Vorlesung Kommunikationssysteme
Wintersemester 2024/25

IP Datagramme

Christoph Lindemann

Comer Buch, Kapitel 22

Zeitplan

Nr.	Datum	Thema
01	18.10.24	Organisation und Internet Trends
02	25.10.24	Programmierung mobiler Anwendungen mit Android
	01.11.24	Keine Vorlesung
03	08.11.24	Protokolldesign und das Internet
04	15.11.24	Anwendungen und Netzwerkprogrammierung
05	22.11.24	LAN und Medienzugriff
06	29.11.24	Ethernet und drahtlose Netze
07	06.12.24	LAN Komponenten und WAN Technologien
08	13.12.24	Internetworking und Adressierung mit IP
09	20.12.24	IP Datagramme
10	10.01.25	Zusätzliche Protokolle und Technologien
11	17.01.25	User Datagram Protocol und Transmission Control Protocol
12	24.01.25	TCP Überlastkontrolle / Internet Routing und Routingprotokolle
13	31.01.25	Ausblick: TCP für Hochgeschwindigkeitsnetze
14	07.02.25	Review der Vorlesung

Überblick

Ziele:

- Einblick in den Aufbau, die Erzeugung und den Transport von IP Datagrammen

Themen:

- Aufbau von IP Datagrammen
- Headerformate von IP Datagrammen
- Weiterleitung von Datagrammen im Internet
- IP Kapselung
- IP Fragmentierung

IP Datagramme

Einführung

- Bisher:
  - Architektur des Internet
  - Adressierung im Internet

- Format der Pakete im Internet
- Kapselung von Datagrammen, Weiterleitung, Fragmentierung, Zusammensetzung von Fragmenten

Verbindungsloser Dienst

- Internet verwendet **verbindungslosen Dienst** als grundlegenden Übertragungsdienst
- **Zuverlässiger, verbindungsorientierter Dienst** baut auf diesen auf und nutzt diesen

Virtuelle Pakete (1)

- Verbindungsloser Dienst erweitert Prinzip von Paket Switching
  - Individuelle Pakete werden unabhängig voneinander im Internet übertragen
  - Paket enthält Informationen über Empfänger
- **Weiterleitung** der Pakete erfolgt durch **Router**
  - Nutzt Zieladresse der Pakete um nächsten Router auf Pfad zu finden

Virtuelle Pakete (2)

- Verwendung von Format von Hardware Frames für Internet Pakete?
  - Internet besteht aus heterogenen Netzwerken mit inkompatiblen Frameformaten
  - Adressierung in Netzwerken unterschiedlich, Router kann nicht einfach Frame Header anpassen
- Internet Protokoll definiert von Hardware unabhängiges Paketformat
- **Virtuelles, Universelles Paket**
  - Virtuell: Nicht an Hardware gebunden und von Hardware nicht erkannt/verstanden
  - Universell: Jeder Host/Router im Internet hat Protokollsoftware um Internet Pakete zu verstehen

IP Datagramm (1)

- **IP Datagramm**: Bezeichnung für Internet Pakete in TCP/IP Protokollen
- Besteht aus **Header** und **Payload** (Daten)
- Menge an Daten in Datagrammen ist nicht fest
- Wird von Sender/Applikation nach Zweck festgelegt
  - Beispiel: Jede Tasteneingabe als ein Datagramm oder große Datagramme für Videostreaming

IP Datagramm (2)

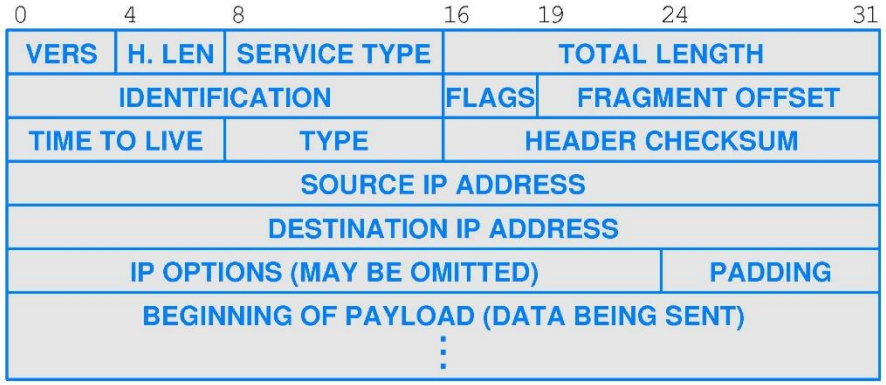
- IPv4 erlaubt bis zu 64K Oktetts als Payload
- Header meist viel kleiner als Payload
- Große Datagramme helfen Durchsatz zu maximieren (Header ist Overhead)
- Zu große Datagramme können zu Problemen führen, Erklärung später

