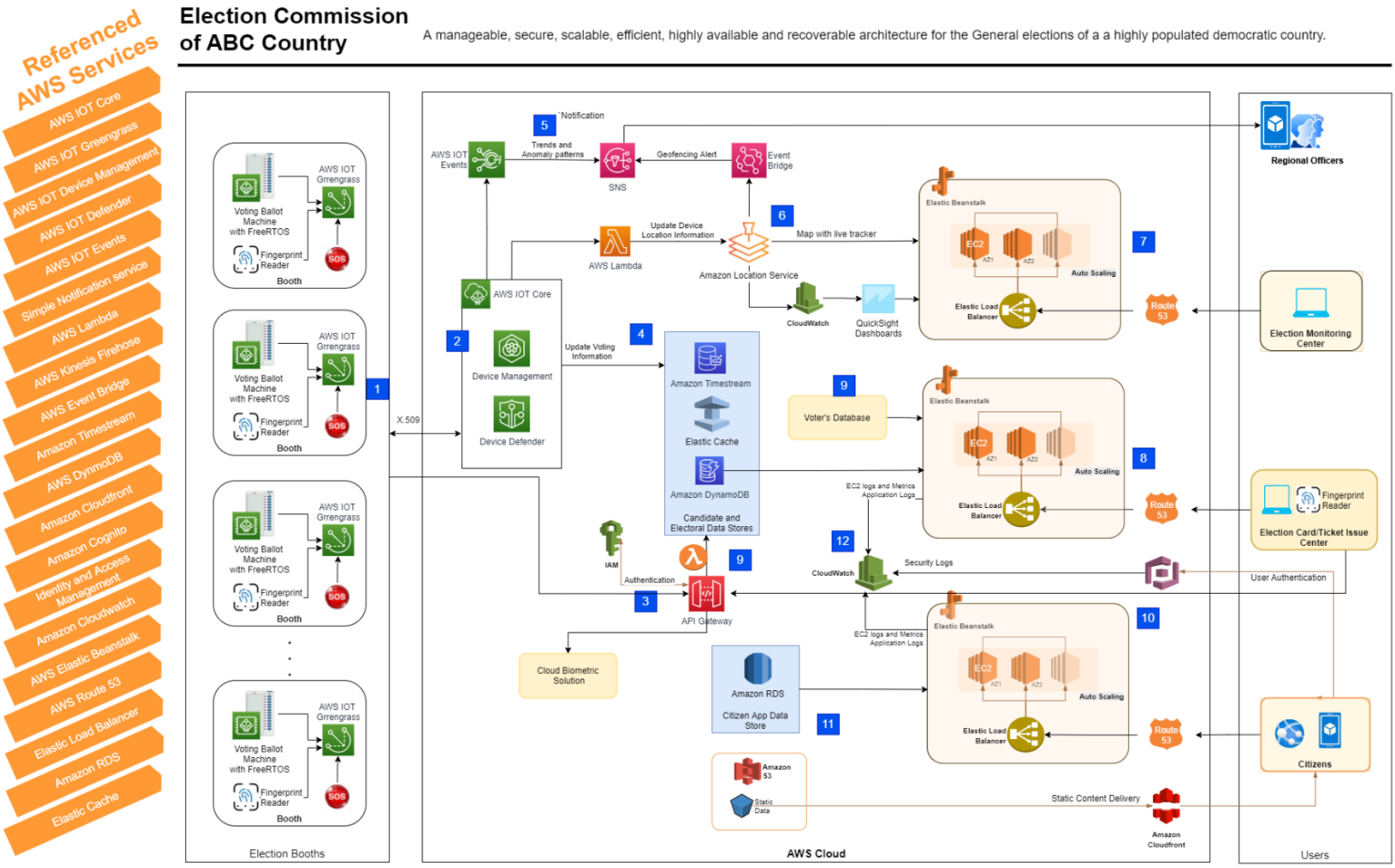
AWS Architecture Design – Assignment RFPs (1+2.A)

