PR Rechnernetze und Netzwerkprogrammierung 621.801/621.802	WS 2017 / 2018
Institut für Informationstechnologie (ITEC)	Rainer/Timmerer
Übungsblatt 08	

U 8.1 Das UDP-Protokoll

Wenden Sie Ihr Wissen aus der Vorlesung an und sehen Sie sich zusätzlich RFC 768 (User Datagram Protocol - UDP) an und beantworten Sie folgende Fragen:

- Welche Funktionalität stellt UDP (nicht) zur Verfügung?
- Wie sehen die UDP-Header aus?
- Was ist ein Pseudo-Header?
- Wie groß können UDP-Segmente sinnvollerweise sein?

Geben sie Beispiele an, wo UDP verwendet wird (wurde).

U 8.2 aauTCP/IP-Referenzmodell: Transportschicht

Nachdem Sie im letzten Übungsblatt HTTP zwischen einem Client und einem Server auf der Applikationsebene realisiert haben, sollen Sie nun die Datenübertragung auf der Transportebene realisieren. Als Protokoll für den Transport soll (der Einfachheit halber) UDP verwendet werden.

Achtung: HTTP wird eigentlich über TCP realisiert; dies werden wir im nächsten Übungsblatt implementieren!

Dieses Protokoll dient nur zum Anwendungs-(De)Multiplexing bzw. zum Ver-/Entpacken der HTTP-Nachrichten (`HTTPClientMsg` und `HTTPServerMsg`).

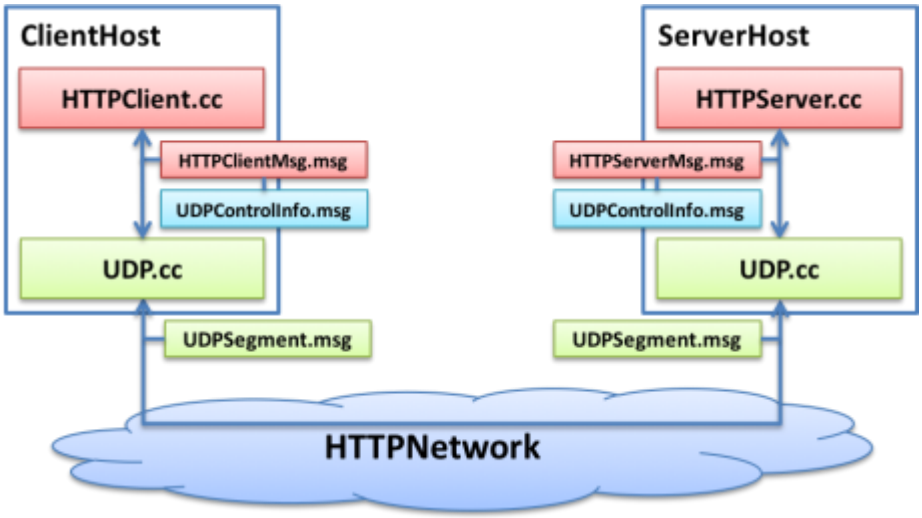

Abbildung 1. UDP-basierte HTTP Kommunikation auf der Transportschicht.

Jede HTTP-Nachricht sollte in einem UDP-Segment versendet werden. Verwenden Sie zum Implementieren der `UDPSegement.msg` (der Einfachheit halber) nur Quell- und Zielports (d.h., `int srcPort` und `int destPort`). Nutzen Sie zum Verpacken der HTTP-Nachrichten die Methode `encapsulate()`, zum Auspacken die Methode `decapsulate()`. **Dazu müssen alle Nachrichten vom Typ `packet` sein.**

Neben den HTTP-Nachrichten müssen auch zusätzliche Informationen zur Transportschicht übergeben werden. Dies geschieht mit der OMNeT++-Message `UDPControlInfo.msg` bestehend aus `InetAddress srcIP`, `int srcPort`, `InetAddress destIP` und `int destPort`. **Achtung**: Die `UDPControlInfo.msg` ist nicht teil des UDP-Segments und muss vor dem Versenden entfernt

werden bzw. an der Empfängerseite wieder erzeugt werden. **Hinweis:** Die `srcIP` bzw. `destIP` wird erst in den kommenden Übungsblättern (Netzwerkschicht) relevant und kann vereinfacht gehandhabt werden (z.B. 127.0.0.1).

Erweitern Sie die vorgegebene Datei, um alle notwendigen Informationen übergeben zu können und binden Sie diese in die Kommunikation zwischen den Schichten in den Dateien der Applikationsschicht ein (`HTTPServer.cc` und `HTTPClient.cc`).

***app*-Verzeichnis:**

- Wie im Übungsblatt 7
- Zur Implementierung der Funktionalität des Clients bzw. des Servers verwenden Sie bzw. erweitern Sie die Implementierung von Übungsblatt 7. D.h. beim Senden erzeugen Sie die `UDPControlInfo` und fügen Sie diese an die `HTTPClientMsg` bzw. `HTTPServerMsg` (mit `..->setControlInfo()`) an bzw. beim Empfangen können Sie diese wiederum entfernen (mit `..->removeControlInfo()`).

***udp*-Verzeichnis:**

- **UDP.ned**: Eine einfache Definition eines UDP-Moduls. Hier müssen Sie nichts ändern.
- **UDPSegment.msg**: Die Definition des UDP-Segments.
- **UDPControlInfo.msg**: Die Definition der UDP-ControlInfo. Hier müssen Sie nichts ändern.
- **UDP.cc** bzw. **UDP.h**: Die Implementierung UDP-Schicht. Achten Sie auf eine korrekte Unterscheidung zwischen `arrivedOn("fromUpperLayer")` und `arrivedOn("fromLowerLayer")` sowie darauf, dass die `UDPControlInfo` nicht über das Netzwerk verschickt wird, sondern nur zur Kommunikation zwischen den Schichten verwendet wird (d.h. vertikal, nicht horizontal).

***networks*-Verzeichnis:**

- **Client/ServerHost.ned**: Eine Definition des HTTP-Client/Server-Knotens. Definieren Sie diesen anhand von Abbildung 1.
- **HTTPNetwork.ned**: Die Definition des Netzwerks, das simuliert werden soll. Verwenden Sie die Netzwerkdefinition von Übungsblatt 7 und verbinden Sie ClientHost mit ServerHost.