Exercise 4 (Unix): Custom Protocol over UDP

Objective

Design and implement a simple application-level protocol over UDP using C++ on Unix/Linux.

Task

You will design a minimal custom protocol and implement two programs:

1. Protocol Design

Your protocol should contain the following fields:

- **Sequence Number** (1 byte)
- **Payload** (up to 100 bytes)
- **Checksum** (1 byte) – simple XOR of all payload bytes

Example message format:

```
| Sequence Number (1 byte) | Payload (N bytes) | Checksum (1 byte)
|
```

2. Sender (Client)

- Sends a UDP packet every second.
- Increments the sequence number.
- Computes and includes checksum.

3. Receiver (Server)

- Listens on a UDP port.
- Receives and parses each packet.
- Verifies checksum.
- Prints sequence number, payload, and checksum validity.

Optional Challenge

- Add an acknowledgment mechanism.
- Handle lost or invalid packets.

Starter Code

sender.cpp (Stub)

```
#include <iostream>
#include <cstring>
#include <unistd.h>
#include <arpa/inet.h>
#include <chrono>
#include <thread>

int main() {
    // TODO: Create UDP socket

    // TODO: Fill in server information (IP, Port)

    uint8_t sequence = 0;

    while (true) {
        // TODO: Prepare payload

        // TODO: Compute checksum

        // TODO: Send message to server

        sequence++;
        std::this_thread::sleep_for(std::chrono::seconds(1));
    }

    // TODO: Close socket

    return 0;
}
```

receiver.cpp (Stub)

```
#include <iostream>
#include <cstring>
#include <unistd.h>
#include <arpa/inet.h>

int main() {
    // TODO: Create UDP socket
```

```
    // TODO: Bind to a port

    while (true) {
        uint8_t buffer[1024];
        struct sockaddr_in client_addr;
        socklen_t addr_len = sizeof(client_addr);

        // TODO: Receive message from client

        // TODO: Parse sequence number, payload, checksum

        // TODO: Verify checksum

        // TODO: Print message details
    }

    // TODO: Close socket

    return 0;
}
```

Learning Goals

- Learn how to design a simple application protocol.
- Practice checksum calculation and validation.
- Understand the importance of integrity checks in UDP communication.

Suggested Questions for Students

1. Why do we include a checksum in the protocol?
2. How does the receiver verify the integrity of the packet?
3. How could you handle packet loss in this protocol?

Hint

You will need to use:

- `socket()`
- `sendto()` / `recvfrom()`
- `bind()`

- `close()`