Headerformate von IP Datagrammen

Format von IPv4 Datagramm Header (1)

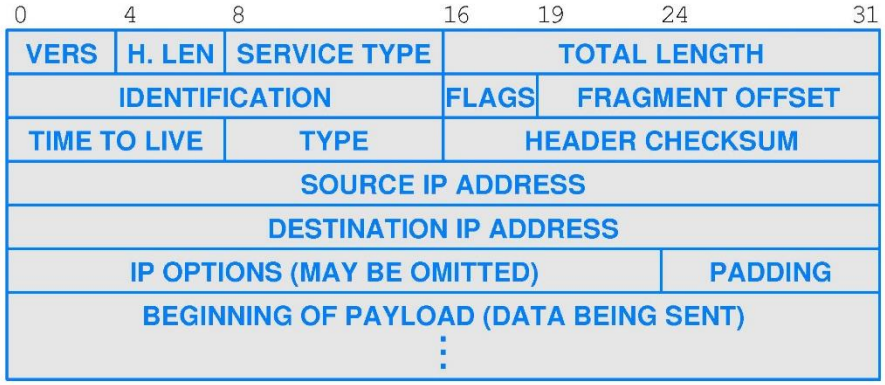
- Enthält Informationen um Datagramm weiterzuleiten
- Adresse des Sender (**Source**), Adresse des Empfänger (**Destination**), Typ der Daten in Payload
- Adressen sind IP Adressen (keine MAC Adressen)
  - Beziehen sich auf ursprünglichen Sender und endgültiges Ziel, nicht die Router dazwischen
- Felder haben feste Größe (effiziente Verarbeitung)

Format von IPv4 Datagramm Header (2)



- VERS: Version (für IPv4: „4“)
- H.LEN: Größe in 32 Bit Einheiten, falls keine Optionen: „5“
- SERVICE TYPE: Dienst des Datagramm (fast nie genutzt)
- TOTAL LENGTH: Gesamtmenge an Daten in Header und Payload
- IDENTIFICATION: Eindeutige Nummer um Fragmente zusammen zu setzen
- FLAGS: Ist Datagramm Fragment bzw. letztes Fragment?
- FRAGMENT OFFSET: Position des Fragment im vollständigen Datagramm
- TIME TO LIVE: Von Router dekrementiert, bei 0 von Router verworfen und Fehlermeldung

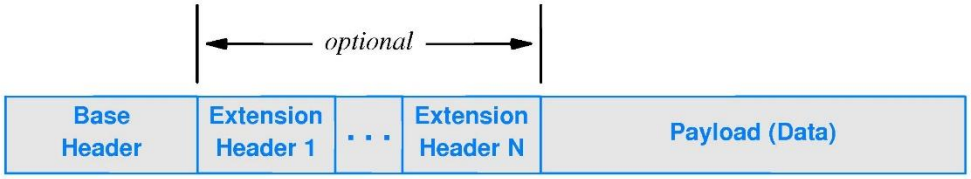
Format von IPv4 Datagramm Header (3)



- TYPE: Typ der Payload
- HEADER CHECKSUM: Checksumme der Felder in Header
- SOURCE IP ADDRESS: 32 Bit Adresse von original Sender (nicht des vorherigen Router)
- DESTINATION IP ADDRESS: 32 Bit Adresse des endgültigen Ziels (nicht des nächsten Router)
- IP OPTIONS: Steuerung von Weiterleitung/Verarbeitung, fast nie genutzt
- PADDING: Auffüllen auf 32 Bit Grenze

Format von IPv6 Datagramm Header

- Komplett neues Header Format
- Header geteilt in **Base Header** und kleinere, optionale **Extension Header**
- Wie in IPv4 folgt darauf Payload



IPv6 Base Header Format (1)

