Generating Keys and Certificates for SSL/TLS Communication

1. Create the Keystore and Certificate for the Server

1. **Open a terminal** (Command Prompt on Windows or terminal on Linux).

2. Run the following command to create a keystore with a private key and a self-signed certificate:

   ```
   keytool -genkeypair -alias server -keyalg RSA -keysize 2048 -keystore
   server.keystore -validity 365
   ```

   - Enter a password for the keystore (e.g., `password`).
   - Fill in the requested information (e.g., name, organization, location).
   - The `server.keystore` file will be created.

3. **Export the server's certificate**:

   ```
   keytool -export -alias server -keystore server.keystore
   -file server.cer
   ```

   - Enter the keystore password when prompted.
   - The `server.cer` file will be created. This file contains the public certificate of the server.

2. Create the Truststore for the Client

1. **Import the server's certificate into a truststore**:

   ```
   keytool -import -alias server -file server.cer -keystore
   client.truststore
   ```

   - Enter a password for the truststore (e.g., `password`).
   - Type `yes` to trust the server certificate.
   - The `client.truststore` file will be created.

3. Testing Across Two Systems

- **For two Windows systems**:

  1. Transfer the `server.keystore` file to the server machine.

2. Transfer the `client.truststore` file to the client machine.
3. Update the paths to the keystore and truststore in the `SecureServer` and `SecureClient` code to point to these files on their respective machines.

- **For one Linux and one Windows system**:

  1. Transfer the `server.keystore` file to the Linux machine.
  2. Transfer the `client.truststore` file to the Windows machine.
  3. Ensure that the file permissions on Linux allow access to the `server.keystore` file for the user running the server:

     ```
     chmod 600 server.keystore
     ```

  4. Update the paths in the code on both systems to correctly point to the files. For Linux, use absolute paths (e.g., `/home/user/server.keystore`).

Configuring Java Code to Use Keystore and Truststore

1. On the **server** machine, ensure the server code includes:

   ```
   System.setProperty("javax.net.ssl.keyStore", "path/to/server.keystore'
   System.setProperty("javax.net.ssl.keyStorePassword", "password");
   ```

2. On the **client** machine, ensure the client code includes:

   ```
   System.setProperty("javax.net.ssl.trustStore", "path/to/client.trusts
   System.setProperty("javax.net.ssl.trustStorePassword", "password");
   ```

Running the Applications

1. **Start the server**:

   - On the server machine, run the `SecureServer` class. Ensure the `server.keystore` file is accessible at the configured path.

2. **Run the client**:

   - On the client machine, run the `SecureClient` class. Ensure the `client.truststore` file is accessible at the configured path.

Verifying the Communication

- The client should send a "Hello" message to the server, and the server should respond with "Goodbye".
- If the configuration is correct, the communication will be encrypted and secure.