Creating, Hosting, and Sustaining a Website on the Cloud through AWS

Samuel Diaz

MIS 690: Cloud Computing

San Diego State University

November 01, 2024

Table of Contents

A. System Creation, Deployment, & Maintenance

I. Project Overview

Having an online presence is crucial in today's era for employers to seek candidates that match the skills necessary to complete the job they are filling. A personal website, if done correctly, can have a significant impact on the decision to invite the candidate for a job interview and know more about them. However, if you are a small business or an individual attempting to showcase your skills, setting up and maintaining your own website infrastructure can be costly and require strong networking skills. Hence, making it unfeasible to setup and maintain the infrastructure in the long run.

Thankfully, the emergence of cloud computing helps bridge this gap by providing infrastructure services through an interactive online interface users can easily manage, or Infrastructure as a Service (IaaS). These services are offered at a low cost, with users paying only for what they use. In the field of web development, individuals and small business entities can benefit from this by only paying for the resources they need to run a website. As an example, I do not expect my own personal website to be accessed by thousands of visitors, hence I will pay a lower rate due to the minimal number of HTTP request messages being sent to the server.

My project is set to use Amazon Web Services to host and monitor an online personal website. I will prove that with moderate technological skills, it is financially feasible for an individual to create, deploy, and maintain a website through cloud computing services, in this case AWS. Also, I will implement cloud best practices and security measures to keep my website safe and free from attackers. All of this will be achieved while staying under AWS's free tier subscription.

II. Creating the Personal Website (samueldiaz.tech)

As an individual skilled in programming, I have decided to learn the web development languages to create my own HTML5, CSS3, and Javascript code to build my own personal website. As a result, this has saved me a few bucks from needing to use my own Content Management System (CMS) to develop my website.

CMSs are a great tool to sue for those individuals who lack programming skills. Although CMS providers start by offering free services, they might require you to upgrade to a higher tier account if you want to make your website more visually appealing and interactive, such as by adding animations or pictures. Also, CMS providers tend to charge for use of pre-written templates to design your website. While it remains cost–effective to create your own website using a CMS, it is best to code your own website from scratch. You will save money and have unlimited creativity on its modification and design. CMS are recommended for those without any coding experience or who lack the creativity to create a visually appealing UI/UX design.

III. AWS Services Used

The following is a brief overview of the AWS services used to deploy and maintain the personal website.

1. **Amazon S3 (Simple Storage Service)**:
    - Description: Amazon S3 is an object storage service that enables the storage and retrieval of any amount of data at any time. It provides high scalability and high durability and easily integrates with other AWS services.

- Use Case: Two S3 buckets were created. A publicly accessible S3 bucket was deployed to store static assets and host the website. An encrypted, private S3 bucket was used to store the CloudTrail logs.
  - *Note: S3 was chosen over EC2 for this specific project. S3 is ideal for static web hosting since it is more cost-effective, automatically scalable, integrates with CloudFront, and has access to several preset security features.*

2. **AWS IAM (Identity and Access Management)**:
   - Description: AWS IAM helps securely control access to AWS resources. It provides access control and identity federation.
   - Use Case: IAM roles were employed for access control and to enhance security.
3. **AWS CloudTrail**:
   - Description: CloudTrail is a tool that focuses on operations and risk management. it provides governance, compliance, and auditing of your AWS account. It captures API calls and delivers log files to an S3 bucket.
   - Use Case: CloudTrail was used to monitor activity within the AWS environment. Specifically, the S3 bucket where the website data is stored.
4. **Route 53:**
   - Description: Route 53 is a scalable and highly available domain name system web service. It connects end-user requests to system resources on the internet.
   - Use Case: Route 53 was implemented for domain name creation and management.
5. **Key Management Service**:

