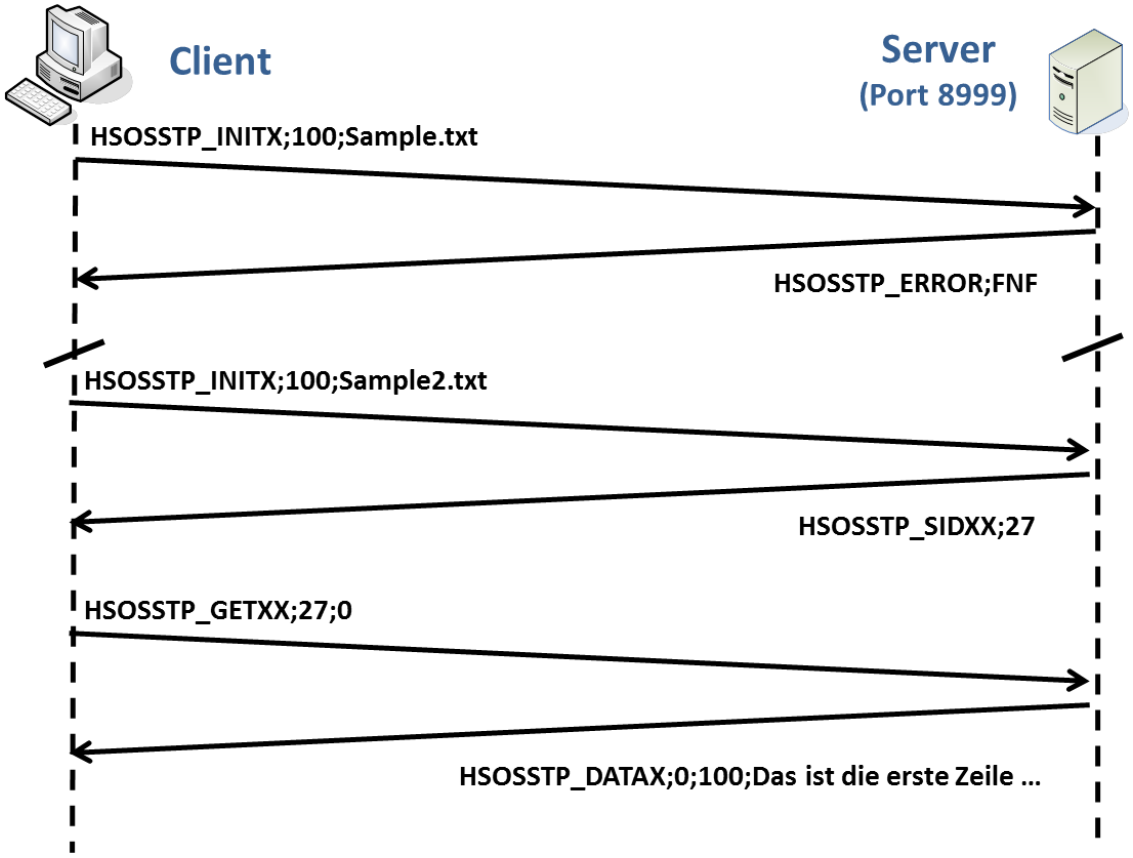
Aufgabenblatt 3

Umsetzung eines Protokolls zum Datentransfer

In dieser Aufgabe soll eine Applikation bestehend aus Client und Server auf Basis von **Sockets** für den Transfer von Dateien zwischen Rechnern implementiert werden. Der Datentransfer soll dabei in Teilen (*Chunks*) festgelegter Größe erfolgen. Die Größe der Chunks wird zu Anfang der Kommunikation durch den Client festgelegt.

Die Funktionsweise des **Protokolls** soll durch das folgende Sequenzdiagramm eines exemplarischen Ablaufs demonstriert werden:



1. Der Client initiiert den Transfer. Der Client sendet dazu eine Nachricht: **`HSOSSTP_INITX;<chunk size>;<filename>`** an den Server. Hierin gibt **`<chunk size>`** die Größe der transferierten Datenblöcke in Bytes an. Der zweite Parameter definiert den Namen der Datei, die vom Server zum Client zu transferieren ist.
2. Es soll zu Anfang ein Sitzungsschlüssel zwischen Client und Server vereinbart werden. Der Server sendet nach der initialen Anfrage des Clients den Sitzungsschlüssel in einer Nachricht: **`HSOSSTP_SIDXX;<session key>`** an den Client. Als Sitzungsschlüssel wird ein Integer-Wert verwendet, der während der Laufzeit des Servers eindeutig sein soll (er kann also z.B. als Zähler realisiert werden). Der Sitzungsschlüssel wird intern im Server mit dem Zustand des zu der Sitzung gehörenden Datentransfers verknüpft (Filedeskriptor der zu transferierenden Datei und Anzahl der übertragenen Chunks).

