Proof of Concept

for

Drone Shuttles Ltd.

Document Author: Pradeep Chopra

Document Version: 1.0

Table of Contents

1 Introduction

This solution is designed as a Proof of Concept for Drone Shuttles Ltd. The solution takes into account all the major technical, functional and operational requirements mentioned in the requirement document "*CA Technical Assignment V.2*".

2 Requirement

The solution is designed considering following high level requirement points:

a. To host Ghost platform in a cloud environment.
b. Scalability to meet spikes of up-to 4 times the normal traffic.
c. Georedundant solution
d. Highly available solution
e. Cover all Security aspects
f. Disaster recovery site
g. Seamless and quick deployments without downtime
h. Multiple environments to support various lifecycle of the project
i. Monitoring, Alerting and Reporting solution
j. Serverless function for posts deletion
k. User, Role management for different teams
l. Cost effective and maintainable solution

3 Proposed Solution

This section describes various aspects of the proposed solution.

3.1 Cloud Platform

Amazon Web services (AWS) cloud technology stack has been proposed & used for this solution poc.

3.2 Network details

3.2.1 Bastian Network

<table>
<tr><td>Description</td><td colspan="4">This VPC is to host Bastian/ Jump-box server. The network is designed separately and outside the actual application VPC to keep it fenced from outside direct login threats.</td></tr>
<tr><td>VPC Name</td><td colspan="4">Transit-Tst-Vpc</td></tr>
<tr><td>Network CIDR</td><td colspan="4">10.10.0.0/20</td></tr>
<tr><td>Availability Zones</td><td colspan="4">Eu-north-1a, Eu-north-1b
(For POC only 1 EC2 is created in 1 availability zone, for actual solution, at least 2 EC2s in 2 different availability zone may be put behind a load balancer)</td></tr>
<tr><td>Subnets</td><td colspan="4"><table><tr><th>Name</th><th>Type</th><th>Availability Zone</th><th>IPv4 CIDR</th></tr><tr><td>Transit-Tst-Public-Sub1</td><td>Public</td><td>eu-north-1a</td><td>10.10.1.0/24</td></tr><tr><td>Transit-Tst-Public-Sub2</td><td>Public</td><td>eu-north-1b</td><td>10.10.2.0/24</td></tr></table></td></tr>
<tr><td>Network Rules</td><td colspan="4">• Public IP assigned to EC2 (Bastian host)
• VPC has internet gateway assigned
• SSH to application private subnets via VPC peering connection with Application stack VPC</td></tr>
<tr><td>Restrictions</td><td colspan="4">Access to EC2 may be restricted from the organization network CIDRs in order to avoid unwanted SSH requests from open internet.</td></tr>
</table>

3.2.2 Application & Database Network

<table>
<tr><td>Description</td><td>This VPC is to host the application and database stack. The network is designed in a multi-tier architecture to separate, scale and control the access to various layers of the solution.</td></tr>
<tr><td>VPC Name</td><td>Drone-Tst-AppVpc</td></tr>
<tr><td>Network CIDR</td><td>192.168.0.0/20</td></tr>
<tr><td>Availability Zones</td><td>Eu-north-1a
Eu-north-1b</td></tr>
<tr><td>Subnets</td><td>
<table>
<tr><th>Name</th><th>Type</th><th>Availability Zone</th><th>IPv4 CIDR</th></tr>
<tr><td>Drone-Tst-WEB-Public-Sub1</td><td>Public</td><td>eu-north-1a</td><td>192.168.1.0/24</td></tr>
<tr><td>Drone-Tst-WEB-Public-Sub2</td><td>Public</td><td>eu-north-1b</td><td>192.168.2.0/24</td></tr>
<tr><td>Drone-Tst-APP-Private-Sub2</td><td>Private</td><td>eu-north-1a</td><td>192.168.3.0/24</td></tr>
<tr><td>Drone-Tst-APP-Private-Sub2</td><td>Private</td><td>eu-north-1b</td><td>192.168.4.0/24</td></tr>
<tr><td>Drone-Tst-DB-Private-Sub1</td><td>Private</td><td>eu-north-1a</td><td>192.168.5.0/24</td></tr>
<tr><td>Drone-Tst-DB-Private-Sub2</td><td>Private</td><td>eu-north-1b</td><td>192.168.6.0/24</td></tr>
</table>
</td></tr>
<tr><td>Network Rules</td><td>