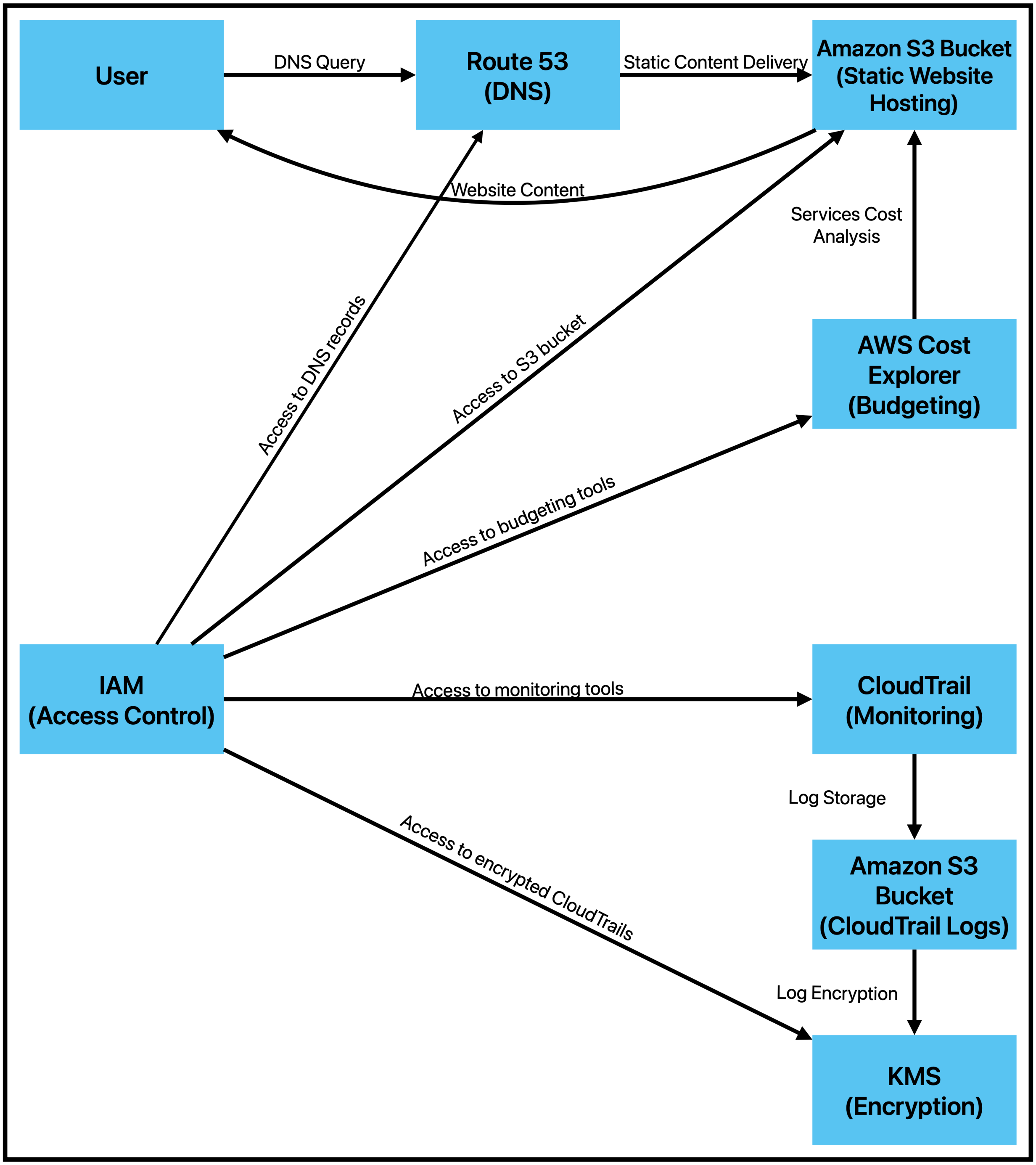
- Description: KMS is a managed service that enables you to create and control the encryption keys used to secure your data.
- Use Case: KMS was used to securely generate and manage AWS encryption keys, specifically for CloudTrail.

6. **AWS Cost Explorer/AWS Cost Anomaly Detection**
   - Description: AWS Cost Explorer provides tools to view and analyze costs and usage of resources. AWS Cost Anomaly Detection uses machine learning to identify anomalous spending patterns.
   - Use Case: These tools were used to ensure the website stays within budget constraints.