3. Kann ein Transfer nicht initiiert werden oder tritt während des Transfers ein Fehler auf, so wird an den aufrufenden Prozess eine Fehlermeldung (**HSOSSTP_ERROR;<*reason*>**) ausgegeben.
4. Der Client sendet nun iteriert Anfragen der Art **HSOSSTP_GETXX;<*session key*>;<*chunk no*>** an den Server. Die Chunks werden beginnend mit 0 durchnummeriert. Da die Größe der Chunks bei den **HSOSSTP_GETXX** Nachrichten nicht mitgeschickt wird, muss im Server eine Zuordnung dieses Wertes zu dem Sitzungsschlüssel mitgespeichert werden.
5. Der Server sendet daraufhin das entsprechende Paket vereinbarter Größe zurück. Diese werden mit einem Prefix **HSOSSTP_DATAX;<*chunk no*>;<*actual chunk size*>;<*data*>** versehen **(<*chunk no*>** und **<*actal chunk size*>** geben die Nummer und die Größe des gerade transferierten Chunks an; **<*data*>** die transferierten Daten).
6. Die Daten werden durch den Client gelesen und in der lokalen Datei gespeichert.

Realisieren Sie die o.g. Funktionen in Form eines **iterativen UDP-Servers** und eines **Clients**. Der Server soll in der Lage sein, mehrere Transfersitzungen **gleichzeitig zu bedienen**. Der Client soll mit der IP-Adresse des Servers, der **Größe der Chunks** und dem **Namen** der vom Remote-System zu transferierenden Datei als Parameter aufgerufen werden können. Also beispielsweise:

```
HSOSSTP_client 131.173.110.1 100 Sample.bin
```

Der Server wird ohne Parameter auf dem lokalen Host gestartet.

Aus Gründen der Kompatibilität (sie sollen auch die Server der anderen Teilnehmer verwenden können) seien die folgenden Fehlermeldungen vorgegeben:

- **HSOSSTP_ERROR;FNF**: angeforderte Datei kann nicht gefunden werden
- **HSOSSTP_ERROR;CNF**: angeforderter Chunk kann nicht transferiert werden
- **HSOSSTP_ERROR;NOS**: keine Session vorhanden

Beantworten Sie im Protokoll die folgenden Fragen zu Ihrer Lösung:

- Was ist speziell bei Verwendung von UDP als Transportprotokoll zu berücksichtigen?
- Warum sollte für die Umsetzung des Protokolls UDP und nicht TCP verwendet werden?
- Wie kann man das Ende einer Datentransfer-Sitzung erkennen? Beschreiben Sie die bei Ihrer Lösung ausgetauschten Nachrichten durch ein Sequenzdiagramm.
- Wie werden Datentransfer-Sitzungen im Server terminiert?
- Was passiert bei Wiederaufnahme einer Sitzung?
- Was passiert beim Zugriff auf eine nichtexistierende Sitzung?
- Wie groß kann die Chunk-Size maximal eingestellt werden?

Sie sollen die Funktion des Systems bei Verteilung des Clients und Servers auf unterschiedliche Hosts demonstrieren. Testen Sie mit Ihrem Client mindestens einem Server einer anderen Gruppe!

Freiwillige Zusatzaufgabe:
Messen Sie die Transferzeiten für verschiedene Einstellungen der Chunk-Size (128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768). Tragen Sie die gemessenen Laufzeiten (diese können durch Aufruf der Systemfunktion **`time()`** gewonnen werden) in Abhängigkeit der Chunk-Size in einem Diagramm auf. Was stellen Sie fest?

Hinweise:

Zur Vereinfachung können Sie die folgenden Annahmen treffen:

- Die Auswahl des **Implementierungsansatzes** (C/C++ oder Java; idealerweise werden 2 verschiedene Programmiersprachen eingesetzt) steht Ihnen frei. Achten Sie darauf, dass auch bei Textdateien die Steuerzeichen korrekt übertragen werden. Dateien sollten also *binär* gelesen werden und wie gelesen im Datagramm verpackt werden.
- Die Anwendung soll fest auf **Port 8999** laufen.
- Es kann davon ausgegangen werden, dass Client und Server jeweils ein **festes Arbeitsverzeichnis** verwalten, aus dem Dateien gelesen (Server) und in das die übertragenen Dateien (Client) geschrieben werden. Es müssen also keine Pfade hinsichtlich der zu transferierenden Dateien angegeben werden. Verhindern Sie aber das Schreiben der vom Server gelesenen Datei durch den Client!
- Machen Sie sich vor Beginn der Implementierungsarbeiten den genauen Ablauf (z.B. anhand eines Sequenzdiagramms) klar und überlegen Sie sich eine **Modularisierung** Ihrer Lösung in Form von geeigneten Funktionen und Datenstrukturen.
- Beispiel-Dateien (**`Sample.txt`**, **`Sample.bin`**) befinden sich im Lernraum.
- Mit dem Linux-Kommando **`diff`** können Sie die transferierten und die Original-Dateien miteinander vergleichen.

Für das Testat ist ein doppelseitiger Ausdruck (inkl. Namen der Gruppenmitglieder) der Ergebnisse abzugeben.

Testierung: 11./12.4.2023