TECHNISCHE UNIVERSITÄT BERLIN
Fakultät IV – Elektrotechnik und Informatik
Fachgebiet Intelligente Netze und Management verteilter Systeme
Prof. Anja Feldmann, Ph.D.
Gregor Maier, Jörg Wallerich, Andi Wundsam

10. Blatt Praktikum Protokolldesign WS 07/08

Aufgabe 1: (100 Punkte) Dateitransfer im P2P-Netzwerk

Nachdem das P2P-Protokoll in der Version 0.2 nun Nachrichten mit Payload unterstützt und solche Nachrichten durch das Overlay geroutet werden können, soll nun abschließend die Übertragung von Dateien implementiert werden.

Ein Download wird mittels einer `GET`-Nachricht (siehe Blatt 5) eingeleitet. Ein solcher `GET`-Request ist für die Version 0.2 des Protokolls folgendermaßen spezifiziert:

```
GET FROM <node id> FOR <node id> KEY <filename> MESSAGE-ID <id> TTL <ttl> P2P/0.2\r\n
Content-Length: 0\r\n
Application: transfer\r\n
\r\n
```

Das `FROM`-Feld spezifiziert dabei denjenigen Knoten, der eine Datei herunterladen will, während das `FOR`-Feld eine Knoten spezifiziert, von dem wir wissen, dass er die gewünschte Datei liefern kann (siehe `SEARCH` und `FOUND`).

Wenn eine solche Nachricht von dem mit `FOR` angegebenen Knoten empfangen wird, kann sofort mit der Übertragung der angeforderten Datei begonnen werden. Dazu werden die Daten in kleinen Blöcken von maximal 1024 Bytes in den Body von mehreren Response-Nachrichten gepackt und dann verschickt. Alle diese Nachrichten sind Response Nachrichten vom Typ `TRANSPORT`. Der Antwortcode für alle Nachrichten mit Ausnahme der letzten ist 350. Die letzte Nachricht hat einen anderen Antwortcode und signalisiert so das Ende der Übertragung. Der Antwortcode der letzten Nachricht ist dann 270.

Eine Antwort-Nachricht zur Übertragung von Dateien sieht dann folgendermaßen aus:

```
P2P/0.2 350 TRANSPORT FOR <node id> FROM <node id> MESSAGE-ID <id> KEY <filename> TTL <ttl>\r\n
Application: transfer\r\n
Content-Length: <num bytes, max. 1024>\r\n
Offset: <offset in bytes>\r\n
\r\n
<payload>
```

Die `FOR` und `FROM`-Felder funktionieren wie üblich und dienen der Spezifikation von Absender und Empfänger der Nachricht. Das `KEY`-Feld in der ersten Zeile ist immer das selbe wie in der Anfrage. Beachte jedoch, dass jede Nachricht eine eigene `MESSAGE-ID` bekommt!

Die Applikation wird als `transfer` spezifiziert um sie als Download-Nachricht erkennen zu können. Der `Offset`-Header besagt, wo in der Datei das erste Byte der Payload hingehört. Dies ist notwendig, um mit umsortierten Nachrichten effizient umgehen zu können (entspricht ungefähr den Sequenznummern in TCP). Wir gehen davon aus, dass keine Nachrichten verloren gehen, deshalb werden *keine* Empfangsbestätigungen gebraucht.

Die Knotensoftware soll alle eingehenden Antwortnachrichten wieder korrekt zu einer Datei auf dem Rechner zusammenfügen.

Abzugeben sind:

- Dein Programm (sowohl Quelltext als auch ggf. compiliert im Falle von C, C++ und Java)
- Ggf. eine Beschreibung, was funktionieren sollte, dies leider aber immer noch nicht tut...

Details zur Abgabe der Aufgaben: siehe FAQ
(unterhalb `http://www.net.t-labs.tu-berlin.de/teaching/ws0708/PD_labcourse/`**).**
Abgabedatum: Dienstag, 22. Jannuar 2008, 11:59h s.t.