BTrust Identity Platform
Deployment & Configuration Guide

1. Product Overview

BTrust is a comprehensive, microservices-based identity verification and orchestration platform. It enables financial institutions to manage KYC, document verification, liveness checks, and authentication workflows securely within their own AWS environment.

The solution consists of:

1. **Front Services:** Client for end-users and administrators (Console, Portal, Onboarding).
2. **Platform Core:** The main orchestration engine and API gateway.
3. **AI & Identity Services:** Specialized engines for OCR, Liveness, Biometric matching, and Privacy.

Front Services

- **Console**: A Back Office interface service used for the management and ongoing operation of system processes.
- **Onboarding**: Enables the management of processes for end customers devices
- **Portal**: A document submission and accessibility for end customers.

Platform Core Components

- **Gateway**: mediate communication between system components and provide API access
- **Core**: The central component for managing system processes and data.

AI & Identity Services

- **Authorization Service:** Manages identification, authentication, authorization, and system access tokens.
- **OCR**: A service for document and ID scanning and recognition.
- **Liveness**: A service for liveness verification and authentication.
- **Video Statement**: A service for visual and voice-based declarations.
- **Face Match and Face Split**: A facial comparison service for identity verification based on end customers' images.
- **OTP**: A one-time password authentication service that sends codes via email or SMS.

2. Infrastructure Prerequisites

Before deploying the containers, you must provision the following resources. The containers are stateless and rely on these external services.

A. Network & Compute

- **VPC:** A VPC with Private and Public subnets across at least 2 Availability Zones.
- **Load Balancer:** An Internet-facing Application Load Balancer (ALB).
- **ECS Cluster:** Amazon ECS (EC2 Launch Type is **mandatory** for AI services).
- **Hardware Requirement:** The AI services (OCR, Liveness, FaceMatch) utilize AVX instructions. **You must use Intel/AMD (x86_64) instances** (e.g., C5, C6i, M5). **Do not** use Graviton (ARM) instances for these specific services.
    - *Note: Ensure your AWS Service Quotas allow for sufficient vCPU for these instance types.*

B. Storage & Databases

- **Relational Database:** Amazon RDS for MySQL (Version 8.0+).
    - **Charset:** Must be utf8mb4.
    - **Encryption:** Enabled (Storage Encrypted).
    - **Parameters:** Set slow_query_log=1, long_query_time=5.
- **NoSQL Database:** MongoDB (v4.4+). Self-hosted on EC2 or via MongoDB Atlas.
- **Caching:** Amazon ElastiCache for Redis (Cluster Mode Enabled).
- **Object Storage:** Amazon S3 Buckets. **Server-Side Encryption (SSE-KMS) is mandatory.**
- **File System (Critical):** Amazon EFS. Required for injecting configuration files and credentials.

C. Messaging

- **Streaming:** Amazon Kinesis Data Streams (Create two streams: analytics and audit).

3. Security Configuration (IAM)

The platform uses a "Tenant Isolation" model. Containers assume a specific Tenant Role to access S3.

You must create an IAM Role with the following configuration:

1. **Role Name:** storage-tenant-role-<YOUR_COMPANY_UUID>
   - *Replace <YOUR_COMPANY_UUID> with a unique ID you generate (e.g., a UUID v4).*
2. **Trust Policy:** Allow your ECS Task Role to assume this role, conditional on an External ID.
   JSON