Public (Web) subnets assigned to Application Load Balancer, to take HTTP/HTTPS requests from internet users
Private subnets (App) assigned to application layer, accepting traffic only on port 443 from web public layer via Application load balancer.
Private subnets (DB) assigned to RDS MySQL database and listens only from private application layer on port 3306.
SSH to App private subnets via VPC peering connection with Application stack VPC
Internet access to Private app layer via NAT gateway

</td></tr>
<tr><td>Restrictions</td><td>Access to application EC2 instances restricted only through Bastian network layer.</td></tr>
</table>

3.3 High Level Design

Following Is a high-level solution design consisting of 4 layers:

a. **Web Layer:** Public Internet facing layer
b. **Application Layer**: Private layer hosting application EC2 instances, Access to the servers inside this layer is restricted to outside world, only following access is opened:
    i. HTTP/HTTPS via Application Load Balancer
    ii. SSH access via Bastian Network
    iii. Connectivity between Private Database layer
    iv. Access to AWS resources like S3, CloudWatch, etc. through IAM policies
c. Database Layer: Private layer hosting RDS database, access to this layer is strictly restricted to only Application Layer.
d. Bastian layer: Public layer hosting Bastian EC2 instance, SSH is enabled on Bastian server from Internet but Restricted to IP Network of Drone Shuttles Ltd. Organization.

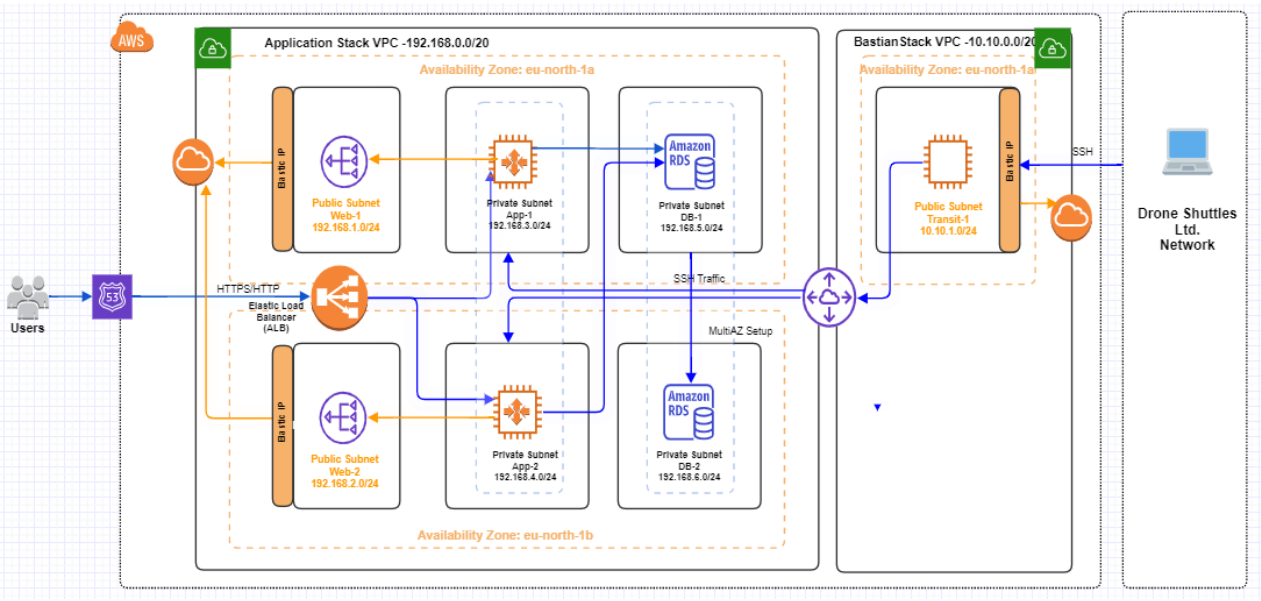

Diagram-1: Solution design

3.4 Database

AWS offers Database managed service called RDS which has been assumed for this solution. Ghost framework requires MySQL server for proper functioning. As part of POC, MySQL has been created in a single availability Zone with following parameters:

Parameter	**Value**
DatabaseInstanceIdentifier	myDB
DatabaseName	ghostdb
DatabaseUser	Admin
DatabaseBackupRetentionPeriod	7 days
DatabaseAllocatedStorage	20GB
DatabaseInstanceClass	db.t3.micro
MultiAZDatabase	False

Alternative to RDS, MySQL may also be hosted within Database Layer on EC2 instances. But there will be additional operations and maintenance overhead with that solution.