IV. System Architecture

The architecture can be summarized with the following diagram:

(next page)



V. Implementation Details

Note: For a detailed explanation of the creation, deployment, and final result of the process for each service, refer to the MIS690_AWSProjectScreenshots file attached to this document.

Summarized Implementation Details:

1. **Creation of S3 bucket**: A publicly accessible S3 bucket holds the HTML, CSS, JavaScript, and images necessary to display the website. A second private, encrypted S3 bucket holds the CloudTrail logs.
2. **Use IAM to Designate Users and Roles**: Created two users, 'Admin' and 'Developer', assigning only the necessary resources to complete their tasks. Additionally, created two roles, 'Auditor' and 'ReadOnlyBilling', to give temporary access to specific resources for at most an hour. Users are meant to be long-term while roles are meant to be temporary.
3. **Manage Domain Name on Route 53**: Domain name management takes place through Route 53, ensuring the website requests direct users to the correct S3 bucket.
4. **AWS CloudTrail**: Tracks the modifications and updates each role contributes to the S3 resources, ensuring integrity. It also logs data events, such as visitor and user access to the website.
5. **Key Management Service**: Encrypts the CloudTrail file within the S3 bucket to add a layer of security. If someone were to access this file, they would require a decryption key to view its contents.

VI. Security Configurations

- **Principles of Least Privilege**:
    - **(AWS IAM) Creation of an Admin Account**: As a best practice, I have chosen to follow the Amazon AWS documentation and create an Admin account to avoid using the root account for tasks that do not require root-level access. This admin account requires MFA authentication to strengthen sign-in requirements.

  - **(AWS IAM) Handling Custom Policies**: This admin account uses the policy ‘AdministratorAccess’ which is a managed policy provided by AWS that grants full access to all AWS resources and services. No further modifications or restrictions were required for the Admin user. However, it is always advisable to apply the principles of least privilege to every role, group, and policy before deploying. For example, if my business held personal health information, I would want to protect this data from the Admin user as it would be violating HIPAA laws.

- **Separation of Duties:**
  - **(AWS IAM) User and Roles Creation:** Created the ‘Admin’ user to employ AWS best practices. Additionally, ‘Developer’ user created with permissions limited to S3 and EC2 resources, adhering to the principles of least privilege. These permissions allow developers to manage website resources while minimizing unnecessary access to other AWS services.

    In addition to the creation of users, two roles were developed to give access to specific resources for at most an hour. The ‘Auditor’ role gives temporary read-only access to CloudTrail logs while the ‘ReadOnlyBilling’ role provides temporary, read-only access to billing statements. The ‘Auditor' role permits auditors to review account activity for security and compliance purposes. This ensures transparency while maintaining tight control over sensitive log data. The ‘ReadOnlyBilling’ role enables users to view account charges and usage without the ability to modify billing configurations or access other sensitive areas. This role was created for auditors who might need temporary access to billing statements to verify compliance with financial regulations or validate spending against organizational policies.

- **Data Lifecycle Management:**
    - **(Amazon S3) Enabling Bucket Versioning**: Versioning for buckets has been activated to ensure any modifications made are retrievable.
    - **(Amazon S3) Creating Lifecycle Policies**: Lifecycle policies have been activated to minimize storage by automatically deleting obsolete bucket versions that have reached 180 days since creation. To allow flexibility, the top three most recent versions of the bucket will be kept indefinitely. This way we can ensure a sustainable balance between data minimization and data retention. Furthermore, we can minimize costs while maximizing logging needed to ensure old bucket versions are available for retrieval in a timely manner. Since this is my personal website with unregulated access, implementing data governance for storing, retrieval, and maintaining the website was fairly easy. If for example, my website held accounting or financial information I would have to be more mindful of the data retention policies based on my region and location. The Sarbanes–Oxley Act might take precedence over my data and I might be required to store this data for at least seven years.
- **Data Segregatio**n:
    - **(Amazon S3) Data Isolation for Enhanced Security**: The CloudTrail logs were placed in their own, separate bucket. Placing CloudTrail logs inside the publicly accessible website could allow unauthorized access or expose sensitive logging information. So,

isolation of the logs in a separate encrypted bucket reduces the risk of data breaches and aligns with AWS best practices for securing audit and compliance data.

