Christian Schindelhauer

Freiburg, den 07. Juli 2010
Besprechung 14. Juli 2010

Übungen zur Vorlesung
Systeme II / Netzwerke I
Sommer 2010
Blatt 11

AUFGABE 1:

1. Starten Sie auf `http://cone.informatik.uni-freiburg.de/` und klicken sie zufällig auf einen beliebigen Link. Fahren Sie durch zufälliges Klicken fort und dokumentieren Sie auf welchen Webseiten Sie landen.

2. Starten Sie erneut auf `http://cone.informatik.uni-freiburg.de/` und versuchen Sie durch gezielte Benutzung von Links (keine Suche) mit möglichst wenigen Klicks zu `http://www.fifa.com/` zu gelangen. Dokumentieren Sie ihr Ergebnis.

3. Benutzen Sie verschiedene Suchmaschinen (u.a. Google und Baidu sowie noch zwei beliebige andere), um nach "*Christian Schindelhauer*" zu suchen und vergleichen Sie die Ergebnisse.

AUFGABE 2:
Erstellen Sie eine Klausuraufgabe aus einem beliebigen Themenbereich der Vorlesung und tragen Sie die von ihnen erstellte Klausuraufgabe in der Übungsgruppe vor.

AUFGABE 3:
In dieser Aufgabe sollen Sie mit dem Simple Mail Transfer Protocol vertraut werden und einen Client entwickeln, der eine unverschlüsselte Verbindung zu einem SMTP-Server aufbaut, ihm eine E-Mail mit einer Binärdatei sendet und die Verbindung wieder abbaut. Die nachfolgenden Programmvorlagen verwenden Java. Sie können natürlich auch eine beliebige andere Programmiersprache verwenden, mit der Sie auf gleiche Weise SMTP implementieren können.

1. Vervollständigen Sie die Methode `sendMail` so, dass Sie mit ihr eine Verbindung zum SMTP-Server aufbauen können.

2. Die Methode `sendMail` verwendet zum Versenden der Mail einen bestimmten Standard. Ermitteln Sie diesen. *Hinweis: Lesen Sie RFC2387.*

3. Verwenden Sie nun die Methode `send`, um dem Server folgende Textzeilen zu senden. Es ist wichtig, dass Sie für jede einzelne Zeile jeweils einen `send` Aufruf verwenden. Wenn Sie ein `boundary` senden, müssen Sie diesem einen Zeilenumbruch voranstellen. Der Server steht Ihnen unter der Adresse `smtp.malteahl.de` zur Verfügung. Verwenden Sie als Empfänger `mail@malteahl.de`. Verhalten Sie sich fair. *Hinweis: Der Server sendet Ihnen ein* `OK` *oder* `ERROR` *für jede Header/Tail Zeile, die Sie ihm zusenden. Die Methode* `warteAufServer` *können Sie im Header/Tailer beliebig einfügen, falls Sie vom Server keine Nachrichten zu sehen bekommen.*

```
HELO theWorld
MAIL FROM: <student@email.xy>
RCPT TO: mail@malteahl.de
DATA
MIME-Version: 1.0
Subject: Multipart-Test
From: MAIL <student@email.xy>
Content-Type: multipart/mixed;\n boundary="meinboundary"
\n--meinboundary
Content-Type: text/plain; charset="utf-8"\n
Hallo! Hier meine Aufgabe zum Vorrechnen.
\n--meinboundary
\n--meinboundary--
```

4. Packen Sie Ihren Programmcode in eine Zip-Datei und benennen Sie sie mit Ihrem Namen. Wandeln Sie diese nun zum Versenden in eine Reihe von ASCII-Zeichen um. Dafür kommt der BASE64-Algorithmus zum Einsatz, der einen beliebigen Bit-Strom in 6 Bit Wörter aufteilt. Mit 6 Bit können Sie 64 verschiedene Zeichen eindeutig kodieren. BASE64 verwendet alle Groß- und Kleinbuchstaben, die Zahlen 0 bis 9 und die Zeichen '+' und '/'. Das '=' wird zum Auffüllen verwendet. Auf diese Weise lässt sich jeder beliebige (8-Bit) Binärdatenstrom in eine auf allen Systemen kompatible Darstellung bringen. Verwenden Sie zum Kodieren ein Programm aus dem Internet oder einfach ein Mail-Programm wie "Thunderbird".

5. Senden Sie dem Server nun dieselbe Mail, diesmal jedoch um einen weiteren "Part" ergänzt:

```
Content-Type:application; name=MaxMuster.zip
Content-Disposition: attachment;filename="MaxMuster.zip"
Content-transfer-encoding: base64
AEfrE....Na=
--meinboundary
```

6. *(Zusatz)* Finden Sie heraus, wozu man `SMPT-AUTH` verwendet. Versuchen Sie die Mail an Ihre eigene Mailadresse zu senden. Nennen Sie einige Risiken bzgl. der Sicherheit bei diesem Verfahren.

```
public static void sendMail(String mailServer,
  String recipient,
  String filename){
//Zum Mail-Server verbinden...
Socket s = ... ;
BufferedReader in = new BufferedReader
      (new InputStreamReader(s.getInputStream(), "utf-8"));
BufferedWriter out = ... ;
String boundary="meinboundary";
send( in,out, "HELO theWorld");
send(in,out,".");
send(in,out, "QUIT");
s.close();
}
```

```
public static void send(BufferedReader in,
  BufferedWriter out,
  String s) {
out.write(s + "\n");
out.flush();
System.out.println("Client: "+s);
if(in.ready())System.out.println("Server: "+in.readLine());
}
```

```
public static void warteAufServer(BufferedReader in ){
while(!in.ready());
System.out.print(in.readLine());
}
```