TKN Telecommunication Networks Group

Praktikum
Rechnernetze und Verteilte Systeme

Block 2

— Verbindungsorientierte und verbindungslose Datenübrtragung —

Termin: 27.-29.10.2014 & 3.-5.11.2014

1 Theoretische Vorbereitungsaufgaben

Die folgenden Aufgaben sollen Ihnen helfen, sich auf den Vorbereitungstest vorzubereiten:
Aufgabe 1:
Machen Sie sich mit den Unterschieden zwischen verbindungsorientierter und verbindungsloser Datenübertragung vertraut und beantworten Sie jeweils für „Datagram-Service“ und „zuverlässigen Byte-Strom“ die folgenden Fragen:

- Ist garantiert, dass Pakete der gleichen Route folgen?
- Wird ein Verbindungsaufbau benötigt?
- Kommen Pakete korrekt an?
- Kommen Pakete in der richtigen Reihenfolge an?
- Was ist der Unterschied zwischen einem Datagram und einem Stream?

Aufgabe 2:
Machen Sie sich mit der Berkeley Socket API vertraut und beantworten Sie die folgenden Fragen:

- Geben Sie jeweils für die folgenden API-Aufrufe an, was diese genau bewirken:
  - socket
  - bind
  - listen
  - accept
  - connect
  - send
  - recv
  - close
- Skizzieren Sie den Ablauf der API-Aufrufe einer Stream Socket Kommunikation zwischen Client und Server unter Verwendung von Berkeley Sockets im fehlerfreien Fall. Wiederholen Sie die Aufgabe ebenfalls für Datagram Sockets.

Aufgabe 3:
Wozu werden Portnummern verwendet?

2 Praktische Aufgaben

Die praktischen Aufgaben sind in Kleingruppen von i. d. R. 3 Personen zu lösen. Die Ergebnisse des ersten Termins führen Sie im zweiten Termin dem Tutor vor. Reichen Sie bitte des weiteren den Quelltext bzw. Lösungen bis Sonntag vor dem dem zweiten Termin 23:55 Uhr per ISIS ein.

Im zweiten Termin werden vertiefende praktische Aufgaben gestellt, während der Tutor Lösungen des ersten Termins abnimmt. Reichen Sie bitte den Quelltext bzw. Lösungen dieser Aufgaben bis Sonntag vor dem nächsten Termin 23:55 Uhr per ISIS ein.

Es besteht in beiden Terminen grundsätzlich Anwesenheitspflicht.

2.1 Im ersten Termin zu lösen

Auf der ISIS-Seite zur Veranstaltung finden Sie zusätzliche Literatur und Hilfen für die Bearbeitung der Aufgaben! Zusätzlich können wir den kostenlosen „Beej's Guide to Network Programming Using Internet Sockets“[1] empfehlen.

Aufgabe 4:
Zuerst sollen Sie für einen fertigen Datagram- und Stream-Socket Server zwei entsprechende Clients programmieren. Die von uns verwendeten Programme nutzen als Datagram-Service UDP (User Datagram Protocol) und als zuverlässigen Byte-Strom TCP (Transmission Control Protocol). Einen bereits kompilierten Server stellen wir Ihnen auf der ISIS-Seite zur Verfügung (Aufruf: `tcpUdpServer <tcpPort> <udpPort>`).

Weiterhin wird Ihnen ein Grundgerüst für den Stream-Socket Client gestellt welches mit wenigen Änderungen auch für den Datagram-Client genutzt werden kann. Für die ersten Tests erhalten Sie auch kompilierte Versionen der fertigen Clients.

Zum Kompilieren der Clients unter Linux verwenden Sie den Befehl:
`gcc -lnsl -lresolv <dateiname.c> -o <ausgabe>`

Ihre Clients sollen vom Benutzer über die Kommandozeile zwei natürliche Zahlen entgegennehmen, sowie die IP Adresse und Port-Nummer des Servers. Die beiden Zahlen sollen an den Server gesendet werden. Dieser berechnet den größten gemeinsamen Teiler und gibt das Ergebnis über die Standardausgabe aus. Die korrekte Funktionsweise Ihrer Clients soll für Sie anhand von Debug-Informationen nachvollziehbar sein.

