```
#include <iostream>
#include <fstream>
#include <sstream>
#include <string>
#include <openssl/ssl.h>
#include <openssl/bio.h>
#include <openssl/err.h>

#define SERVER_URL "https://seg-receiver.example.com:443/api/receive"
#define CERT_FILE "seg_sender.crt"
#define KEY_FILE "seg_sender.key"
#define CA_FILE "rootCA.pem"

void send_xml(const std::string& xml_file) {
    SSL_CTX* ctx = SSL_CTX_new(TLS_client_method());
    if (!ctx) {
        std::cerr << "SSL context creation failed!\n";
        exit(EXIT_FAILURE);
    }

    // Load certificates
    if (!SSL_CTX_load_verify_locations(ctx, CA_FILE, nullptr) ||
        !SSL_CTX_use_certificate_file(ctx, CERT_FILE, SSL_FILETYPE_PEM) ||
        !SSL_CTX_use_PrivateKey_file(ctx, KEY_FILE, SSL_FILETYPE_PEM)) {
        std::cerr << "SSL certificate configuration failed!\n";
        SSL_CTX_free(ctx);
```

```
    exit(EXIT_FAILURE);
}

BIO* bio = BIO_new_ssl_connect(ctx);
SSL* ssl = nullptr;

if (!bio || BIO_set_conn_hostname(bio, SERVER_URL) <= 0) {
    std::cerr << "Failed to connect to server!\n";
    BIO_free_all(bio);
    SSL_CTX_free(ctx);
    exit(EXIT_FAILURE);
}

BIO_do_connect(bio);
BIO_get_ssl(bio, &ssl);
SSL_set_mode(ssl, SSL_MODE_AUTO_RETRY);

// Read XML file
std::ifstream file(xml_file);
if (!file.is_open()) {
    std::cerr << "Error opening XML file.\n";
    BIO_free_all(bio);
    SSL_CTX_free(ctx);
    exit(EXIT_FAILURE);
}
```

```
    std::ostringstream buffer;
    buffer << file.rdbuf(); // Read entire file into buffer
    file.close();

    // Send XML data securely
    std::string xml_data = buffer.str();
    BIO_write(bio, xml_data.c_str(), xml_data.size());

    std::cout << "Document sent successfully!\n";

    BIO_free_all(bio);
    SSL_CTX_free(ctx);
}

int main(int argc, char* argv[]) {
    if (argc < 2) {
        std::cerr << "Usage: " << argv[0] << " <XML-file>\n";
        return EXIT_FAILURE;
    }

    send_xml(argv[1]);
    return EXIT_SUCCESS;
}
```