- **Auditing and Compliance**:
    - **(AWS CloudTrail) Logs Management Strategy**: CloudTrail was enabled to track user activity, such as any modifications made to the S3 bucket. This services supports data analytics, allowing the administrator to track attributes such as eventName, sourceIPAddress, and userAgent. The first attribute, eventName, provides the action taken by an individual accessing the website, for example, GetObject. The second attribute, sourceIPAddress, tracks the IP address of the user accessing the website. Lastly, userAgent identifies the client accessing the website, such as a browser or a script.
- **Encryption**:
    - **(AWS KMS) SSE-KMS to Encrypt Logs**: Use Server-Side Encryption with AWS Key Management Service for encrypting CloudTrail logs to ensure the protection of data at rest. Using SSE-KMS ensures encryption keys are stored and managed securely, providing additional security features compared to default encryption. CloudTrail encrypts the log files before storing them in the specified S3 bucket. Decrypting the logs requires permissions to use the specified KMS key. This adds an extra layer of security, ensuring that only authorized users with access to both the S3 bucket and the KMS key can view the logs.
    - **(Amazon S3) Bucket Encryption for Data at Rest**: The two buckets created have been encrypted using Server-side encryption with Amazon S3 managed keys (SSE-S3). This

encryption mechanism provides strong protection without requiring the management of encryption keys, adhering to AWS security best practices.

B. Exploring Additional System Security Measures

I. Project's Limitation and Scope

It is worth mentioning that more than 85% of the monthly AWS Free Tier limit was consumed by the end of the project. As the project is meant to be kept at a minimal cost, some configurations were not implemented to stay within the AWS Free Tier requirements. This introduces limitations in terms of scalability, traffic handling, and the use of premium security features. My website is meant to remain affordable and scalable, while meeting essential security requirements.

II. Security Measures Considerations

This section presents supplementary security considerations that could have been implemented to fortify the concept of security in depth for our website. These are optional enhancements for future scalability rather than necessary for a minimal-cost personal website.

1. **Strengthen Sign-In Configurations**:
   - Instead of enforcing MFA solely on the root and admin account, it can be applied for all users for added security.
   - User passwords should be updated regularly, with an expiration policy in place to mandate periodic changes.
2. **Modify Static Website Architecture**:
   - Instead of hosting the website in an S3 bucket, the website can be hosted in the EC2 environment. This way, we can use Amazon Inspector to harden the EC2 instance according to CIS recommendations. For the purpose of this project, an EC2 instance is not required given the website is static, of minimal size, and S3 provides the

resources necessary to secure and quickly access the website. Furthermore, EC2 charges substantially more for its use of resources. Hence, an EC2 instance would be best for a website requiring substantial data processing, additional security considerations, and constant updates.

- A VPC configuration can be applied to route traffic to secure the EC2 instance used to host the website.

3. **Additional Firewall Protection**

   - Implementing a Web Application Firewall (WAF) could further enhance the security of the hosted website. This service protects against common vulnerabilities. Unfortunately, this is a pay-per-use platform that charges on an hourly/monthly basis. Also, this is targeted primarily for e-commerce websites or those handling sensitive data.

4. **Use of an Intrusion Detection System (IDS):**

   - Using AWS GuardDuty to detect threats can help mitigate risks. It uses AI/ML to analyze patterns and detect anomalous behaviors. With a free trial, but also pay per use.

5. **Enhance Logging with CloudWatch:**

   - Similar to CloudTrail, CloudWatch logs activity but it’s purpose is to collect metrics to provide insights into ways to manage and optimize system performance. This is also a pay-per-use service and not optimal for the purpose of this project.