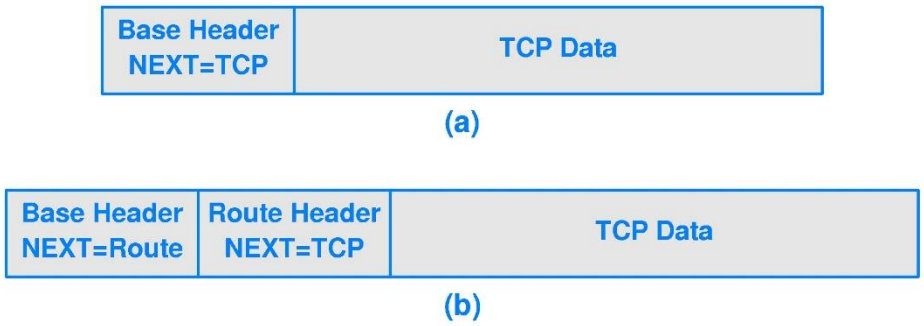
- Doppelt so groß wie IPv4 Header, enthält aber weniger Information
  - IP Adressen viermal so groß wie in IPv4
- Wie in IPv4 bezieht sich Source auf original Sender und Destination auf endgültiges Ziel (nicht Router dazwischen)

IPv6 Base Header Format (2)

- VERS: Version („6" für IPv6)
- TRAFFIC CLASS: Klasse des Dienst
- FLOW_LABEL: Für Label Switching, mittlerweile weniger wichtig
- PAYLOAD_LENGTH: Größe von Payload in Oktetts
- NEXT_HEADER: Typ der folgenden Information (Extension Header, Payload)

Extension Header

- Extension Header können feste Größe haben (festgelegt in Standard)
- Bei variabler Länge enthalten Sie Feld mit Längenangabe
- Base Header und Extension Header enthalten **NEXT_HEADER**



(a) Datagramm mit Base Header und TCP Payload
(b) Datagramm mit zusätzlichem Route Extension Header

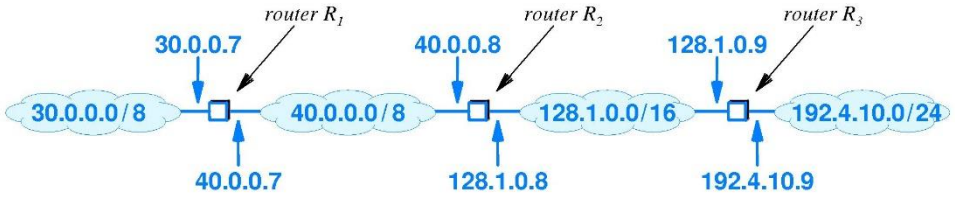
Weiterleitung von IP Datagrammen im Internet

Weiterleitung von Datagrammen (1)

- Internet nutzt **Next-Hop Forwarding**
  - Router auf Pfad nutzt Zieladresse aus Header um nächsten Hop zu finden
  - Leitet an Hop weiter (weiterer Router oder Ziel)
- IP Router nutzen **Weiterleitungstabelle (Forwarding Table)**
  - Initialisierung bei Start des Router, Aktualisierung bei Änderungen von Topologie
- Enthält Einträge mit Ziel und nächstem Hop

Weiterleitung von Datagrammen (2)

- Größe von Weiterleitungstabelle proportional zur Anzahl an Netzwerken, nicht Hosts
  - Ziele als Netzwerk angegeben



(a)

Destination	Mask	Next Hop
30.0.0.0	255.0.0.0	40.0.0.7
40.0.0.0	255.0.0.0	deliver direct
128.1.0.0	255.255.0.0	deliver direct
192.4.10.0	255.255.255.0	128.1.0.9

(b)

(a) IPv4 Internet mit vier Netzen
(b) Weiterleitungstabelle in Router R_2

Netzwerk Präfix und Matching (1)

- Maske in Weiterleitungstabelle um Netzwerkanteil der Adresse zu erhalten
- Zu Ziel IP Adresse D muss Eintrag in Weiterleitungstabelle gefunden werden
- Mit Maske wird Präfix von D extrahiert und mit Destination Feld in Weiterleitungstabelle verglichen
- Bei Übereinstimmung Weiterleitung an Next Hop

Netzwerk Präfix und Matching (2)

- Effizienter Vergleich durch Bitmasken
- Test, ob (Mask[i] & D) == Destination [i]
- Weiterleitungstabelle aus letztem Beispiel
  - Ziel 192.4.10.3 erreicht Router R_2
  - Vergleich für erste drei Einträge schlagen fehl
- Weiterleitungstabellen in Praxis sehr groß
- Kann Default Eintrag für nicht aufgeführte Ziele enthalten

