



Vernetzte Systeme

Lösungsvorschlag 3

Vorbemerkung

Die vollständigen Quellcode-Dateien der Lösungsvorschläge können von unserer Homepage heruntergeladen werden. Auf diesem Lösungszettel finden Sie Ausschnitte und ein paar Anmerkungen dazu.

1 Registrierung beim Server

- a) Sie können die gegebene Zahl auf verschiedene Arten in einzelne Bytes zerlegen. Beispielsweise ist es möglich, mit Hilfe von Verschiebe(Shift)-Operatoren, Divisions/Modulo-Operatoren und/oder anderen Bitmanipulationen die Aufgabenstellung zu erfüllen. Grundsätzlich ist darauf zu achten, dass Sie lediglich die unteren beiden Bytes der (unter Umständen grösseren) gegebenen Zahl betrachten.

```
out.write((value & 0xff00) >> 8);  
out.write(value & 0xff);
```

Um das höherwertige der unteren beiden Bytes zu schreiben, wird dieses zuerst mittels des bitweisen AND-Operators (&) maskiert und dann um acht Bits mit dem Bit-Verschiebe-Operator (>>) nach rechts geschoben. Mit der `write`-Methode eines `OutputStreams` lassen sich einzelne Bytes übertragen.

- b) Die hier vorgeschlagene Lösung führt die zuvor aufgeteilten Bytes mittels bitweiser Or-Verknüpfung wieder in eine Zahl zusammen. Java's `read`-Methode liefert direkt den gelesenen Wert zurück.

```
(in.read() << 8) | in.read();
```

- c) Um sich beim Server zu registrieren, schicken Sie über die bestehende Verbindung die notwendigen Daten im spezifizierten Format.

Zuerst senden Sie den Wert von `REGISTER` (`0x01`) als Kennung. Der eigene Port wird mit Hilfe der zuvor geschriebenen `write16`-Prozedur übertragen. Die Länge des Namens ermitteln Sie mit Hilfe von `string.length()`. Die Übertragung des Namens fällt leicht, wenn man die `getBytes`-Methode eines Strings und die entsprechende `write`-Methode des `OutputStreams` verwendet, die ein ganzes Byte-Array übertragen kann.

```
out.write(0x01); // send REGISTER command  
write16(port); // send port number  
out.write(name.length()); // send length  
out.write(name.getBytes(),0,name.length()); // send name  
out.flush(); // flush the stream
```

- d) Sie erhalten die im spezifizierten Format angegebenen Daten und müssen diese bearbeiten. Es werden nach und nach alle Daten empfangen. Nach der Überprüfung des Message-Typs verwenden wir die zuvor geschriebene `read16`-Funktion, um die Anzahl der Benutzer in Erfahrung zu bringen. Sie benötigen nun eine Schleife, in der für jeden dieser Benutzer weitere Informationen empfangen werden. Dass heisst, es wird zuerst die Länge des Namens ausgelesen, danach der Name selbst. Anschliessend folgen vier Bytes, welche die IP-Adresse darstellen, unter welcher der Benutzer erreichbar ist. Abschliessend wird noch der Port wiederum mittels der `read16`-Funktion ausgelesen und Informationen auf der Konsole angezeigt.

```

if ((byte)in.read() != 0x03) throw new IOException("Unrecognized protocol message.");
int number = read16(); // read number of users
for (int i=0; i<number; i++) {
    int nameLength = in.read(); // read length of name
    byte[] nameBuffer = new byte[nameLength];
    in.read(nameBuffer); // read the name as byte array
    String name = new String(nameBuffer); // convert byte array to string
    byte[] ip = new byte[4];
    in.read(ip); // read ip as byte array
    int port = read16(); // read the port
    System.out.println(name + "@" + formatIP(ip,port)); // format and print to standard output
}

```

Ausgelassen ist hier die Formatierung der IP:Port-Adresse, die sich in der Methode `formatIP` verbirgt und im vollständigen Quellcode nachgelesen werden kann.

2 Paketvermittlung

- a) Die Verzögerung V_{a1} von A nach R1 ist

$$V_{a1} = \frac{L}{R} = \frac{10MB}{10Mbps} = \frac{80Mb \cdot s}{10Mb} = 8s$$

Für V_{a2} von R1 nach R2 berechnen wir analog $V_{a2} = 20ms$. Für die Übertragung von A nach B ergibt sich somit eine Verzögerung von $V_a = 2 \cdot V_{a1} + 2 \cdot V_{a2} = 16.04s$.

- b) Die Verzögerung V_{b1} von A nach R1 ist 80ms. Die Verzögerung V_{b2} von R1 nach R2 ist 0.2ms. Somit wird die gesamte Verzögerung zu $V_b = 101 \cdot V_{b1} + 2 \cdot V_{b2} = 8080.4ms$

Man kann sich das hier wie folgt überlegen: Die 10Mbps-Strecken stellen hier den Engpass dar; sie verursachen den grössten Teil der Verzögerung. Also wird zuerst die Verzögerung der Übermittlung der 100 Pakete auf der ersten 10Mbps-Strecke (A nach R1) berechnet. (= $100 \cdot V_{b1}$). Dann muss das letzte Paket noch die restlichen Router passieren (= $2 \cdot V_{b2} + V_{b1}$).

