**Group Members**

* **Adam Goh Zheng Shan , 2301303 ,** [**2301303@sit.singaporetech.edu.sg**](mailto:2301303@sit.singaporetech.edu.sg)
* **Brandon Poon**
* **Zul**
* **Gabriel Peh**
* **Kwek Wei Jie**

**Design Introduction**

**1. Overview**

This protocol ensures **reliable data transfer over UDP**, compensating for its lack of **built-in reliability**. It integrates **structured message formats, an acknowledgment (ACK) mechanism, and a transaction sequence** to enable efficient, loss-resilient communication.

**2. Message Format**

Each message follows a structured format:

* **Command ID (1 byte)** – Identifies request/response type.
* **Source/Destination IP (4 bytes) & Port (2 bytes)** – Addressing information.
* **Payload (Variable)** – Data content for specific operations.

For file transfers, an additional **header** includes:

* **Session ID (4 bytes)** – Tracks file transactions.
* **Sequence Number (4 bytes)** – Ensures correct order.
* **File Offset (4 bytes) & Data Length (4 bytes)** – Manages chunked transmission.

Common commands include:

* **REQ\_DOWNLOAD (0x2)** – Request file.
* **RSP\_DOWNLOAD (0x3)** – Server response with metadata.
* **REQ\_LISTFILES (0x4) / RSP\_LISTFILES (0x5)** – File listing.

**3. ACK Scheme**

* UDP lacks **guaranteed delivery**, so this protocol uses:
* **ACKs for received packets** (prevents unnecessary retransmissions).
* **Timeout-based retransmission** (configurable between **10ms–500ms**)
* **Negative Acknowledgments (NAK)** (for missing packets)
* **Configurable packet loss rate (0.0 - 1.0)** to simulate real-world conditions.

**4. Transaction Sequence**

**File Transfer Flow:**

1. **Client Requests File** → Sends REQ\_DOWNLOAD with file details.
2. **Server Responds** → Sends RSP\_DOWNLOAD with metadata.
3. **Data Transmission** → Server sends UDP packets with sequence numbers.
4. **Client Acknowledges** → Sends **ACKs**; lost packets trigger **NAKs**.
5. **Retransmissions** → Server resends missing packets until confirmed.
6. **Completion** → Server sends termination message; client verifies integrity.

**5. Verification of Design**

Our team verified the **custom UDP-based file transfer system** through extensive testing, ensuring **correctness, resilience to packet loss, and performance efficiency**.

**1. Functional Testing**

* Verified that files were successfully **sent, received, and stored** without corruption.
* Tested **file listing functionality** to ensure accurate retrieval of available files.

**2. Packet Loss & Recovery Testing**

* Simulated different **packet loss rates (0.0 - 1.0)** to test retransmission mechanisms.
* Ensured **ACK/NAK responses** were correctly processed, triggering retransmissions when needed.

**3. Multi-Client & Concurrent Transfers**

* Tested simultaneous file transfers with **multiple clients** to verify server stability.
* Ensured clients consistently received **the correct file list** and **file data**.

**4. Stress Testing**

* Transferred **large files (≥100MB)** to evaluate system performance.
* Logged **packet sequence numbers, retransmissions, and ACKs** to verify efficient data flow.

**5. Debugging & Log Analysis**

* Captured **log files** detailing message exchanges, packet losses, and retransmission counts.
* Used **console outputs and Wireshark captures** to monitor UDP packet flow and ensure expected behaviour.

Files being sent:

A screenshot of a computer

AI-generated content may be incorrect.

**6. Individual Contributions**

**Kuek Wei Jie** (SIT ID: 2301325, DigiPen ID: k.weijie)

* Set up the UDP-TCP connection for seamless communication.
* Implemented packet loss handling to ensure data integrity.
* Developed the file listing functionality for retrieving available files.

**Adam Goh Zheng Shan** (SIT ID: 2301303, DigiPen ID: goh.a)

* Designed and implemented reliable data transfer using an ACK-based system.
* Developed the file transmission process for sending and receiving data.
* Built and tested the file download functionality to ensure efficient transfers.