6. **Application Hardening through Benchmarking**:

- We could apply system hardening concepts assuming, we utilized an EC2 instance to host our website. AWS Inspector is an effective tool for this purpose as it uses benchmarks from prominent information security organizations to harden applications. As an example, AWS Inspector uses CIS Benchmarks as part of its vulnerability and security assessment capabilities for Amazon EC2 instances and container images. This service is able to perform security best practices checks based on the CIS Benchmarks for Amazon Linux, Ubuntu, Windows, and other operating systems. For hardening, AWS Inspector scans EC2 instances against CIS Benchmarks and identifies areas where the instance configuration deviates from the recommended settings.

7. Multi-Region Redundancy

- Deploy the website within multiple regions to add redundancy in case of system failures. To save costs, the personal website was deployed in a single region, without redundancy or disaster recovery configurations. This means that the website is vulnerable to region-specific outages.

C. Support Plan for the System

I. Business Budget & Expenses

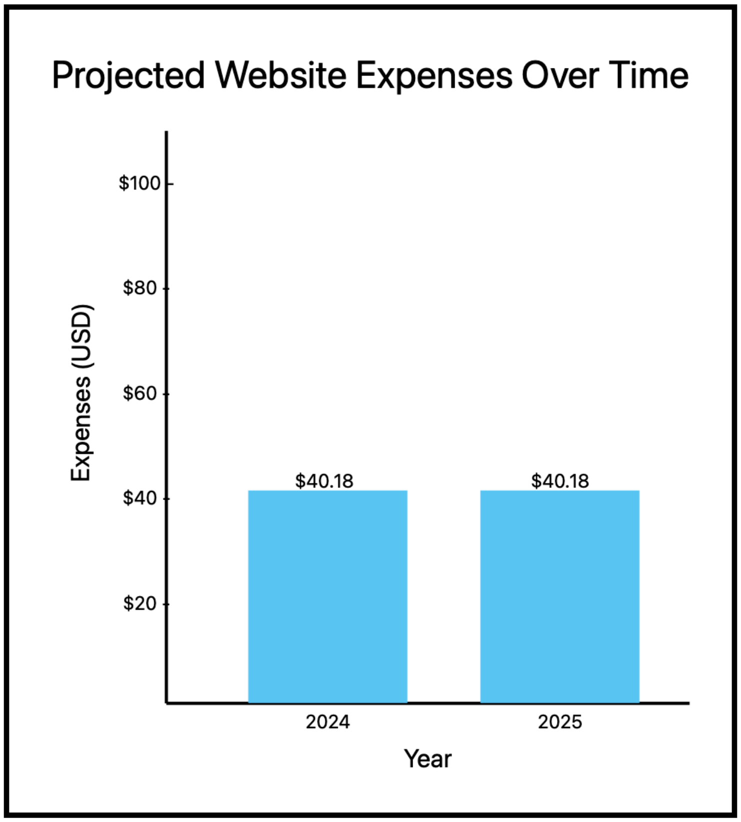
Costs will be maintained at a minimum to demonstrate the financial feasibility of cloud computing services. The following table summarizes the current expenses incurred to create and deploy the website from 11/04/24 – 12/31/24.

DATE	COMPANY	EXPENSE	AMOUNT
11/02/24	Amazon	Website’s Domain Name (1-year)	$40.18
Total			**$40.18**

The following table summarizes the projected expenses within the next year 01/01/24 – 12/31/25. It is important to note that by the project’s completion, over 85% of the monthly AWS Free Tier limit had been utilized. With this in mind, expenses are not anticipated to exceed the monthly AWS Free Tier monthly limit.

DATE	COMPANY	EXPENSE	AMOUNT
11/02/25	Amazon	Renewal of Website’s Domain Name (1-year)	$40.18
Total			**$40.18**

Overall, the expense are expected to remain constant over the next year (not accounting for potential inflation adjustments):



II. Maintenance Schedule

Regularly scheduled monitoring and maintenance of a system allows for a secure, reliable integration. It is crucial for user to assess the system's health at its specific point in time to intervene or remediate attacks as necessary. The following is a detailed schedule for the system's maintenance. It follows the concepts of the CIA triad to best support the goals of confidentiality, integrity, and availability, ensuring the system remains secure and operational. I've also added additional details to include what should be done if the system were to become large scale.