3.5 Security

Following security measures have been considered in the solution:

1. AWS console login:
    - Enable Multifactor Authentication (MFA) for all accounts.
    - Strong password policy comprising minimum length, combination of alphanumeric and special characters.
    - Enable password expiry rules

2. EC2 instances:
    - Using custom light weight ubuntu AMI with updated security and OS patches
    - Disable unwanted open ports in AMI
    - Enable Secure Key + MFA based login for Bastian SSH login.
    - Enable login to EC2 instances in private network only via Bastian server using public, private key pair.

3. Security in transit (Web):
    - Use HTTPS protocol with TLS certificates on web servers
    - Load balancer to route HTTP requests to HTTPS

4. Multi-Tier Architecture
    - Use Multitier architecture for security fencing between Public, Private layers.
    - Allow HTTPS traffic only through secure Application load balancer on specific port
    - Disable all unused ports in public and private subnets using security groups
    - Enable Database access on port 3306 only though application private layer
    - Use NAT Gateway for outgoing traffic from private subnets

5. Identity and Access Management (IAM)
    - Create IAM groups for different teams and follow the principal of minimal privileges to the group.
    - Users must belong to at least 1 group
    - Create IAM role for EC2 to use different AWS services like S3, CloudWatch, etc.

6. S3 security
    - Block public access of the buckets
    - Enable versioning
    - Encrypt buckets data

Additional Service offerings:

1. Web Application Firewall-WAF (Optional)
   This is an optional service in this solution but good to have for critical web applications.

Amazon WAF is a managed service which acts as a filter before the Load Balancer and protects the application against potential threats, web exploits like SQL injections, Cross site scripting etc.

2. AWS CloudTrail (Optional)
   This is also an optional service but very useful in risk auditing, compliance, governance of your AWS account. CloudTrail records every action in the form of events done in AWS account via console, cli or via API's.

3. AWS Shield (Optional)
   AWS shield protects from Distributed Denial of Service (DDoS). This is also an important offering to consider in case the web application, website is a business critical. Implementation of AWS shield along with combination of CloudFront, ELB, etc. may provide complete protection from all layers of DDOS threats.

4. AWS GuardDuty (Optional)
   GuardDuty can detect signs of account compromise (including EC2 instances, S3 data), detects threats identifiers such as AWS resource access from an unusual location, unusual application programming interface (API) calls, calls from malicious IP address, etc. and alarms appropriately.

3.6 Scalability

Scalability is an important aspect of a production application stack and needs to be designed and planned properly in case of applications which may receive sudden surge of traffic. Amazon provides a lot of flexibility and elasticity and supports scalability at all levels.

For our use cases the scalability has been planned considering various layers of solutions.

3.6.1 Route53

In case of a web application the request first lands at DNS server for name resolution, we are using AWS native service Route53 as DNS service which is a managed and highly scalable service.

3.6.2 Load Balancer

Amazon offers a highly scalable and secure Load balancing service named Elastic Load Balancer (ELB), we have considered ELB as Application Load Balancers which is capable of handling any level of spikes in the incoming requests.

3.6.3 Application stack

Amazon provides flexibility to plan Vertical as well as Horizontal scaling to any level.

User needs to plan the application-level scaling mainly considering 2 points:

a. Select right EC2 instance type to fit into the application requirements with optimal CPU, Memory and network capacities. It is important to strike a balance between capability and cost. For our POC, considering low demand of default Ghost application, we have used t3.micro instance which comes with 2 vCPUs (1 Core), 1 Gib of RAM and 5Gib of network performance.

b. Horizontal scaling: Applications have been placed into an autoscaling group with minimum 2 EC2 instances and maximum 8 EC2 instances (4 in each availability zones eu-north-1a, eu-north-1b) to

provide a scalability up to 4 times. AWS autoscaling refers to various consumptions matrices and triggers the scale-up or scale-down rules based on the autoscaling configuration. In our use case we have considered CPU consumption of >80% to trigger scale-up rule and CPU consumption of <40% to trigger scale-down rule.

3.6.4 Database

Amazon offers RDS as secure with scalable storage. User needs to select optimal compute and memory combination; AWS takes care of storage scalability.

3.6.5 NAT Gateway

NAT gateway is a managed and highly scalable offering by AWS.

3.6.6 Internet Gateway

Internet gateway is a managed and highly scalable offering by AWS.

3.7 High availability

Application stack has been designed for high availability. While high availability of underlying infrastructure services (eg. ELB, Route53, NAT/Internet GWs, etc.) is guaranteed by AWS. Application stack has been designed in a way that AWS autoscaling service continuously monitors the stack for increased or decreased resource utilization and controls the scaling of the solution to maintain proper availability and functioning of the solution. On low utilization, autoscaling makes sure a minimum setup is always available.