SOLUTION DOCUMENT

Varun Kumar

June 20, 2021



1.1 - Architecture Design

Requirements:

- Setting up candidate and electoral data management solution.
- Issue Election cards and tickets.
- Setting up systems to conduct election on a single day across country.
- Scaling Infra to meet the demand during election week.
- Ability to easily deploy the Blueprint Infra stack for other local elections.
- Prevent forgery, booth capturing and false/repeat voting.
- Integration of all Booths with central system for online and secure results transfer.
- Real-Time tracking of Voting machine's GPS location on country map.
- Real time Notifications to Regional Officers on trends and Anomaly patterns.
- Citizen facing web and mobile application

Assumptions:

1. The country already has a central database and APIs for the Voters Data along with photographs to be verified during the Elections Card/ticket issue.
2. The voters can vote from anywhere in the country and not just in their constituency.
3. The country has more than one AWS region available.
4. The Cloud Biometric API (AWS marketplace) provides capability to convert and store fingerprints at the required scale (with multiple multi-region instances).

Solution Flow:

1. Electronic Voting Machines are deployed in the Election booth without internet connectivity, and only allowed outbound communication to edge gateway running AWS IoT Greengrass to connect and publish data using open standard MQTT protocol.
2. AWS IoT Core to be used to manage devices, connect to AWS cloud, update over-the-air (OTA), and secure the devices.
3. The fingerprint-based verification is done for each voter using the Biometric API system and a look up is performed in ElasticCache on the AWS cloud against the voting ticket and/or Voter Id to prevent repeat voting. Lambda function will be called to update the Voting Status of the Voter in ElasticCache and DynamoDB. Amazon API Gateway is the single secure endpoint invoked from AWS IoT Greengrass for all API calls.
4. Once the voter is validated, he/she will be allowed to enter the Voting compartment and vote for their chosen representative by pressing the voting button. All the voting will be sent to the AWS IOT core sent by AWS IoT Greengrass which will further be stored into amazon Timestream database through IOT rule. If there is some issue with connectivity to the IOT core, the voting data will be stored in the AWS IoT Greengrass and will get sync once the connectivity is restored.
5. AWS IoT Device Defender will continuously monitor security metrics from devices and AWS IoT Core for deviations from the expected behaviors for each device. AWS IoT Device Defender will send alarm to the Amazon SNS, which will further send the notification to the Regional Officer and Monitoring Center to take any action.
6. Amazon Location service will help in tracking real-time tracking of Voting machine's GPS location on country map. Location Tracker maintains device position history, and the associated geofence collection publishes enter/exit events to the default event bus in EventBridge which further use SNS to send the event to the Regional Officer. The IoT Rule in AWS IoT Core invokes the Lambda function. (No out-of-the-box IoT rule currently exists for Amazon Location Service.) to send location to the Location Tracker.
7. Election Monitoring Center will access the application through Amazon Route 53 and Elastic load balancer configured in Elastic Beanstalk, authorized through AWS Identity and Access Management (IAM). ELB will send the request to the healthy and available EC2 instances in Elastic Beanstalk - specific environment (Dev/Test or Production). The monitoring application will have embedded map from Location service and Quicksight dashboards of CloudWatch logs.
8. Election Card/Ticket Issue center will also access the application same way as Election Monitoring Center. The application will store the information of the Voter and the tickets issued in the DynamoDB.
9. Election Card/Ticket Issue application will access the Biometric API using Amazon API Gateway for the fingerprint storage and access against the Voter Information from Country's Voter's Database.
10. Citizens will access (sign-in) their mobile app and website through Cognito. Cognito will also enable sync user specific data across multiple devices.
11. The mobile-backend application residing on Beanstalk will process the request and store the data on Amazon RDS. Static data (such as images and scripts) to Mobile app will be delivered by CloudFront with low latency and high transfer speed for better user experience.
12. All the logs and metrics from various AWS resources will be stored to AWS CloudWatch for monitoring.