Personal Website Maintenance

- **Daily**:
    - Confidentiality:
        - Check inbox for any email from the AWS Management Console regarding any critical alerts or issues. (Also, Integrity + Availability, depending on the type of alert).
    - Integrity:
        - Daily log revision would be optimal, but it is tedious for a single individual to log into the console and review the system on a daily basis. Therefore, it is crucial to enable automatic emails to my account each time CloudTrail detects any access, modifications, or changes made to the S3 buckets. If needed, I will review the access logs to my S3 bucket for any abnormal behavior in CloudTrail.
    - Availability:
        - Ensure website is accessible and responding as expected.
- **Weekly**:
    - Confidentiality:
        - Check your IAM user's permissions and access keys. (Also, integrity)
    - Integrity:
        - Confirm the bucket's public access settings have not been inadvertently altered.
        - Check S3 bucket policies and access control lists (ACLs) for unintended changes.
        - Ensure you are using Multi-Factor Authentication (MFA) for your account when logging in.

        - Review key logs in CloudTrail for unusual activity, using saved searches or filters to make log reviews quick and efficient.
    - Availability:
        - Verify that static content is loading correctly.
        - On Route 53, confirm DNS records are accurate and functioning.
- **Monthly**:
    - Confidentiality:
        - Review and rotate encryption keys in KMS.
    - Integrity:
        - Use versioning or checksums to ensure the integrity of your data stored in S3.
    - Availability:
        - Use AWS Cost Explorer to confirm you're within your expected budget.
        - Ensure any backups or S3 versioning policies are capturing the data and can be restored.
        - Confirm the lifecycle policy on the S3 bucket works appropriately
        - Ensure Route 53 health checks are passing.
- **Annually:**
    - Confidentiality:
        - Review IAM policies for any possible redundancies or over-privileged roles.
        - Identify and delete any unused resources like old S3 buckets, IAM roles, or DNS records.
        - Ensure SSE-KMS keys are properly rotated and access policies are secure.

- Integrity:
  - Validate that key policies remain restrictive, with appropriate IAM roles having access.
- Availability:
  - Renew the domain registration via Route 53.
  - Assess whether you need to make improvements, such as website code updates and modifications, adding caching and content delivery, or introducing any further automation for log reviews or security configurations. (Encompasses all three)

Maintenance for Large-Scale, Dynamic Website (i.e., commercial):

Note: Adjustments would have to be made, transitioning from storing the website in an S3 bucket to an EC2 instance which better handles dynamic and large scale data. Also, a database would be mandated to store the large amounts of customer data. Note, these changes will enhance security but incur additional costs.

- **Daily**:
  - Confidentiality:
    - Monitor CloudTrail logs for anomalous activities, focusing on unauthorized access to resources.
  - Integrity:
    - Monitor CloudTrail logs for anomalous activities, focusing on unauthorized changes.
  - Availability:

    - Use AWS CloudWatch to monitor key metrics like EC2 instance CPU, memory usage, disk I/O, and S3 request rates.
    - Review logs for the EC2 instance to identify unusual patterns like excessive 404s or spikes in traffic.
- **For every week**:
  - Confidentiality:
    - Verify permissions and policies on S3 buckets to ensure they remain secure.
    - Ensure IAM users and roles follow the principle of least privilege.
  - Integrity:
    - Use AWS Config to ensure that resources adhere to defined configurations, preventing unintended changes.
  - Availability:
    - Tune indexes to improve query performance.
    - Confirm EC2 instance backups are current and can be restored.
    - Review resource usage (CPU, memory, storage) and scale instance types or add resources as needed.
    - On S3, confirm lifecycle policies are archiving or deleting unused static assets as intended.
    - On Route 53, review DNS settings, such as failover, and verify they are accurate.
    - If using AWS WAF, ensure that rules for blocking threats like SQL injection or DDoS attacks are up to date.
  - All Three (Confidentiality, Integrity, Availability)

- Check for available OS and software updates on EC2 instance.