Netzwerk Präfix und Matching (3)

- **Hostspezifische Routen** möglich (Anderer Pfad als andere Hosts im Netz)
  - Maske deckt komplette Adresse ab
- Adressmasken können überlappen
  - Einträge für 128.10.0.0/16 und 128.10.2.0/24
  - Datagramm für 128.10.2.3 trifft für beide
- Weiterleitung nutzt längsten Präfix (**Longest Prefix Match**)
- Einträge mit längsten Präfix zuerst von Software analysiert
  - Eintrag für 128.10.2.0/24 wird genutzt

Best-Effort Delivery

- IPv4 und IPv6 verwenden **Best-Effort Delivery**

- Mögliche Probleme
  - Duplikate der Datagramme
  - Datagramme verspätet oder in falscher Reihenfolge
  - Fehlerhafte Daten
  - Verlorene Datagramme

- Gründe
  - Hardware anfällig für Interferenz durch Rauschen
  - Verschiedene Pfade der Datagramme

- Zusätzliche Protokolle um Fehler zu beheben

IP Kapselung

IP Kapselung (1)

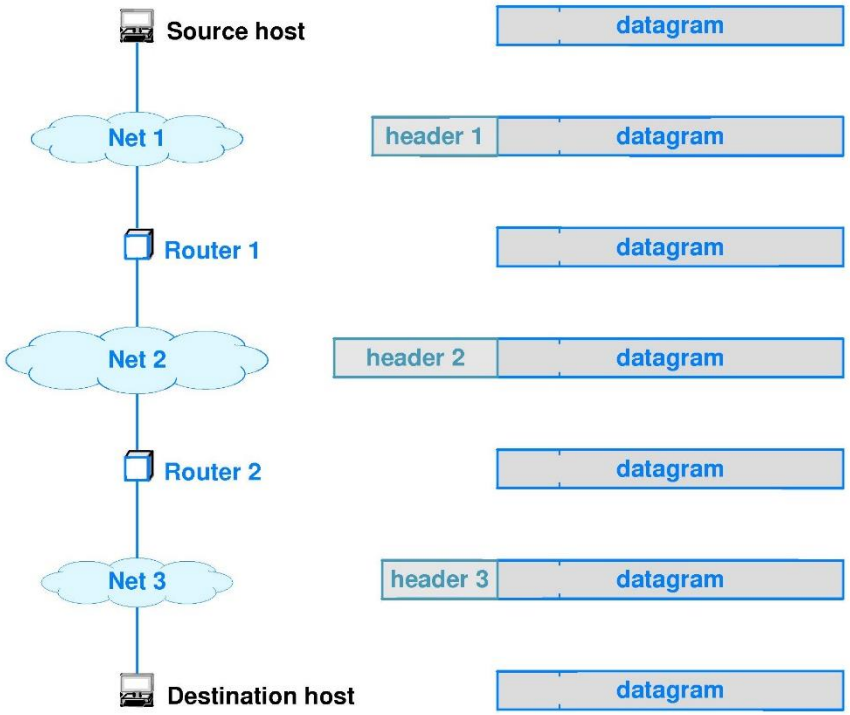
- Hardware versteht Datagramm Format nicht
- IP Datagramm wird als Teil der Payload in Frame **gekapselt**
- Hardware analysiert oder ändert Daten der Payload nicht
- Typ in Frame Header gibt an, ob Payload IP Datagramm enthält
- Frame enthält MAC des nächsten Ziel (wird von Sender zu IP Adresse des Next Hop ermittelt)

IP Kapselung (2)

- Kapselung erfolgt für jedes Netzwerk
- Router ermittelt nächsten Hop
- Kapselt Datagramm in Frame
- Übertragt Frame in physisches Netzwerk
- Empfänger entfernt Frame und kapselt neu für nächstes Netzwerk, falls nicht selbst endgültiger Empfänger

IP Kapselung (3)

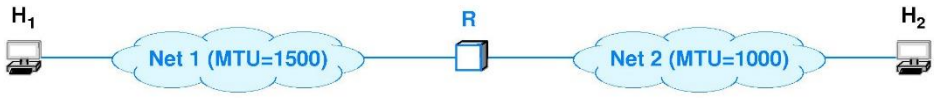
- Verschiedene physische Netzwerke → unterschiedliche Frameformate und Headergrößen
- Beispiel:
  - Netz 1 Ethernet mit passendem Frame
  - Netz 2 Wi-Fi Netzwerk mit passendem Frame



