Final Project Abstract: Cloud Web App Hosting with PCI-DSS Security

Background

Established in 2023, BubbleTech Teas (BTT) is an innovative subsidiary of Bubble Tea Worldwide, known for its unique blend of traditional and modern flavors. BubbleTech Tea aims to build and deploy e-commerce web applications serving consumers and businesses with ordering and payment capabilities. With that aim, BubbleTech Tea's mission is to expand its offerings beyond physical retail locations and be a digitally-focused company revolutionizing the bubble tea experience through technology, offering convenience, variety, and a personalized experience for each customer.

Objective

The project aims to design a proof of concept for an AWS-hosted web application using iFrame or API services from payment processing service providers such as Stripe. It will focus on modeling cloud security and PCI-DSS compliance through a proof of concept using a security-focused AWS architecture that will host a web application, including processing endpoint calls via iFrame or API, leveraging Stripe's test endpoint, while using a secure database for logging payment transactions for demonstration.

Geography & Serving Areas

For the initial proof of concept, BubbleTech Tea will focus on the United States-based market. Once the final architecture is confirmed from the proof of concept, BBT will build and deploy a web app and public cloud infrastructure for a United States production launch. Future prospects will include Canadian and European markets, each with its own data hosting, privacy, and security requirements, evolving from the US model.

Payment Processing

In collaboration with Stripe's test environment, BubbleTech Teas will demonstrate a secure transaction processing environment. The system will use Stripe's test endpoints and credentials to simulate real-world payment scenarios without actual financial transactions. Upon completion of a test transaction, a transaction ID is generated by Stripe and securely stored in an AWS DynamoDB database. This approach ensures operational integrity and transaction traceability.

Compliance & Security

As a company handling sensitive financial data, BubbleTech Teas adheres strictly to PCI-DSS standards. The AWS infrastructure is configured to comply with regulatory requirements, ensuring data privacy and security. The company employs state-of-the-art encryption, network security protocols, and continuous monitoring systems to safeguard customer information. Initially deployed in the United States, BBT's AWS infrastructure is designed to be adaptable to different regulatory and commercial environments. Plans for future expansion include adapting to Canadian and European markets and furthering compliance with local data protection and privacy laws such as PIPEDA (Canada) and GDPR (Europe).

Security challenges of hosting web applications that process payments:

- **Encryption:**
    - Data in Flight: Systems processing sensitive information like credit cards are vulnerable targets for breaches; data must be isolated and stored securely.
    - Data at Rest: Data transmissions between credit card payment processors and web applications are vulnerable without proper encryption controls implemented.
- **iFrames:**
    - Injection Attacks: Also known as Cross-Site Scripting (XSS), can make iFrames vulnerable, where scripts get injected into the websites used by consumers.
    - Error Handling: Building web applications that do not enforce the parameterization of inputs can result in security exposure through error handling attacks.

Problem Domain

Any solution involving processing credit cards is required to adhere to Payment Card Industry Data Security Standard (PCI-DSS) requirements and include a process to perform attestation annually. There is a set of PCI-DSS security standards designed to ensure that all companies that accept, process, store, or transmit credit card information maintain a secure environment. The PCI-DSS standards are broken down into 12 essential requirements:

- **Requirement 1:** Install and maintain a firewall configuration.
- **Requirement 2:** Do not use vendor-supplied defaults for system passwords and other security parameters.

- **Requirement 3:** Protect stored cardholder data.
- **Requirement 4:** Encrypt transmission of cardholder data across open, public networks.
- **Requirement 5:** Protect all systems against malware and regularly update anti-virus software or programs.
- **Requirement 6:** Develop and maintain secure systems and applications.
  Requirement 7: Restrict access to cardholder data by business need-to-know.
- **Requirement 8:** Identify and authenticate access to system components.
- **Requirement 9:** Restrict physical access to cardholder data.
- **Requirement 10:** Track and monitor all access to network resources and cardholder data.
- **Requirement 11:** Regularly test security systems and processes.
- **Requirement 12:** Maintain a policy that addresses information security for all personnel.

Scope of work

This project's scope involves developing a secure and scalable AWS-hosted web application that integrates Stripe payment processing via iFrames or APIs. It aims to reinforce security against common threats like data breaches and XSS, uphold PCI-DSS and privacy standards like CPCA, and ensure cross-platform compatibility. This project will tackle technical complexities through methodical configuration, environment integrity, and security controls. T

Project Requirements

This Proof of Concept (PoC) has many requirements necessary to secure environments in public cloud-hosted applications to protect enterprises and public consumers from credit card-based security attacks:

- **Infrastructure Requirements:**
  - **AWS Services Integration:** Utilize AWS services like EC2 for web server hosting, S3 for storage, and DynamoDB for database management.
  - **Scalable Architecture:** Design the infrastructure to be scalable for future expansion into Canadian and European markets.