- **For every month**:
    - Confidentiality:
  - Review and rotate encryption keys in KMS.
  - Integrity:
    - Review CloudTrail logs for unusual or unauthorized actions.
  - Availability:
    - Review EC2 Auto Scaling Groups and ensure scaling policies are appropriate for traffic patterns.
    - Analyze CloudWatch metrics to identify and address bottlenecks.
    - Use AWS Cost Explorer to identify unnecessary expenses
    - Assuming use of CloudFront, ensure cache policies and distribution settings are optimized for performance and cost.
    - Fine-tune database queries and indexing strategies.
    - Confirm backup retention settings align with business needs.
  - All Three (Confidentiality, Integrity, Availability)
      - Conduct vulnerability scans using AWS Inspector.
    - Patch the EC2 instance OS, database, and application frameworks. (
- **Quarterly:**
  - Confidentiality:
    - (+Integrity) Review IAM policies, EC2 security groups, and S3 bucket configurations.

    - Availability:
        - Test your disaster recovery plans by restoring EC2 instances from backups or replicating static assets from S3.
        - Reassess Auto Scaling policies for EC2 to ensure they meet current traffic demands.
        - Evaluate whether additional regions or Availability Zones are necessary for redundancy.
        - Review and optimize the database schema to ensure scalability and performance.
    - All Three (Confidentiality, Integrity, Availability)
        - Conduct penetration testing or hire a third party for a full security assessment.
- **Annual**:
    - Confidentiality:
        - (+Integrity) Rotate KMS keys, IAM access keys, and other sensitive credentials.
    - Integrity:
        - Review all database security configurations.
    - Availability:
        - Remove outdated backups, snapshots, or unused resources.
        - Assess if your EC2 instance types, Auto Scaling strategies, and other resources remain optimal for your current and projected needs.
        - Automate backups, patching, and scaling with tools like AWS Systems Manager, Lambda functions, and Auto Scaling policies.
    - All Three (Confidentiality, Integrity, Availability)

- Audit all AWS services and resources for unused or misconfigured items.
- Ensure compliance with current industry standards if applicable.
- Review AWS compliance tools and reports to validate adherence to relevant regulations.

III. Roles and Responsibilities

Roles for Personal Website Maintenance

The trade-off between security, scalability, cost effectiveness, and the effort required to manage the website has been carefully considered. The personal website maintenance schedule is designed for a sole individual to handle are responsibilities efficiently.

Roles for Large-Scale, Dynamic Website (i.e., commercial):

1. Cloud System Administrator [$70,000 salary]:
   - Manages the underlying infrastructure, including EC2 instances and S3 storage.
   - Applies OS updates, patches, and performs server hardening.
   - Configures and maintains IAM roles, permissions, and policies.
   - Ensures backups are created and tested for disaster recovery purposes.
   - Keeps up to date with the latest data policies (i.e., data retention, access, modification, etc.) governing the region.
2. Cloud Security Analyst(s) [$75,000 salary]:
   - Monitors security logs and investigates potential incidents using CloudTrail and AWS Security Hub.
   - Manages encryption policies and KMS key rotation.

- Configures and monitors AWS Web Application Firewall (WAF) to mitigate threats like SQL injection or DDoS attacks.
- Conducts periodic vulnerability scans and penetration testing.

3. Full Stack Software Engineer or Web Developer(s) [$101,000 salary]:
   - Manages website content, application updates, and codebase changes.
   - Updates static and dynamic content on the website.
   - Ensures compatibility and optimization of web applications with the AWS environment.
   - Implements caching and performance optimization strategies using tools like AWS CloudFront.
   - Troubleshoots issues with the website's functionality.
4. Database Administrator [$80,000 salary]
   - Manages the database used for dynamic content.
   - Performs regular database optimization, indexing, and consistency checks.
   - Configures and monitors database backups and ensures restorability.
   - Analyzes and optimizes database queries to improve performance.
5. DevOps Engineer [$108,000 salary]
   - Automates infrastructure management and deployment using tools like CloudFormation or Terraform.
   - Configures and maintains Auto Scaling Groups, Load Balancers, and Route 53 DNS failover settings.
   - Monitors and optimizes resource usage and costs with tools like AWS Cost Explorer.

