Local Certificate Authority Service: Progress Documentation

Abhishek Yashwani
Under the guidance of Dr. Balwinder Sodhi

May 2025

Indian Institute of Technology, Rupnagar
Department of Computer Science and Engineering

Abstract

This document details the current progress of the Local Certificate Authority Service, an open-source web application for managing Certificate Signing Requests (CSRs) and issuing X.509 certificates. It describes the implemented features, system architecture, and limitations as of May 2025, based on the developed prototype. The project is being open-sourced to encourage community contributions.

Contents

1 Introduction

The Local Certificate Authority Service is a web-based platform designed to facilitate the management of digital certificates within an enterprise Public Key Infrastructure (PKI). The system provides a user-friendly interface for end users to submit CSRs and for administrators to approve or reject them, issuing X.509 certificates upon approval. This document outlines the capabilities implemented in the prototype as of May 2025, focusing on the Certificate Authority (CA) and Registration Authority (RA) functionalities. It is intended for contributors to the open-source repository to understand the project's current state.

2 System Overview

The Local Certificate Authority Service is a prototype implementation of a lightweight CA system, supporting:

- User registration and authentication with email verification.
- CSR generation, submission, and admin approval/rejection workflows.
- Issuance of X.509 v3 certificates signed by a CA.
- Basic revocation tracking.
- User and admin dashboards for managing CSRs and certificates.
- Email notifications for key actions.

The system is built with a Node.js/Express backend, a React frontend, and a PostgreSQL database, using OpenSSL for CSR generation and Gmail SMTP for notifications.

3 Implemented Features

This section details the functionalities implemented, aligned with the CA and RA modules.

3.1 Certificate Authority (CA) Module

The CA module handles certificate issuance and CSR processing:

- **Certificate Issuance**:
  - Issues X.509 v3 certificates for approved CSRs using the `node-forge` library.
  - Certificates include domain, issuer, serial number, and validity period (default: 1 year).
  - Certificates are stored in the `issued_certificates` table and saved as PEM files.

- **CSR Handling**:
  - Generates CSRs via OpenSSL, accepting attributes like domain, company, division, city, state, country, email, and key length (e.g., 2048 bits).
  - Stores CSRs in the `csr_requests` table with status (`pending`, `approved`, `rejected`).
- **Revocation Tracking**:
  - Tracks revoked certificates in the `revoked_certificates` table, storing certificate ID, revocation date, and reason.
- **Key Management**:
  - Uses a single CA private key and certificate (`ca-key.pem`, `ca-cert.pem`) for signing.
  - Keys are stored in the filesystem.

3.2 Registration Authority (RA) Module

The RA module manages user registration and CSR workflows:

- **User Registration**:
  - Users register via a web portal, providing username, email, and password.
  - Email verification sends a 6-digit OTP via Gmail SMTP, stored in the `users` table.
  - Supports `admin` and `client` roles.
- **CSR Submission and Approval**:
  - Users submit CSRs through the frontend, specifying CSR attributes.
  - Admins view pending CSRs, approve (triggers certificate issuance), or reject with a reason.
  - Email notifications inform users of approval or rejection.
- **User Interface**:
  - **User Dashboard**: Allows CSR submission, viewing CSRs/certificates, downloading PEM files, and displaying stats (total CSRs, pending, approved, active certificates).
  - **Admin Dashboard**: Lists pending/all CSRs, supports approval/rejection, and shows system-wide stats.

3.3 Security Features

- **Authentication**: JWT-based login with bcrypt-hashed passwords.

- **Role-Based Access Control (RBAC)**: Restricts admin functions to `admin` role.
- **Data Encryption**: HTTPS for API communication.
- **Email Security**: OTPs for signup and password reset.

4 System Architecture

The system comprises:

- **Backend**: Node.js with Express, handling APIs for user management, CSR processing, and certificate issuance. Key components:
  - `certificateController.js`: Manages CSR generation, approval, and certificate issuance.
  - `emailService.js`: Sends OTPs and notifications via `nodemailer`.
  - `opensslService.js`: Generates CSRs using OpenSSL.
- **Frontend**: React with Material-UI, providing user and admin dashboards. Uses Axios for API calls and Formik/Yup for form validation.
- **Database**: PostgreSQL (`ca_db`) with tables:
  - `users`: User data (ID, username, email, password, role).
  - `csr_requests`: CSR details (user ID, domain, status).
  - `issued_certificates`: Certificate data (user ID, CSR ID, PEM).
  - `revoked_certificates`: Revocation records.
- **External Services**:
  - Gmail SMTP for email notifications.
  - OpenSSL for CSR generation.

5 Limitations

As a prototype, the system has constraints:

- Lacks real-time certificate validation (no OCSP or CRL).
- Uses a single CA key pair with no HSM integration.
- Limited audit logging (database logs only, no tamper-evident mechanisms).
- No support for multiple PKIs or advanced certificate policies.
- Basic revocation tracking without integration with external systems.

6 Conclusion

The Local Certificate Authority Service is a functional prototype implementing core CA and RA functionalities. It supports user registration, CSR management, certificate issuance, and basic revocation tracking, with secure authentication and email notifications. While limited in scope, it serves as a foundation for further development by the open-source community.