OPENVIDU_SETUP -

What OpenVidu is, all Docker containers, SSL setup with Let's Encrypt, session and token flow, required EC2 ports, dashboard access, all management commands.

OpenVidu Setup Documentation

ChourangiHealth Video Consultation Platform

What is OpenVidu

OpenVidu v2.32 is an open-source WebRTC platform that handles all real-time video and audio communication between participants. It runs as a set of Docker containers on the server and exposes a REST API that the backend uses to create sessions and generate connection tokens.

OpenVidu Server Components

OpenVidu runs the following Docker containers together:

- **openvidu-server —** Core signaling server that manages sessions and connections
- **openvidu-proxy —** Nginx reverse proxy that handles HTTPS and routes traffic
- **openvidu-kms —** Kurento Media Server that handles WebRTC media streams
- **openvidu-coturn —** TURN and STUN server for NAT traversal so users behind firewalls can connect
- **openvidu-app —** Default OpenVidu Call UI (we disabled this to serve our own frontend)

Installation

OpenVidu v2.32 is installed on the EC2 instance at the path /opt/openvidu using the official OpenVidu installer. The installer sets up all Docker containers and creates the necessary configuration files.

To install OpenVidu on a fresh EC2 instance run:

curl https://s3-eu-west-1.amazonaws.com/aws.openvidu.io/install_openvidu_latest.sh | bash

This installs everything at /opt/openvidu.

Configuration File

The main configuration file is located at:

/opt/openvidu/.env

Key settings in this file:

```
DOMAIN_OR_PUBLIC_IP=chourangi.duckdns.org
CERTIFICATE_TYPE=letsencrypt
LETSENCRYPT_EMAIL=ashutosh@xtensible.in
OPENVIDU_SECRET=openvidu-v2-32
OPENVIDU_USERNAME=OPENVIDUAPP
```

- DOMAIN_OR_PUBLIC_IP must be set to your domain name (not raw IP) when using Let's Encrypt
- CERTIFICATE_TYPE can be selfsigned, letsencrypt, or owncert
- OPENVIDU_SECRET is the password used by the backend to authenticate with OpenVidu REST API
- OPENVIDU_USERNAME is always OPENVIDUAPP for OpenVidu v2

SSL Certificate Setup with Let's Encrypt

Browsers require HTTPS for WebRTC camera and microphone access. HTTP only works on localhost. Without SSL every user gets a security warning and must manually accept the risk before the app works.

Steps to enable Let's Encrypt SSL:

1. Register a domain pointing to your server IP (we used DuckDNS for a free subdomain)
2. Ensure ports 80 and 443 are open in your server firewall
3. Edit /opt/openvidu/.env and set:
    - DOMAIN_OR_PUBLIC_IP=yourdomain.duckdns.org
    - CERTIFICATE_TYPE=letsencrypt

    - LETSENCRYPT_EMAIL=your@email.com
4. Restart OpenVidu — it will automatically obtain the certificate from Let's Encrypt

After restart the dashboard is accessible at https://yourdomain.duckdns.org/dashboard with no browser warnings.

Disabling the Default OpenVidu Call App

By default OpenVidu serves its own video call UI at the root path. To serve our own React frontend we disabled it by editing /opt/openvidu/docker-compose.yml:

Find the line:

WITH_APP=true

Change it to:

WITH_APP=false

Then restart OpenVidu.

Serving Custom Frontend via OpenVidu Nginx

OpenVidu's Nginx container mounts the folder /opt/openvidu/custom-layout inside the container at the same path. We copy our React build files there and create a custom Nginx location config.

Copy React build files:

cp -r /path/to/frontend/build/* /opt/openvidu/custom-layout/

Create Nginx config at /opt/openvidu/custom-nginx-locations/frontend.conf:

```
location / {
    root /opt/openvidu/custom-layout;
    index index.html;
    try_files $uri $uri/ =404;
}
```

```
location /index.html {
    root /opt/openvidu/custom-layout;
}

location /api/ {
    proxy_pass http://172.17.0.1:8000;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
}
```

The IP 172.17.0.1 is the Docker bridge IP that allows the Nginx container to reach the FastAPI backend running on the host machine at port 8000.

OpenVidu Commands

All OpenVidu management commands are run from /opt/openvidu:

```
# Start OpenVidu
cd /opt/openvidu && ./openvidu start

# Stop OpenVidu
cd /opt/openvidu && ./openvidu stop

# Restart OpenVidu
cd /opt/openvidu && ./openvidu restart

# Check running containers
docker ps

# View Nginx logs
docker logs openvidu-nginx-1 --tail 50

# View OpenVidu server logs
docker logs openvidu-openvidu-server-1 --tail 50
```

How Sessions and Tokens Work

OpenVidu uses the concept of sessions and connections. A session is a virtual video room. A connection is a participant slot inside that session with a unique token.

Our application uses customSessionId equal to the Calling ID entered by the user. When doctor enters Calling ID 1 the backend creates a session with customSessionId 1. When patient also enters Calling ID 1 the backend receives a 409 response meaning the session already exists, handles it gracefully, and still generates a token for the same session. Both parties receive tokens for session 1 and connect to each other.

Session creation endpoint:

POST /openvidu/api/sessions
Body: { "customSessionId": "1" }
Auth: Basic OPENVIDUAPP:your-secret

Token generation endpoint:

POST /openvidu/api/sessions/1/connection
Body: {}
Auth: Basic OPENVIDUAPP:your-secret

Required EC2 Security Group Ports

Port	Protocol	Purpose
22	TCP	SSH access
80	TCP	HTTP and Let's Encrypt verification
443	TCP	HTTPS web traffic
4443	TCP	OpenVidu WebSocket signaling
3478	TCP/UDP	TURN and STUN server
40000-57000	UDP	WebRTC media streams
10000-20000	UDP	Additional media traffic

All ports should be open to 0.0.0.0/0 for IPv4 and ::/0 for IPv6.

Dashboard Access

OpenVidu dashboard is available at:

https://chourangi.duckdns.org/dashboard

Login with username OPENVIDUAPP and the secret configured in .env file.