IP Fragmentierung

MTU und Fragmentierung (1)

- **Maximum Transmission Unit (MTU)**: Maximalmenge an Daten pro Frame in einer Hardwaretechnologie
- Datagramm muss kleiner oder gleich Netzwerk MTU sein
- Router kann Netzwerke mit verschiedenen MTUs verbinden
  → Datagramm kann zu groß für nächstes Netzwerk sein
- Falls Host H_1 Datagramm zu Host H_2 mit 1500 Oktetts sendet, kann Router Datagramm nicht kapseln



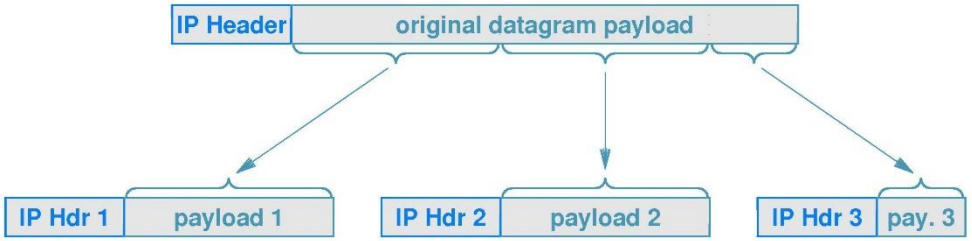
Router verbindet Netzwerke mit verschiedenen MTUs

MTU und Fragmentierung (2)

- **Fragmentierung**: Unterteilt Datagramm in kleinere Teile (**Fragmente**) und sendet jedes in separatem Frame
- In IPv4 bestimmt Router Fragmentierung
- In IPv6 muss sendender Host Fragmentierung durchführen
- IPv4 Fragment hat gleiches Format wie IPv4 Datagramme
  - FLAGS Field bestimmt, ob Datagramm Fragment ist
  - FRAGMENT OFFSET Feld hilft bei Zusammensetzung im endgültigen Ziel

MTU und Fragmentierung (3)

- Router berechnet Maximalmenge an Daten pro Fragment und Anzahl an Fragmente
- Nutzt dazu Netzwerk MTU und Header Größe
- Kopiert Felder wie IP SOURCE und IP DESTINATION in Fragment Header



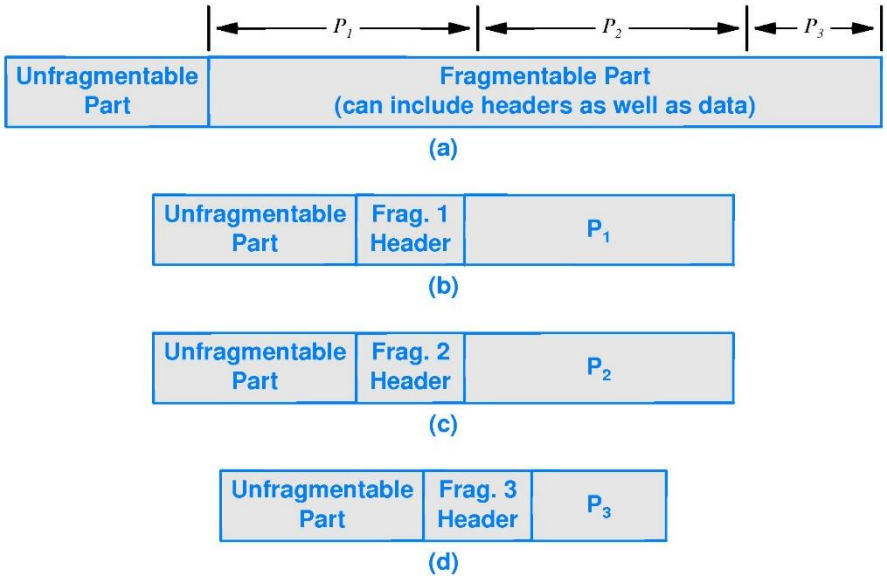
IPv4 Datagramm auf drei Fragmente aufgeteilt, letztes Fragment ist kleiner

Fragmentierung in IPv6 (1)

