# Cloud Security Implementation: Securing WordPress on AWS

## 1. Introduction

Cloud security has become a critical aspect of modern IT infrastructure, ensuring that cloud-hosted applications remain protected against threats such as unauthorized access, data breaches, and cyberattacks. In this project, we implemented a \*\*secure WordPress deployment\*\* on AWS using a LAMP stack and various security measures, including \*\*firewall configurations, SSL certificates, logging, monitoring, and penetration testing\*\*.

This report details the \*\*step-by-step implementation\*\*, challenges encountered, and future enhancements required to improve security.

## 2. Cloud Infrastructure Setup

### A. AWS EC2 Instance Deployment

The first step was to set up an \*\*EC2 instance\*\* on AWS to host our WordPress site. The instance was configured with:

- \*\*Amazon Machine Image (AMI):\*\* Amazon Linux 2023

- \*\*Instance Type:\*\* t2.small with 12GB storage

- \*\*Security Group:\*\* Configured to allow SSH, HTTP, and HTTPS traffic

- \*\*Key Pair:\*\* ED2559 key used for SSH authentication

To ensure better security, only specific IP addresses were allowed for SSH access.

### B. LAMP Stack Installation and Security Hardening

After setting up the EC2 instance, a \*\*LAMP stack (Linux, Apache, MySQL, PHP)\*\* was installed and configured with security enhancements:

- \*\*MariaDB database setup\*\* with root password protection

- \*\*Apache security hardening:\*\*

- Disabled directory browsing

- Restricted HTTP request trace

- Prevented clickjacking attacks using security headers

- \*\*PHP security hardening:\*\*

- Removed unnecessary modules

- Configured file permission restrictions

- \*\*Database Hardening:\*\*

- Removed anonymous users

- Disabled remote root login

- Applied strong encryption for sensitive data

### C. WordPress Deployment

Once the LAMP stack was secured, WordPress was installed:

- \*\*Database Creation:\*\* A dedicated database was created for WordPress, restricting access.

- \*\*File Permissions:\*\*

- wp-config.php was secured with read-only permissions.

- File editing was disabled within WordPress settings to prevent unauthorized modifications.

- \*\*User Authentication:\*\*

- 2-Factor Authentication (2FA) was enabled using a WordPress security plugin.

- Strong password policies were enforced.

## 3. Security Enhancements

### A. SSL Certificate Integration

To enable \*\*HTTPS\*\*, an \*\*SSL certificate\*\* was acquired and implemented:

- \*\*Domain Registration:\*\* Namecheap was used to register a domain.

- \*\*SSL Certificate Activation:\*\* A free SSL was issued and installed.

- \*\*AWS Certificate Manager:\*\* The certificate was imported to AWS for better scalability.

### B. Auto-Scaling and Load Balancing

To ensure high availability and fault tolerance, an \*\*AWS Auto-Scaling Group and Load Balancer\*\* were configured:

- \*\*Virtual Private Cloud (VPC) Configurations:\*\*

- Created public and private subnets across two availability zones.

- Configured \*\*NAT Gateways\*\* to enable private subnet communication.

- \*\*Bastion Host Deployment:\*\* Used to securely connect to EC2 instances within private subnets.

- \*\*Elastic File System (EFS) and RDS:\*\*

- EFS stored WordPress files across multiple instances.

- RDS MySQL hosted the database, ensuring redundancy.

- \*\*Load Balancer:\*\*

- HTTPS traffic was routed via the Application Load Balancer.

- Web Application Firewall (WAF) was enabled to prevent DDoS attacks.

### C. Automated Monitoring and Email Alerts

To ensure real-time security monitoring and proactive alerting, automated \*\*Amazon Simple Notification Service (SNS)\*\* email notifications were configured for:

- \*\*High CPU Usage Alerts:\*\* CloudWatch alarms were set to trigger when CPU utilization exceeded 70%.

- \*\*Unauthorized Login Attempts:\*\* CloudTrail logs tracked SSH access attempts and sent alerts on failed login attempts.

- \*\*Unusual Network Traffic:\*\* AWS GuardDuty was integrated to detect suspicious inbound/outbound traffic patterns.

- \*\*WordPress Admin Page Access:\*\* Any unauthorized access attempts to the WordPress admin panel triggered email alerts via SNS.

### D. WordPress Plugin Security

Several security plugins were installed and configured:

- \*\*WordFence Security:\*\*

- Added \*\*firewall protection\*\*

- Enabled \*\*2FA authentication\*\*

- Provided real-time malware scanning

- \*\*Sucuri Security:\*\*

- Monitored website traffic and detected anomalies

- \*\*Jetpack Security:\*\*

- Logged activity and monitored vulnerabilities

These plugins ensured that the website was safeguarded from common attacks such as brute force and SQL injection.

## 4. Logging and Monitoring

### A. AWS CloudWatch and CloudTrail

To maintain security oversight, AWS logging tools were configured:

- \*\*CloudWatch:\*\* Monitored CPU usage, memory, and unusual activity.

- \*\*CloudTrail:\*\* Tracked all API calls made within the AWS environment.

- \*\*Automated Email Notifications:\*\*

- Configured SNS to send email alerts on security incidents.

- Alerts triggered for excessive CPU usage, unauthorized access attempts, and anomalous behavior.

- Event-driven notifications ensured rapid response to security threats.

### B. Third-Party Monitoring with DataDog

A \*\*DataDog agent\*\* was integrated into the EC2 instance for additional insights:

- Monitored \*\*live processes\*\*

- Analyzed \*\*network traffic\*\*

- Provided \*\*visual alerts\*\* in case of anomalies

- Integrated with \*\*Slack and Email Alerts\*\* for instant notifications.

## 5. Conclusion

This project successfully demonstrated the deployment of a \*\*secure WordPress website on AWS\*\* by implementing \*\*best security practices\*\* at both infrastructure and application levels. By integrating \*\*logging, monitoring, and security testing\*\*, we created a resilient cloud-based solution.

Future improvements will focus on \*\*enhancing IAM controls, automating security audits, and implementing additional redundancy mechanisms\*\* to further strengthen cloud security.