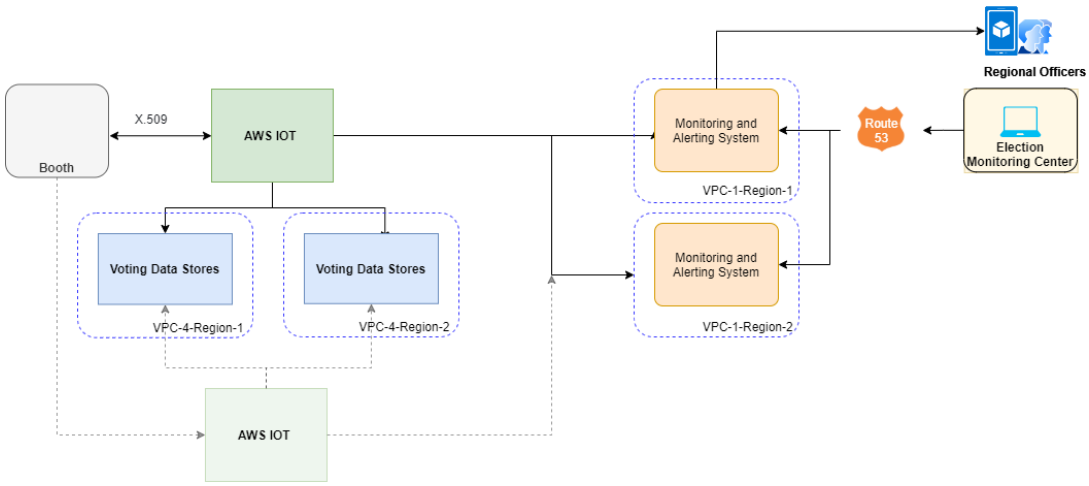
Key Requirements/ Considerations

- **Security**
    - **Prevent forgery, booth capturing and false/repeat voting.**
        - Biometric (fingerprints) verification to be used on the booths to allow only registered voters and prevent false/repeat voting.
        - AWS IOT Device defender to be used to detect identify security issues and anomalies in the voting patterns and sending the notification immediately to the Regional Officers.
        - SOS option with the booth staff and in-charge to intimate any booth capture or forgery.
        - Blocking of machine from the Cloud to stop voting until action taken by the local authority in case of any forgery alert.

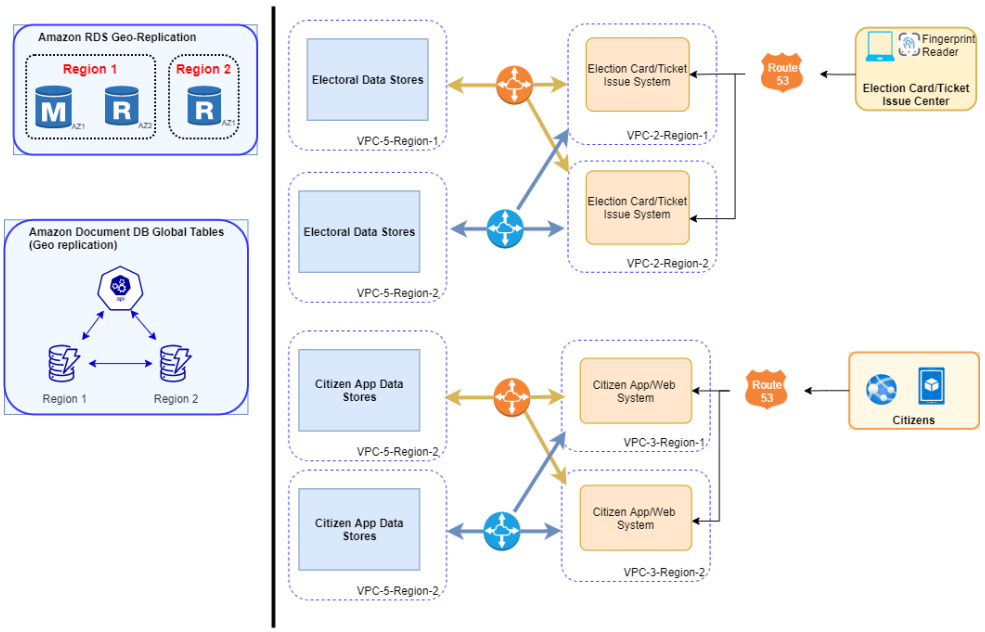
- **Security of data (at rest and in transit)**
  - Devices to connect to AWS IoT using X.509 certificates for secure transfer and all traffic to and from AWS IoT is sent securely over TLS 1.2.
  - AWS IoT Greengrass authenticates and encrypts device data for both local and cloud communications so that data is never exchanged between devices and the cloud without proven identity.
  - Security at rest is implemented through enabling encryption with keys in KMS for Timestream, DynamoDB, ElasticCache and RDS.