- Wie in IPv4 wird Teil des Datagramm Header in jedes Fragment kopiert, Payload Länge wird angepasst
- Keine zusätzlichen Felder für Fragmentierung im Base Header, stehen in **Fragment Extension Header**
  - Existenz des Header zeigt an, dass Datagramm Fragment ist
  - Enthält gleiche Informationen wie Fragment Informationen in IPv4 Header

Fragmentierung in IPv6 (2)

- Header werden unterteilt in fragmentierbar und nicht fragmentierbar
  - Nicht fragmentierbare Header werden in jedes Fragment kopiert
  - Werden von zwischenliegenden Routern benötigt



Fragmentierung in IPv6 (3)

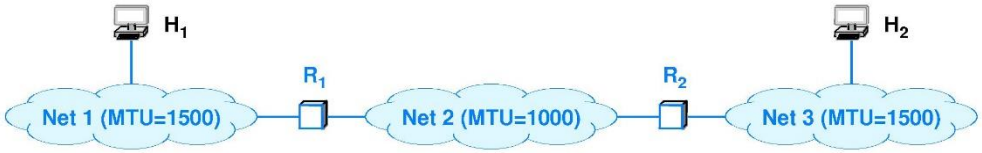
- Ursprünglicher Sender muss über Fragmentierung entscheiden
- Router verwirft Datagramm und sendet Fehler mit ICMP
- **Path MTU**: Minimale MTU des Pfad
- Host muss MTU von jedem Netzwerk auf Pfad lernen → **Path MTU Discovery**
  - Host wählt initiale Größe von Datagramm (z.B. Ethernet MTU 1500) und sendet
  - Erhält er Fehler von einem Router, reduziert er MTU bis Übertragung funktioniert
  - Maximalgröße von nachfolgenden Datagrammen beschränkt

Zusammenfügen von Fragmenten (1)

- Empfänger erkennt an Flags, ob Datagramm Fragment ist
- Fragmente haben dieselbe Zieladresse
- IP spezifiziert, dass nur der Empfänger die Fragmente zusammenfügen soll (nicht die Router)
  - Weniger Zustandsinformationen in Router
  - Routen können sich dynamisch ändern

Zusammenfügen von Fragmenten (2)

- Host H_1 sendet IPv4 Datagramm mit 1500 Oktetts zu Host H_2
- Router R_1 zerlegt es in zwei Fragmente und sendet an R_2
- Wird IPv6 verwendet, zerlegt Host H_1 Datagramm
- Router R_2 fügt Fragmente nicht zusammen



Drei Netzwerke verbunden mit zwei Router

Sammeln der Fragmente

- Fragmente können verloren gehen oder in falscher Reihenfolge ankommen
- Fragmente von verschiedenen Datagrammen können bei Empfänger ankommen
- Empfänger bestimmt Datagramm eines Fragment anhand Adresse des Senders und IDENTIFICATION Feld
- FRAGMENT OFFSET gibt Stelle des Fragment in ursprünglichen Datagramm an

Verlust von Fragmenten

- Bei Verlust eines Fragments kann Datagramm nicht zusammengefügt werden
- Empfänger muss Fragmente speichern, da Fragment verspätet ankommen kann
- IP spezifiziert maximale Zeit bis Fragmente verworfen werden → Reduziert Speicherbedarf
  - Bei Erhalt von erstem Fragment startet Empfänger **Reassembly Timer**
  - Läuft Timer ab und nicht alle Fragmente wurden erhalten, werden alle Fragmente verworfen
  - Kein Mechanismus um Sender über fehlende Fragmente zu informieren

Fragmentierung von IPv4 Fragmenten

- Fragment kann Netzwerk mit kleinerer MTU erreichen (In IPv6 durch MTU Path Discovery verhindert)
- Fragment kann wie ein normales Datagramm weiter fragmentiert werden
- Empfänger muss nicht wissen, ob Fragmente weiter fragmentiert wurden
  - Keine zusätzlichen Informationen in Header notwendig
  - Spart CPU Zeit

Zusammenfassung

- Datagramm Header enthält Informationen um Datagramm zu Ziel zu übertragen
- IP Software in Router verwendet Weiterleitungstabelle um nächsten Hop zu finden
- Nur Adresse von endgültigem Ziel und nicht nächstem Hop steht in Datagramm Header
- IP Datagramm wird in Frame gekapselt
- Unterschiedliche MTU in verschiedenen Netzen → Datagramme können fragmentiert werden