Aufgabe 92: xmt-httpd (14.0 Punkte)

Implementieren Sie einen mehrfädigen Webserver `xmt-httpd` (eXtended Multi-Threaded HTTP Daemon), der HTTP-Anfragen auf einem wählbaren Port *port* entgegennimmt und Dateien innerhalb eines festen Verzeichnisbaums *www-path* ausliefert. Das Programm wird wie folgt aufgerufen:

```
mt-httpd <www-path> <port> <threads> <bufslots>
```

Eine Anfrage sieht wie im folgenden Beispiel aus: `GET /doc/index.html`
Der Pfadname enthält hierbei keine Leerzeichen. Weitere Wörter, die dem Pfadnamen eventuell folgen, **müssen** vom Server **ignoriert** werden. Die Anfragezeile wird entweder mit `\r\n` oder mit `\n` terminiert.

a) Einfädiger Webserver

Beginnen Sie zunächst mit einem einfädigen Webserver, in dem der Hauptthread die Anfrage sofort bearbeitet, bevor eine neue Verbindung entgegengenommen wird (**accept(2)**). Testen Sie Ihren Webserver mit dem WWW-Pfad `/proj/i4sp2/pub/aufgabe92/wwwdir` und rufen Sie dann in einem Webbrowser (z. B. Firefox) die folgende URL auf, woraufhin eine einfache Seite mit einem Link erscheinen sollte:

```
http://<RECHNERNAME>:<PORT>/index.html
```

Sorgen Sie dafür, dass das `SIGPIPE`-Signal ignoriert wird (siehe *josh*), damit der Webserver nicht bei vorzeitigen Verbindungsabbrüchen terminiert.

b) Mehrfädiger Webserver

Erweitern Sie den Server nun so, dass mehrere Arbeiter-Threads (siehe *palim*) die Anfragebearbeitung übernehmen und der Hauptthread nur noch für die Verbindungsannahme zuständig ist. Es werden *threads* Arbeiter-Threads beim Programmstart erzeugt, die dann für die Laufdauer des Programms bestehen bleiben. Verwenden Sie zum Austausch gemeinsamer Daten zwischen den Arbeiter-Threads und dem Hauptthread einen Ringpuffer (siehe *jbuffer*) mit *bufslots* Einträgen. Der Hauptthread nimmt nun nur noch die Verbindungen an und trägt die zugehörigen Socket-Dateideskriptoren in den Ringpuffer ein. Die Arbeiterthreads entnehmen dann jeweils einen Dateideskriptor aus dem Ringpuffer und führen die Bearbeitung der zugehörigen Verbindung durch.

c) Automatische Anzeige von Verzeichnissen

Passen Sie den Server so an, dass nicht nur Dateien ausgeliefert, sondern auch Verzeichnisse aufgelistet werden. Sollte die übergebene URL ein Verzeichnis sein, in dem sich keine `index.html`-Datei befindet, geben Sie alle Dateien in diesem Verzeichnis (siehe *crawl*), die nicht mit `.` beginnen aus. Optional können die Dateinamen auch sortiert (siehe *wsort*) ausgegeben werden.
Zur Formatierung der Ausgabe können Sie auf die Funktionen im Modul `2html` zurückgreifen.

d) Ausführen von Perl-Skripten

Erweitern Sie den Server um die Möglichkeit Perl-Skripte auszuführen, deren Ausgabe an den Client gesendet wird (siehe *clash* und *josh*). Aus Sicherheitsgründen sollen nur Dateien, die auf `.pl` enden und deren Eigentümer das Ausführen-Recht hat, ausgeführt werden. Jedes Perl-Skript soll in einem eigenen Prozess ausgeführt werden. Achten Sie darauf, die entstehenden Zombies sofort einzusammeln.
Zum Testen liegen einige Perl-Skripte im Verzeichnis `/proj/i4sp2/pub/aufgabe92/wwwdir/perl-test` bereit.

Hinweise zur Aufgabe:

- Erforderliche Dateien: `xmt-httpd.c`
- Eine ausführliche Beschreibung des HTTP-Protokolls in der Version 0.9 finden Sie unter `http://www.w3.org/Protocols/HTTP/AsImplemented.html`.
- Der Server soll sowohl IPv4- als auch IPv6-Verbindungen bedienen.
- Im Verzeichnis `/proj/i4sp2/pub/aufgabe92` finden Sie die Schnittstellen der Module `sem` und `bbuffer` sowie einige nützliche Hilfsroutinen in den Modulen `i4httools` und `2html`. Verwenden Sie insbesondere die Funktion `checkpath()`, da es sonst leicht möglich ist, über Ihren Webserver an beliebige Dateien aus z. B. Ihrem Home-Verzeichnis zu gelangen. Sie können außerdem die bereitgestellten Funktionen zum Übertragen einer Fehlerseite bei einer nicht-existierenden Datei bzw. einer fehlerhaften Anfrage verwenden.
- Im Vorgabeverzeichnis befinden sich auch die Semaphor- und Puffermodule. Sie können allerdings auch Ihre Lösungen der letzten Übung verwenden.
- Die Schnittstellenvorgaben der Module `jbuffer` und `sem` finden Sie außerdem in einer Online-API-Beschreibung auf der Übungswebseite. Die Verwendung der Module `i4httools` und `2html` ist optional – Sie können beide auf eigene Bedürfnisse anpassen.

Hinweise zur Abgabe:

Bearbeitung: Zweiergruppen
Abgabe bis spätestens Donnerstag 21.07.2011, 18:00 Uhr