  - **Network Security:** Implement VPC, security groups, and network access control lists (ACLs) to secure the network layer.
- **Data Encryption:**
  - **In-Transit and At-Rest Encryption**: Ensure all data, especially transaction and payment details, are encrypted both in transit (using SSL/TLS) and at rest (using AWS encryption services like KMS).
  - **Encryption Key Management**: Implement AWS Key Management Service (KMS) for managing encryption keys securely.
- **Access Management:**
  - **IAM Policies and Roles**: Use AWS Identity and Access Management (IAM) to define roles and policies, granting the least privilege necessary to perform a task.
  - **Multi-Factor Authentication (MFA)**: Enforce MFA for enhanced security on sensitive accounts.
- **Monitoring and Logging:**
  - **AWS CloudWatch**: Implement CloudWatch to monitor the performance and health of AWS resources.
  - **AWS CloudTrail**: Use CloudTrail to log and track user activity and API usage.
  - **Audit Trails**: Maintain comprehensive audit trails for compliance and investigative purposes.
- **Privacy Considerations:**
  - **Data Minimization**: Collect only the data necessary for processing transactions.
  - **User Consent and Transparency**: Implement clear policies for user consent and data usage transparency, especially for future GDPR compliance in European markets.
- **Compliance with PCI-DSS and Other Regulations:**
  - **PCI-DSS Compliance**: Ensure all components adhere to PCI-DSS requirements for handling cardholder data.
  - **Regular Compliance Audits**: Conduct regular audits to ensure ongoing compliance with PCI-DSS, GDPR (for future European expansion), and other relevant regulations.

- **AWS Compliance Programs**: Leverage AWS programs and tools designed to help meet compliance requirements.

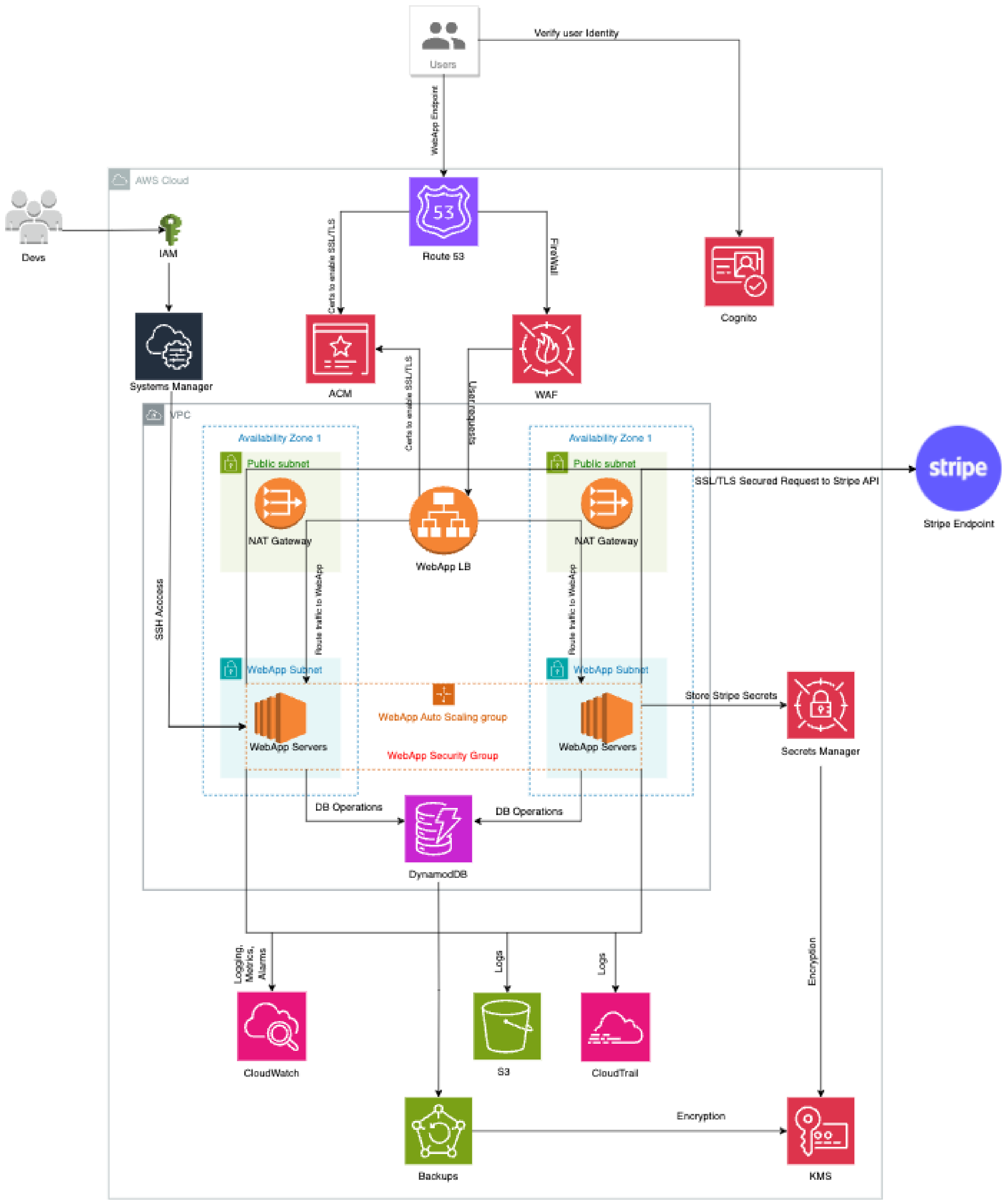
- **Web Presence and Payment Gateway Integration:**
  - **Secure iFrame Implementation**: Use a secure iFrame for Stripe's payment test environment, ensuring isolation from the main application.
  - **Stripe Integration**: Safely integrate with Stripe for payment processing, ensuring secure handling of transaction data.
- **Transaction Processing and Data Handling:**
  - **Secure Transaction Processing**: Implement secure processes for handling and storing transaction IDs and related data in DynamoDB.
  - **Data Retention Policies**: Define and implement data retention policies compliant with PCI-DSS and other relevant regulations.
- **Disaster Recovery and Business Continuity:**
  - **Backup and Recovery Plans**: Implement strategies for data backup and recovery to ensure business continuity.
  - **High Availability Setup**: Design the system for high availability to minimize downtime.