None
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::<YOUR_ACCOUNT_ID>:role/<YOUR_ECS_TASK_ROLE>" },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": { "sts:ExternalId": "<YOUR_CHOSEN_SECRET_TOKEN>" }
      }
    }
  ]
}
```

3.
4. **Permissions:** Attach an Inline Policy allowing s3:PutObject, s3:GetObject, s3:ListBucket, and s3:DeleteObject on your data S3 bucket.

4. EFS & File Setup

Create an Amazon EFS file system and mount it temporarily to create the following directory structure:

1. **/nginx/overrides/**: Upload your SSL Certificate (cert.pem) and Private Key (key.pem).
2. **/credentials/**: Upload your Cloud Service Account JSON file (rename to credentials.json) for the Video Statement service.

5. Container Configuration Guide

Configure the services by passing the following Environment Variables in your ECS Task Definitions.

Group A: Front Services (Public Facing)

These services provide the UI and client endpoints. Map your ALB to these ports.

1. Console (Port 8010)

- *Description:* Back-office interface for management.
- **Env Vars:**
  - AUTH_URL: https://<auth-service-dns>:5800
  - CAPTCHA_ENABLED: 1 (if using Captcha)
- **Secrets:** CLIENT_ID, CLIENT_SECRET (Console credentials), FERNET_KEY.

2. Portal (Port 8020)

- *Description:* Document submission interface for end customers.
- **Env Vars:** AUTH_URL.
- **Secrets:** CLIENT_ID, CLIENT_SECRET (Portal credentials), FERNET_KEY.

3. Onboarding (Port 8050)

- *Description:* Mobile/Web onboarding interface service.
- **Env Vars:**
  - AUTH_URL: https://<auth-service-dns>:5800
  - KINESIS_STREAM_NAME: Your analytics stream.
  - IDENTITY_POOL_ID: AWS Cognito Identity Pool ID.
- **Secrets:** CLIENT_ID, CLIENT_SECRET (Onboarding credentials).

Group B: Platform Core (Sidecar Pattern)

CRITICAL: The Platform service is a multi-container task. You must define **3 containers** (nginx, core, gateway) within the **same** Task Definition.

1. Container: nginx (Port 80)

- **Mounts:** Mount EFS /nginx/overrides to /etc/nginx/overrides (Read-Only).
- **Env Vars:** SSL_CERT, SSL_KEY (PEM content or path).

2. Container: core (Port 5002 - Internal)

- **Env Vars:**
  - GATEWAY_URL: https://localhost:8040
  - DATABASE_URL: MySQL JDBC URL (jdbc:mysql://...).

3. Container: gateway (Port 5008 - Internal)

- **Env Vars:**
  - CORE_URL: http://localhost:5002
  - DB_URI (Mongo), SQL_DB_URI (MySQL).
  - X_COMPANY_ID: The UUID used in the IAM Role name (Section 3).
  - ROLE_EXTERNAL_ID: The secret token used in the IAM Trust Policy (Section 3).

Group C: AI & Identity Services

1. Liveness Cluster (Deploy as 3 separate services/tasks)

- **Liveness Main (Port 5050):**
  - INJECTION_DETECTION_SERVER_URL: https://<injection-service-dns>:8443
  - PRESENTATION_DETECTION_SERVER_URL: https://<presentation-service-dns>:8443
- **Liveness Injection (Port 8443):** Internal service.
- **Liveness Presentation (Port 8443):** Internal service.
  - IDFACE_SERVER_AVAILABLE_PIPELINES: Set to pad-r-1,dfd-2.

2. OCR (Port 5040)

- **Hardware:** Compute Optimized Intel instances only.
- **Env Vars:** DB_URI, KINESIS_STREAM_NAME.

3. FaceMatch (Port 3000)

- **Env Vars:** FACE_DB_REDIS_URI (Redis Connection String).

4. FaceSplit (Port 3030)

- *Description:* Privacy and encryption service.
- **Env Vars:**
  - DATABASE_SCHEMA: face_split (MySQL).
  - DATABASE_TABLE_NAME: part1.
  - S3_BUCKET_NAME: Name of the S3 bucket for noise data.
  - USE_HSM_MOCK: true.
- **Secrets:** DATABASE_URL, DATABASE_USERNAME, DATABASE_PASS (MySQL), S3_ACCESS_KEY, S3_SECRET_KEY.

5. Video Statement (Port 5020)

- **Mounts:** Mount EFS /credentials to /app/credentials/.
- **Env Vars:** APPLICATION_CREDENTIALS = /app/credentials/credentials.json.

6. OTP (Port 5000)

- **Env Vars:** Configure your SMS provider details (CELLACT_USER_NAME, CELLACT_PASSWORD, CELLACT_SENDER_NUMBER).

6. Traffic & Load Balancing

Configure your ALB Listener Rules to route traffic based on Host Header:

Service	Container Port	Recommended Hostname
Console	8010	console.example.com
Portal	8020	portal.example.com
Onboarding	8050	onboarding.example.com
Platform	80	api.example.com
OCR	5040	ocr.example.com
FaceSplit	3030	facesplit.example.com

7. Operations & Troubleshooting

Verification

1. **Access:** Navigate to https://console.example.com.
2. **Health Check:** Access https://api.example.com/actuator/health to verify that the Core service can communicate with downstream components (DB, Redis, AI services). A response of {"status":"UP"} indicates a healthy system.

Logging

All containers are configured to emit logs to awslogs (CloudWatch).

- **Location:** Go to AWS CloudWatch -> Log Groups.
- **Naming Convention:** Logs are typically stored under /ecs/<service-name> (e.g., /ecs/platform, /ecs/ocr).
- **Troubleshooting:** If a service fails to start, check the CloudWatch logs first. Common issues include missing EFS files (Nginx certs) or incorrect database credentials.

Backup & Recovery

- **RDS/Mongo:** It is recommended to enable automated backups/snapshots for your RDS and MongoDB instances.
- **S3:** Enable versioning on your S3 buckets to protect against accidental deletion.

Upgrading

To upgrade to a newer version of the BTrust Platform:

1. Identify the new Image Tag from the AWS Marketplace subscription.
2. Update the ECS Task Definition with the new image tag.
3. Update the ECS Service to force a new deployment. The stateless containers will be replaced with the new version.