- **Application Security.**
  - AWS IoT authentication takes care of authenticating devices, securely ingesting device data, and granting or denying access permissions specified for the devices using AWS IoT policies.
  - Amazon Cognito provides highly scalable user directory for millions of users. Cognito will create and authenticate unique identities for consumers and service providers with identity providers as well as sync user specific data across multiple devices. Amazon Cognito also supports public identity providers—Amazon, Facebook, Google, and SAML identity provider, if required.
  - API calls to be authenticated using IAM users and groups.
  - AWS IoT Device Defender to continuously audits Device configurations to make sure of any tampering.

- **Network and Infrastructure Security.**
  - The applications and database are placed in their specific VPC with VPC peering done for the communication required.
  - AWS IoT Device Defender defines how many ports are open on the device, who the device can talk to, where it is connecting from, and how much data it sends or receives.
  - Access to the AWS resources is managed through IAM and Security groups are defined for the EC2 instances for controlled access.
  - Amazon Web Services is responsible for protecting the global infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure comprises the hardware, software, networking, and facilities that run AWS services.
  - The applications are deployed, and data is also replicated in multiple regions for high availability and Disaster recovery.



2.1 – Electoral Solution High Level Network

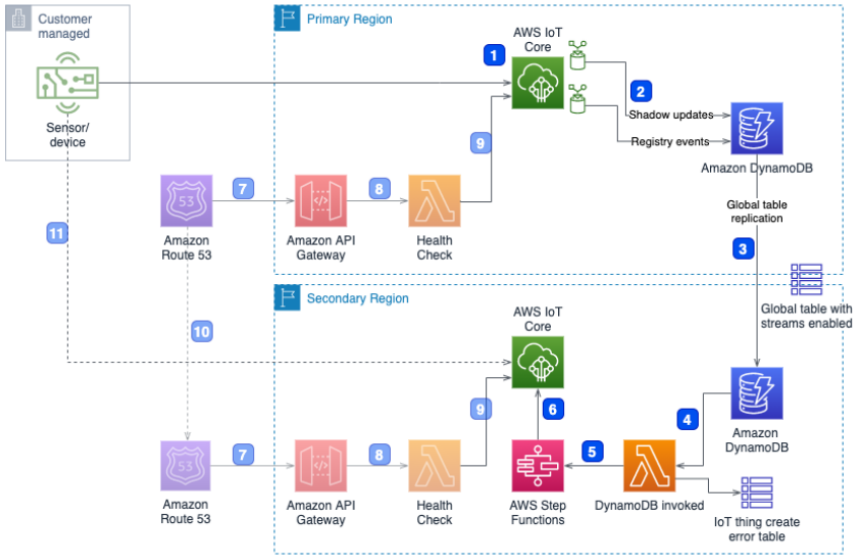


2.2 – Election Card/ticket issue and Citizen app High Level Network (right) and data Geo-replication (left).