3.8 Geo redundancy

Entire solution is divided into 2 geographically separate availability zone eu-north-1a, eu-north-1b in Stockholm region.

3.9 Environments

An SDLC generally requires 3 environments viz, Test, Preprod, Production.

For this POC exercise Test environment has been created using CloudFormation templates, Creation of other environments are also supported in the CloudFormation templates and can be achieved in few minutes by providing required parameters during creation time.

3.10 Automation

Infrastructure and Application deployment automation has achieved using CloudFormation templates. There are 4 separate templates created for following purposes:

a. Template-1: Bastian-Stack-CFT-1.1.yml
   It creates the Bastian VPC and Bastian server for SSH access

b. NetworkAndIAM-Stack-CFT-1.1.yml
   It creates main application VPC, network configuration, IAM settings.

c. RDS-Stack-CFT-1.yml

It creates RDS MySQL database for ghost application.

d. App-Stack-CFT-1.2
It creates application stack, EC2, ELB, Autoscaling, deployment of binaries and configuration, etc.

In the proposed solution the development team needs to place the configuration files and binaries in a predefined S3 bucket and trigger the update using Stack update feature, CloudFormation identifies the changes required in the stack and performs rolling updates accordingly. There is no downtime to service in case of software upgrades.

A complete CI/CD pipelines may also be designed by using external Integration tools like Jenkins, Or by using AWS native tools like Code Commit, Code Build, Code Pipeline.

3.11 User Management

a. AWS console users
Console users may be created in the AWS IAM section, for demonstration purpose 1 user has been created in the exercise. For large teams, organization Active Directory or LDAP tools may be integrated to with AWS using IAM Federation.

b. Linux shell users
Operating System users and groups may be managed using AWS service Fleet Manager, the service requires installation of SSM agent in EC2 machines.

For this exercise default ubuntu user was used with public/private key pair.

c. User Groups
Different user roles must be created for different user teams. For demonstration only admin group is created in this exercise but similarly other groups for teams like operations, development, audit, security, etc may be created with required privileges in AWS console.

d. IAM roles for CLI/SDK
It is important to not hardcode personal credentials into any service or tools using AWS services. In order to protect against stealing of personal credentials, AWS provides IAM roles which may contain various policies to access different resources.
For the exercise, following Roles are created:

Role Name	**Consumer**	**Policies**	**Permissions**
DroneInstanceRole	EC2	DroneS3BucketAccess	- s3:ListBucket - s3:PutObject - s3:DeleteObject - s3:GetObject - s3:CreateBucket
	EC2	DroneCloudwatchAccess	- logs:CreateLogGroup - logs:CreateLogStream - logs:PutLogEvents - logs:DescribeLogStreams

3.12 Monitoring, Logging & Alerting

a. Cloud Watch log
   CloudWatch is a central repository for log storing, querying and reporting. Following log groups are created and log forwarding enabled for them:
   - ApacheAccessLogs
   - ApacheErrorLogs
   - GhostLogs
   - GhostErrorLogs
   - SysLogs

b. Cloud watch Alarms
   - ApacheHttpFailureAlarm
     Alarm generated if Number of http failures in Apache is greater than 100 over 5 minutes.

c. Cloud watch matrices
   - ApacheHttpFailureMetricFilter
     Matrices created to track failures in Apache server. It filters all result codes other than 200. Following filter pattern is used: [x1,x2,x3,x4,x5,x6!=200,...]

d. Notifications via SNS
   - Drone-Alarms SNS topic
     A topic for alerting is created using AWS SNS service which send out email alerts to defined email recipients.

3.13 Disaster recovery site

The solution is hosted in Stockholm region into 2 availability zones (eu-north-1a, eu-north-1b). This provides geographic redundancy to the solution.

We can similarly create another setup into another region using the same CloudFormation templates. We just need to keep copies of latest AMI's, S3 data and RDS backup sync to the Disaster recovery region.

3.14 Lambda functions

There is a requirement for functionality to delete all posts from Ghost application whenever required. This functionality may be achieved using Lambda serverless offering by AWS.

A lambda function may be written to delete the posts directly from RDS database, this requires appropriate permissions and connectivity to RDS database.

Lambda function may be scheduled to run at a defined time every day/week/month.

It may also be invoked programmatically if embedded within the application.

4 Pricing model

AWS claims a low and predictable pricing model, However cost of the solution totally depends on estimation of number of requests per month, during peak hours, during normal hours, Database record projections, Data retention policies, ingress/egress from/to the internet except for accessing AWS services, etc.

For this POC free tier resources have been used.

----End of the document