, including EC2 instances and ELB. This facilitates the detection of performance issues or abnormalities promptly.
  + Alarms and Notifications: CloudWatch alarms will be configured to trigger notifications in case predefined thresholds are breached, allowing for immediate response to potential security threats.
* **AWS CloudTrail:**
  + Auditing AWS API Calls: AWS CloudTrail will be enabled to record API calls, providing an audit trail of actions taken within the AWS environment. This aids in forensic analysis and ensures accountability.
  + Compliance Monitoring: CloudTrail assists in monitoring and ensuring compliance with security policies and governance standards.
* **Continuous Security Testing:**
  + Simulating Attack Scenarios: Our project will use AWS services such as Guard Duty in conjunction with CloudWatch to simulate and detect common attack scenarios, such as SQL injection and DDoS attacks. This proactive approach helps identify vulnerabilities and strengthens the overall security posture.
* **Benefits of Monitoring and Logging:**
  + Early Threat Detection: Real-time monitoring enables the identification of security threats as they occur, allowing for swift response and mitigation.
  + Forensic Analysis: Detailed logs from CloudTrail support forensic analysis, aiding in understanding the sequence of events during a security incident.
  + Compliance Assurance: Continuous monitoring contributes to adherence to compliance standards, reinforcing the robustness of our security measures.

**Screenshots:**

A screenshot of a computer

Description automatically generated

*“Figure: 20”*

A screenshot of a computer

Description automatically generated

*“Figure: 21”*

A screenshot of a computer

Description automatically generated

*“Figure: 22”*

A screenshot of a computer

Description automatically generated

*“Figure: 23”*

A screenshot of a computer

Description automatically generated

*“Figure: 24”*

A screenshot of a computer

Description automatically generated

*“Figure: 25”*

A screenshot of a computer

Description automatically generated

*“Figure: 26”*

# **Backup and Disaster Recovery**

* **Automatic Backups:**
* Snapshots from Amazon RDS
* Automated backups for relational databases governed by retention policies.
* Versioned storage of essential data and static assets is ensured by Amazon S3 versioning.
* **AWS Backup Service:**
  + Manage backups centrally for multiple resources.
  + established backup strategies for RDS databases, EC2 instances, and other crucial parts.
* **Infrastructure as Code (IaC):**
  + Used AWS CloudFormation to specify and create versions of the infrastructure.
  + allows for the full environment to be recreated in the event of a calamity.
* **Cross-Region Replication:** 
  + For further redundancy, key resources were replicated across several AWS regions.
  + lessens the effects of a local power loss.
* **Amazon Glacier for Archiving:** 
  + For less money, store important backups on Amazon Glacier for an extended period of time.
  + To move older backups from S3 to Glacier, set up lifecycle policies.
* **Snapshot Testing:**
  + Make sure your recovery and backup procedures are operating properly by testing them on a regular basis.
  + These covers testing your complete infrastructure’s recovery as well as data backup.
* **High Availability (HA) Architecture:**
  + Keep high availability in mind when you design your application architecture.
  + To provide redundancy, use several Availability Zones, auto-scaling groups, and load balancing.
* **Monitoring and Alerts:** 
  + Use AWS CloudWatch or other monitoring tools to keep a check on your infrastructure.
  + To be informed of possible problems before they get serious, set up alerts for important indicators.

# **Continuous Security Testing**

* **Automated Vulnerability Scanning:**
  + AWS Inspector and other tools are used to conduct routine automated scans to find vulnerabilities.
  + CI/CD (continuous integration/continuous deployment) pipeline with scheduled scans incorporated.
* **Penetration Testing:**
  + To replicate actual attacks, periodic penetration testing is conducted.
  + Emphasis on testing the underlying AWS infrastructure in addition to the application layer.
* **Threat Modelling:**
  + To find possible security risks and weaknesses during the design stage, threat modelling sessions were held.
  + Techniques for mitigation incorporated into the design phase.
* **Code Analysis:**
* Used tools for static code analysis to find security holes in the product development phase.
* Code repositories are continuously checked for security vulnerabilities.

# **Structure for Compliance**

* **Regulatory Standards:**
* Determining the applicable regulations (such as GDPR, HIPAA, etc.) in accordance with the type of online application.
* putting procedures and controls in place to ensure compliance with the requirements.
* **Amazon Compliance Services:**
* Use of AWS compliance services to help ensure that industry-specific standards are followed.
* evaluations and audits on a regular basis to guarantee continued adherence to AWS security best practices.

**Ethical Considerations**

* **User Privacy:**
* To protect user data, privacy-by-design principles are put into practice.
* open and honest dialogue with users about data usage, storage, and collection procedures.
* **Informed Consent:**
* The creation of explicit guidelines for user consent to data processing.
* the deployment of systems for efficiently obtaining and handling user consent.
* **Responsible Data Handling:**
* Sensitive data encryption while it's in use and in transit.
* Conduct routine evaluations and audits of data handling procedures to reduce any hazards.

**Source Code**

**“*Source-Code:*** [**https://github.com/Jsuthar2/ift520**](https://github.com/Jsuthar2/ift520)**”**

**“*Project Live Link:*** [**http://ift520project.edu.s3-website-us-east-1.amazonaws.com**](http://ift520project.edu.s3-website-us-east-1.amazonaws.com)**”**

# **Reference**

Amazon. (n.d.). AWS Architecture Center. Retrieved from AWS:

<https://aws.amazon.com/architecture/>

AWS. (2019). What Is Amazon EC2? - Amazon Elastic Compute Cloud. Amazon.com.

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>

(2023). Amazon.com.

<https://docs.aws.amazon.com/vpn/?icmpid=docs_homepage_networking>

Simplilearn. What is AWS? Retrieved from

<https://www.simplilearn.com/tutorials/aws-tutorial/what-is-aws>

Web Application Deployment on AWS. Dev.to.

<https://dev.to/aws-builders/web-application-deployment-on-aws-31a7>

AWS Trusted Advisor

<https://aws.amazon.com/premiumsupport/technology/trusted-advisor/>

NIST Cybersecurity Framework

<https://www.nist.gov/cyberframework>

AWS Cloud Security

<https://d1.awsstatic.com/whitepapers/aws-security-best-practices.pdf>