Course	COMP 7005
Program	Bachelor of Science in Applied Computer Science
Term	January 2025

- This is an individual programming assignment.

Objective

- Develop a client/server application using UNIX Domain Sockets for inter-process communication.
- This assignment emphasizes file transfer, Caesar cipher encryption, and socket programming concepts.

Learning Outcomes

- Understand and implement UNIX Domain Sockets for client-server communication.
- Apply Caesar cipher encryption to transform data.
- Work with file I/O operations in C.
- Gain experience in designing and testing inter-process communication programs.

Assignment Details

Requirements

You will write two programs.

Client

- Accepts the name of a file as a command-line argument.
- Reads the contents of the file.
- Sends the file's content to the server via a UNIX Domain Socket.
- Receives the encrypted file from the server and prints it to the terminal.

Server

- Accepts a shift value for the Caesar cipher as a command-line argument.
- Listens for client connections via a UNIX Domain Socket.
- Receives the file content from the client.
- Encrypts the file content using the Caesar cipher with the specified shift.

- Sends the encrypted content back to the client.

Constraints

- You may use any language that you like.
- The program must run on a UNIX-like Operating System (e.g. Linux or macOS).
- Use UNIX Domain Sockets exclusively (not Network Sockets).
- Ensure proper error handling for socket operations.
- Implement cleanup mechanisms (e.g., removing socket files) to prevent stale socket files.
- Implement the Caesar cipher encryption. Wrap around characters that go beyond a-z or A-Z.
- Only encrypt alphabetic characters. Preserve non-alphabetic characters as-is.
- Do not use Network Sockets or higher-level libraries for socket programming.
- Do not use external encryption libraries; implement the Caesar cipher manually.

Resources

- Man Pages: man 2 socket, man 7 unix, man 3 getopt, man 3 read, man 3 write.
- Caesar Cipher Explanation: Wikipedia.
- Code samples from your course materials.

Submission

- Ensure your submission meets all the guidelines, including formatting, file type, and submission.
- Follow the AI usage guidelines.
- Be aware of the late submission policy to avoid losing marks.
- ***Note: Please strictly adhere to the submission requirements to ensure you don't lose any marks.***

Evaluation

Topic	Value
Correct implementation of the client program	15%
Correct implementation of the server program	15%
Proper use of UNIX Domain Sockets	20%
Design	20%

Testing	30%
Total	100%

Hints

- Test your programs with small and large files to ensure correctness and reliability.
- Use temporary files for debugging to monitor data sent and received over the socket.
- Consider edge cases in encryption, such as files with no alphabetic characters.
- Clean up the UNIX Domain Socket file (unlink) after the server shuts down to prevent errors on subsequent runs.