- **High performance and throughput on all layers: web, apps and database.**
  - AWS IOT Core Message Broker is a high throughput pub/sub message broker that transmits messages to and from all IoT devices with low latency.
  - The application is deployed on multiple regions and database has replicas and shards for the high performance of the applications.
  - AWS IoT Greengrass also lets connected devices operate even with intermittent connectivity to the cloud. Devices can collect, process, and export data streams, even when they are offline. When the device reconnects, AWS IoT Greengrass synchronizes the data on the device with cloud services such as Amazon Simple Storage Service (Amazon S3), Amazon Kinesis, AWS IoT Core, or AWS IoT Analytics.
  - Global tables in Amazon DynamoDB provides multi-region, and multi-active database to deliver high performance for massively scaled applications. Global tables replicate DynamoDB tables automatically across AWS Regions.
  - With the help of RDS Read replica in multi-AZ , high throughput of the system can be achieved. Moreover, with the help of RDS performance insight the performance of database can be monitored and improved.
  - Global Datastore for Redis provides fully managed cross-Region replication for ElasticCache for Redis to enable low-latency reads and high availability across AWS Regions. Each shard in a replication group has a single read/write primary node and up to 5 read-only replica nodes.

- **Scaling Infra to meet the demand during election week.**
  - AWS Elastic Beanstalk leverages Elastic Load Balancing and AWS Auto Scaling to automatically scale the application in and out based on load.
  - RDS instances can be easily scaled horizontally or vertically depending upon the load.
  - S3 provides indefinite scalability and high availability automatically.
  - AWS Dynamo DB uses AWS Application Auto Scaling and allows to create a *scaling policy* which specifies to scale read capacity or write capacity (or both), and the minimum and maximum provisioned capacity unit settings for the table or index.
  - Amazon Timestream is serverless and does not required capacity to provision. Amazon Timestream scales to process trillions of events and millions of queries per day. It automatically scales to adjust capacity.

- **A self-healing infrastructure that recovers from any failed component.**
    - Beanstalk allows to select deployment of instances in one, more or all the availability zones (Multi-AZ) for availability.
    - With the help of cloud watch, Beanstalk monitors the health of the instances and immediately replaces the unhealthy instances with healthy instances.
    - An Amazon API Gateway resource, which calls a Lambda function is used for the health check of AWS IOT core. This Lambda function is configured as a device in IoT Core. When invoked, the Lambda function connects to IoT Core, and subscribes to a topic and publishes a configured number of messages. The Lambda function expects to receive the same number of messages to the topic it has subscribed to.
    - Amazon Route 53 health checks calls the API Gateway resource to test the MQTT message broker.



2.3 – AWS IOT core BCDR.

    - Amazon DynamoDB, RDS and Elastic Cache have multi-zone and geo-replication enabled which automatically start serving traffic from the replica if one of the nodes fails.
    - For the voting data, the IOT core stores the data in 2 Timestream database in different regions through IOT core rules.
    - In case of the outage of the IOT Core, the IOT Greengrass automatically starts communicating to secondary IOT Core which is shown in figure 2.3.

- **Ability to easily deploy the Blueprint Infra stack for other local elections.**
    - AWS CloudFormation template describes AWS resources to be launched and configured together as a stack. A template can be used to create, update, and delete an entire stack as a single unit. You can manage and provision stacks across multiple AWS accounts and AWS Regions providing a fast and repeatable method for replicating the process.
    - AWS CodePipeline lets you select AWS CloudFormation as a deployment action in any stage of your pipeline and allows the ability to spin up, correctly size, and provision new environments without excessive workloads and extended timeframes.
    - Beanstalk provides option to create multiple environments with RDS, Elastic load balancer, Autoscaling etc. for dev, test and production.
    - Beanstalk with the help of code pipeline supports multiple deployment policies for minimal downtime– all at once, rolling, rolling with an additional batch, immutable.