Um die beiden Zahlen aus den empfangenen Daten zu extrahieren, verwendet unser Server die folgende Funktion. Passen Sie die Clients also entsprechend an:

```
void unpackData(unsigned char *buffer, unsigned int *a, unsigned int *b)
{
        *a = (buffer[0]<<8) | buffer[1];
        *b = (buffer[2]<<8) | buffer[3];
}
```

Aufgabe 5:
Im zweiten Teil dieser Aufgabe sollen Sie einen **einfachen** Stream-Socket Server implementieren, der alle Funktionalitäten besitzt, die in der ersten Aufgabe vorgegeben waren. Es genügt, wenn der Server mit einem Client gleichzeitig kommunizieren kann.

[1] `http://beej.us/guide/bgnet/`

2.2 Im zweiten Termin zu lösen

Aufgabe 6:

Benutzen Sie die Funktion clock_gettime (`man clock_gettime`) um innerhalb Ihrer Programme (Stream- und Datagram-Socket Client) Zeitstempel zu setzen, um zu messen, wie lange das Senden der Zahlen vom Client zum Server insgesamt dauert. Geben Sie die Zeitdifferenz zwischen den Stempeln jeweils auf der Konsole aus. Was stellen Sie beim Vergleich zwischen Stream- und Datagram-Socket Client fest? Haben Sie eine Erklärung?

Reichen Sie ihre Lösung auf ISIS ein. Abzugeben sind:

- Der neue Quelltext Ihrer Clients
- Eine kurze Begründung (max. 3 Sätze) zur Erklärung der Differenz zwischen Stream- und Datagram-Socket Client

Aufgabe 7:

Sie sollen nun Ihren Stream-Socket Client so umschreiben, dass er Daten von HTTP-Servern abrufen kann.

a) Benutzen Sie zuerst das Programm Netcat (`man nc`) um sich mit einem Server Ihrer Wahl auf dem HTTP-Port (80) zu verbinden (`nc www.tu-berlin.de 80`). Senden Sie dann (angeasst an den Server, den Sie verwenden) die folgenden Befehle:

   GET / HTTP/1.1
   HOST: www.tu-berlin.de
   Leerzeile

   Der Server sollte jetzt die entsprechende Seite senden und Netcat sollte sie auf der Konsole ausgeben. Sie können Netcat mit `Strg+C` beenden.

b) Machen Sie eine Kopie Ihres Streaming-Socket Clients und schreiben Sie ihn dahingehend um, dass er ihre Anfrage aus Teil a zu einem Server sendet. (Tipp: Mit Netcat können Sie auch einen einfachen Streaming-Socket Server betreiben, um Ihren Client zu überprüfen, z.B. mit "nc -l <port>". Wählen Sie hierfür temporär einen Port über 1024, da auf UNIX-artigen Betriebssystemen das Betrieben von Diensten auf Ports unter 1024 Root-Rechte voraussetzt.)

c) Geben Sie in Ihrem Client die Antwort, die Sie vom Server erhalten, auf der Konsole aus.

d) Reichen Sie ihre Lösung auf ISIS ein. Abzugeben sind:
   - Der neue Quelltext Ihres Clients
   - Die Ausgabe Ihres Clients in einer Text-Datei

3 Vertiefungsaufgaben

Diese Aufgaben sind zu Ihrere eigenen Vertiefung in Hinblick auf die Klausurvorbereitung gedacht:

Aufgabe 8:
Machen Sie sich mit Naming & Addressing innerhalb des Internet-Stacks vertraut und beantworten Sie die folgende Frage:

- Was ist der Unterschied zwischen Name und Adresse?

Aufgabe 9:
Nennen Sie 4 falsche Annahmen an ein Netzwerk, die bei der Entwicklung eines verteilten Systems zu Problemen führen können. Welche Probleme können jeweils in der Realität auftreten?

Aufgabe 10:
Beantworten Sie für jeden Punkt gesondert die Frage: Welche Gründe sprechen dafür und dagegen für diese Anwendung „Datagram-Service“ oder „zuverlässigen Byte-Strom“ zu nutzen?

- Sprachübertragung
- Dateiübertragung
- Remote-Login
- Multicast-Kommunikation (an mehrere Gegenstellen gleichzeitig senden)

Aufgabe 11:
Beantworten Sie folgende Fragen rund um den TCP/IP Stack:

- Welche Layer gibt es im TCP/IP Modell?
- Angenommen Sie öffnen einen Browser und senden eine Web-Anfrage (HTTP) über TCP über IP über Ethernet. In welcher Reihenfolge werden die Header gesendet?
- Angenommen Sie haben zwei Browser gleichzeitig geöffnet und rufen vom gleichen Server gleichzeitig über HTTP das selbe Dokument ab. Wie unterscheidet der Server zwischen den beiden Anwendungen?
- Was bringt das „Hourglass Model“ zum Ausdruck?