- Manages CI/CD pipelines to streamline development and deployment.

6. Project Manager (Preferably holds a Juris Doctor (JD)) [$140,000 salary]

   - Architect, design, plan, lead, and execute changing project goals and needs.
   - Coordinate with developers for new feature releases or content formats.
   - Stays informed with current data policies governing the region.
   - Coordinates with admin on how to best implement IAM roles given the current state of the law.
   - Mitigates legal matters and addresses issues when necessary.
   - Monitors analytics tools for website performance insights.

D. Incident Response Plan

I. Incident Detection and Response Steps

1. **Detection**:

   - Use AWS CloudTrail to set up real-time monitoring and alerts for unusual activity.
   - Review CloudTrail logs, S3 bucket access logs, and Route 53 DNS query logs for signs of anomalies. These may include IP addresses originating from outside the United States and suspicious API calls, like unauthorized GetObject or DeleteObject operations.
   - Cross-reference access logs to identify whether compromised IAM accounts, roles, or policies were exploited.

2. **Immediate Action**s:

   - If the S3 bucket becomes compromised, modify S3 bucket policies to deny public access temporarily.
   - Immediately deactivate IAM keys, console passwords, and access tokens for accounts suspected of compromise.
   - Rotate all IAM access credentials and require MFA for any new or reissued credentials.
   - Use deny statements in IAM policies or bucket policies to explicitly restrict access for a particular user, role, or source IP.

II. Data Recovery and Backup Strategy

1. **Restore Data**:

- Use encrypted backups from S3 bucket versioning or snapshots to restore affected data.
- Verify data integrity using checksum comparisons to confirm backups are uncompromised.

2. **Rebuild Infrastructure**
   - Reconfigure IAM roles, security groups, and bucket policies to remove vulnerabilities.
3. **Mitigate Backup Risks**
   - Perform an integrity scan of backups to ensure no malicious code or tampered data is stored.
   - Rotate encryption keys for sensitive backup data and ensure backup storage follows the principle of least privilege.

III. Forensic Analysis and Future Improvements

1. Analysis:
   - Investigate logs in CloudTrail to identify the originator of the attack.
   - Conduct a post-mortem study to assess existing vulnerabilities within the architectures.
   - Track event timestamps to reconstruct the sequence of the breach.
   - Analyze any modified files, uploaded scripts, or unusual configurations in the S3 bucket for malicious payloads.
2. Improvement:
   - Introduce recognized intrusion prevention hardware software.

- Enable GuardDuty to identify future threats using machine learning and anomaly detection.
- Introduce additional encryption layers of sensitive data to the users.
- Conduct regular incident response drills to test detection and recovery processes.
- Use AWS Config to track and audit resource configuration changes in real-time.
- Finally, engage with AWS Support to assist in identifying advanced threats or root causes.

E. References

Duckett, J. (2011). *HTML and CSS: Design and build websites*. Wiley.

Duckett, J. (2014). *JavaScript and jQuery: Interactive front-end web development*. Wiley.

AWS Academy. (n.d.). *AWS Academy Cloud Foundations [Course 94879]*. Accessed via Canvas.

AWS Academy. (n.d.). *AWS Academy Cloud Operations [Course 94887]*. Accessed via Canvas.

AWS Academy. (n.d.). *AWS Academy Cloud Security Foundations [Course 94881].* Accessed via Canvas.

PayScale. (n.d.). *Salary data and career research*. Retrieved November 29, 2024, from https://www.payscale.com

Amazon Web Services. (n.d.). *AWS documentation*. Retrieved November 29, 2024, from https://docs.aws.amazon.com/