- c) Pakete müssen mehr Information als nur die zu übertragenden Daten enthalten. Sie müssen zum Beispiel Information über Zieladresse, Absenderadresse, Reihenfolgennummer, Grösse oder die Quersumme mit sich tragen. Diese Daten sind im sogenannten **Header** eines Pakets. Wie Headers aussehen, ist in den entsprechenden Protokollen beschrieben. IP- und TCP-Pakete haben zum Beispiel je einen Header von mindestens 20 Byte, UDP hat einen Header von 8 Byte.

Pakete können also in der Praxis nicht beliebig klein sein, da sie aus einem Header und den eigentlichen Nutzdaten bestehen. Die Übertragung des Headers stellt dabei einen notwendigen Mehraufwand dar.

- d) Bei Verlust oder Bitfehlern sind im Allgemeinen kleinere Pakete zu bevorzugen. Diese können gegebenenfalls einzeln nochmals übertragen werden. Dazu sind jedoch Mechanismen notwendig, um Verlust oder Fehler zu detektieren und den Sender zur erneuten Übertragung aufzufordern. Zum erneuten Senden gibt es zwei Strategien, "go-Back-N" und "selective repeat", auf welche diese Vorlesung im dritten Kapitel eingeht.

3 Das HTTP-Protokoll

- a) Zuerst starten Sie eine `telnet`-Session durch Angabe einer Adresse und der Portnummer:

```
telnet dcg.ethz.ch 80
```

Daraufhin meldet sich der Server mit:

```
telnet dcg.ethz.ch 80
Trying 82.130.103.73...
Connected to pc-4520.ethz.ch.
Escape character is '^]'.

```

Geben Sie `GET` gefolgt von der gewünschten Webseite und der zu verwendenden HTTP-Protokollversion ein. Beachten Sie, das die URL der Webseite durch `"/` abgeschlossen werden muss, sonst erhalten Sie vom Server nicht die gewünschte Web-Seite, sondern nur einen Verweis auf die richtige URL. Die HTTP-Protokollversion wird in der HTTP-Anfrage beschrieben durch `HTTP/[version]`, wobei `[version]` die gewünschte Version bezeichnet. Zur Zeit existieren die Versionen 1.0 und 1.1.

```
GET /lectures/ss07/vernetzte_systeme/ HTTP/1.0
```

In beiden HTTP-Versionen erwartet der Web-Server ein zusätzliches Newline nach der `GET`-Anfrage. Drücken Sie also nochmals die Enter-Taste. Daraufhin wird die gewünschte Seite zum Client übertragen. Die ganze Sitzung sollte also in etwa so aussehen:

```
> telnet dcg.ethz.ch 80
Trying 82.130.103.73...
Connected to pc-4520.ethz.ch.
Escape character is '^]'.
GET /lectures/ss07/vs/ HTTP/1.0

HTTP/1.1 200 OK
Date: Wed, 19 Apr 2006 12:35:49 GMT
Server: Apache/2.0.55 (Debian) DAV/2 SVN/1.2.3 PHP/4.4.2-1 mod_perl/2.0.2 Perl/v5.8.7
X-Powered-By: PHP/4.4.2-1
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <title>Vernetzte Systeme</title>
  .
  .
  mehr html hier
  .
  .
</html>
Connection closed by foreign host.
```

- b) Einen 404 Fehler können Sie provozieren, in dem Sie eine nicht vorhandene Datei abrufen, also etwa `GET /ToteHose.html HTTP/1.0` von unserem Server.

Einen 304 Fehler können Sie provozieren, in dem Sie ein Datum angeben, dass neuer ist, als das Erstellungsdatum der abzurufenden Datei.¹ Zum Beispiel führt

```
GET / HTTP/1.0
If-Modified-Since: Fri, 01 Jan 2010 00:00:00 GMT
```

auf dem Webserver der ETH (`www.ethz.ch`) zum gewünschten Status Code.

Auch folgende Status-Codes sind relativ einfach zu erzeugen:

- 400 Bad Request: HALLO :-)
(anstelle von `GET` auf einem beliebigen Server...).

¹Das Datum erhalten Sie meist beim Abruf der Datei. Viele Webserver führen einen einfachen String-Vergleich durch, so dass man das exakte Erstellungsdatum angeben muss.

- 401 Not Authorized: GET /student/kenoa/private/ HTTP/1.0.
Die Meldung wird empfangen, wenn man eine Datei aus einem passwortgeschützten Bereich abrufen möchte (hier auf n.ethz.ch).
- 403 Forbidden: GET /geheim/NixFuerDich.html HTTP/1.0.
Die Meldung wird zum Beispiel gesendet, wenn Sie ein Dokument mit HTTP abrufen wollen, auf dieses aber nur via HTTPS zugegriffen werden kann.