Proposed Architecture

- **Isolation:**
  - **VPC:** Configure the VPC for network segmentation to isolate sensitive components, notably the Cardholder Data Environment (CDE).
  - **Security Groups and ACLs:** Regularly audit Security Groups and ACLs to ensure they enforce strict access controls in line with PCI-DSS standards.
  - **Systems Manager and IAM**: Grant access to servers via Systems Manager and control access using IAM roles and policies
  - **AWS WAF:** Use AWS WAF with TLS 1.2 on port 443 to protect public endpoints.
- **Data Protection:**
  - **Dynamo Database:** Utilize encryption at rest for Dynamo databases to protect sensitive data in compliance with PCI-DSS.

- **Access Control:**
    - **AWS Cognito:** Leverage AWS Cognito for managing user authentication and access, ensuring compliance with strong authentication requirements.
- **Monitoring and Logging:**
    - **CloudWatch:** Implement comprehensive monitoring for system performance and security-related anomalies; set up alerts for potential compliance violations.
    - **CloudTrail:** Activate CloudTrail across all AWS accounts to log and audit actions. This meets the PCI-DSS tracking and monitoring requirements.
- **Configuration Management:**
    - **Terraform:** Manage infrastructure as code using Terraform to ensure consistent and compliant environment provisioning. This adheres to PCI-DSS guidelines for system development and maintenance.
- **High Availability and Fault Tolerance:**
    - **Elastic Load Balancing (ELB) and Auto Scaling:** Configure ELB and Auto Scaling to maintain service availability, a key aspect of PCI-DSS compliance.
    - **Multi-AZ Deployments for DynamoDB:** Deploy DynamoDB across multiple Availability Zones to ensure data redundancy and system resilience as per PCI-DSS.
- **Business Continuity and Disaster Recovery:**
    - **AWS S3 Buckets:** Use S3 buckets for storing logs, DB snapshots, replicas, and instances.
    - **Amazon DynamoDB Multi-AZ Deployments and S3 Cross-Region Replication:** Implement these strategies to comply with PCI-DSS backup and recovery requirements.
    - **AWS Backup:** Automate backup processes using AWS Backup, ensuring regular testing and restoration capabilities.

- **Disaster Recovery Plan:** Develop and periodically test the disaster recovery plan to meet PCI-DSS requirements for incident response



and recovery.

Deployment Strategy

Bubble Tea Co.'s deployment strategy will center on a robust AWS infrastructure, featuring a segmented VPC with enhanced security through ACLs, Security Groups, and a bastion host for secure access, while AWS WAF will protect public endpoints. Encryption at rest for databases and user identity management via AWS Cognito will ensure PCI-DSS compliance. Monitoring with CloudWatch and CloudTrail, alongside configuration management with Terraform, will maintain a secure and compliant environment. Elastic Load Balancing, Auto Scaling, and multi-AZ deployments for DynamoDB will deliver high availability and fault tolerance. S3 buckets for storage, AWS Backup for data protection, and a comprehensive disaster recovery plan will underpin business continuity. Further, EC2 integration, IAM policies with MFA, and adherence to privacy standards will uphold access control and data integrity, with continuous PCI-DSS compliance ensured through regular audits. Finally, Stripe's payment processing will be securely integrated via a protected iFrame, and transaction IDs will be reliably stored in DynamoDB.

Expected Results & Outcomes

The outcome will be a security-based proof-of-concept, highlighting secure practices, effective access control, and vigilant monitoring to safeguard transactions. Results with this implementation of a secure public cloud architecture in this project, which involves developing a secure AWS-hosted web application with Stripe payment integration, has the following critical result criteria as follows:

- **Access Control** - Strict access control based on roles
- **Monitoring** - Continuous monitoring and log generation
- **Configuration Management** - Persistent configuration, versioning, and orchestration
- **Secure environment** - Security control best practices using OWASP and CIS for AWS public cloud
- **Compliance** - PCI-DSS adherence via secure configuration