**Secure Virtualized MP3 Media Server with Encrypted Access and Process Replication**

CS469 Distributed Systems

*Team Members***:**

Ken Sherwood, Bradley Spence, Thomas Wintenburg

*Instructor***:** Dr. Jeff Hemmes

Regis University

**Project Description Overview**

We will build a secure, distributed MP3 media server that lets clients browse and download audio files by genre, artist, or title. The system uses a concurrent server written in C running on a POSIX Linux environment (Ubuntu 22.04 LTS under WSL or a VM). All client–server communication is encrypted with SSL/TLS using OpenSSL. Clients authenticate using cryptographic methods before any listing or download is allowed.

To provide fault tolerance, we will use process replication: multiple server processes run in parallel and can serve clients independently. If one process crashes, another process continues handling requests. The MP3 data is not replicated per the assignment and all server instances read from the same local MP3 directory. The client is menu-driven (C on Unix) and supports browsing by genre (Country, Pop, Hip Hop, R&B, Rock), searching by artist or title, viewing results, and downloading a selected file.

### Requirements / Objectives

* Client-Server Model: Implement in C using ssl-client.c and ssl-server.c.
* Virtualization: Run inside Linux VMs or WSL to demonstrate deployment.
* Concurrency: Servers handle multiple clients with threads or forked processes.
* Security: All traffic runs over SSL/TLS. Server uses certificates, and clients authenticate with either mutual TLS or a lightweight challenge.
* Replication: Two server processes run in parallel. If one crashes, the client retries on the backup port.
* Distribution Transparency: Clients only see one logical service. Failover is built into the client.
* Coordination: Menu-driven client supports browsing genres (Country, Pop, Hip Hop, R&B, Rock), searching by artist or title, and downloading.
* Consistency: Not required, since MP3 data isn’t replicated.
* Fault Tolerance: Process replication ensures requests continue if one server goes down.

### Expected Outcomes

1. Client opens SSL/TLS session with server.
2. Client authenticates securely.
3. Menu displays genres, with search by artist or title.
4. Results list comes back over SSL/TLS.
5. Client requests file with GET <file> and receives MP3 in chunks.
6. File is saved locally, and download is confirmed.
7. If primary fails, client switches to backup server automatically.
8. Logs capture errors and state for grading.

**Possible Basic UML for Project**

**Client (ssl-client, C)**

• menu: browse/search

• download mp3

• failover: 4433→4434

**Server Instance B**

• (ssl-server, port 4433)

**Server Instance A**

• (ssl-server, port 4433)

**MP3 Storage**

(/srv/media/mp3)

**Virtualized Host (WSL / Linux VM)**