UDP – User Datagramm Protokoll

Marco Gerland
Janina de Jong

Einführung

- IP Datagramme werden durchs Internet geroutet abh. von der IP-Adresse
- Anhand der Ziel-IP-Adresse wird nur der Computer angesprochen nicht der Benutzer oder das jeweilige Programm
- IP-Erweiterung um verschiedene Ziele zu unterscheiden
- Programme müssen unabhängig senden und empfangen können

Das korrekte Ziel identifizieren I

- Die meisten BS unterstützen Multi-Tasking
- Laufendes Programm, BS-Funktion = Prozess
- Jeder Prozess kann Ziel einer Nachricht sein
- Probleme:
  1. Prozesse dynamisch erzeugt und zerstört
  -> Sender können nicht identifizieren

Das korrekte Ziel identifizieren II

2. Möglichkeit Prozesse zu ersetzen (z.B. Reboot), ohne die Sender informieren zu müssen

3. Das Ziel muss anhand der Funktion identifiziert werden, ohne den Prozess zu kennen, der die Funktion bereitstellt.
   (z.B. Fileserver)

- noch wichtiger : Ein Prozess mit 2 oder mehr Funktionen muss entscheiden können welche Funktion benötigt wird

Das korrekte Ziel identifizieren III

- Statt Prozess als Empfänger
  -> Ziel-Computer mit vielen Zielpunkten sog. Protokoll-Ports
- Vorstellung: Gang mit vielen Türen
- Jeder Zielpunkt hat eine positive Zahl
- BS: Mechanismus für Prozesse um die Ports zu identifizieren und zu benutzen

Das korrekte Ziel identifizieren IV

- Die meisten BS bieten gleichz. Zugang zu Ports
- Problem: Prozesse stoppen während der Zugriffsoperation
- Beispiel: Prozess möchte Daten von außen.
- Ports sind gepuffert
- BS-eigene Protokolle: Warteschlangen für Daten
- Für Kommunikation notwendig: Source-IP, Ziel-IP und Port

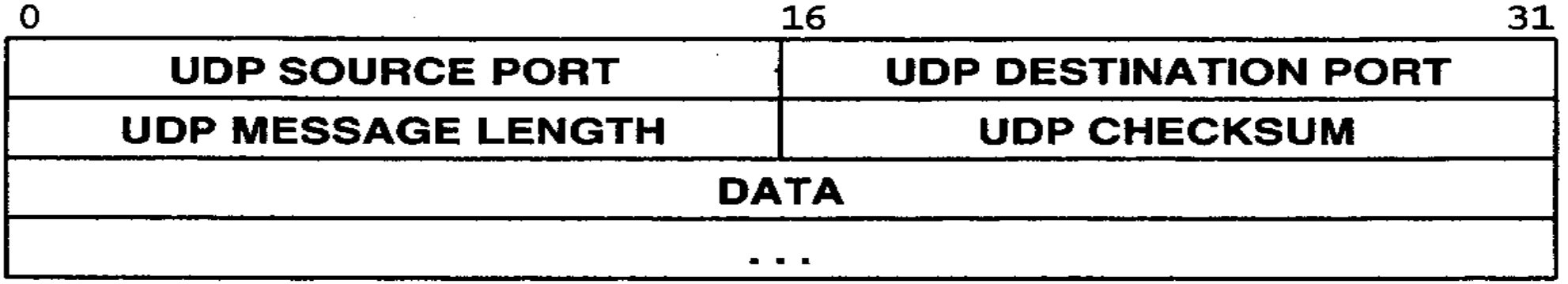
Das UDP

- Mechanismus zur Übertragung von Daten zw. Programmen
- Stellt Ports zur Verfügung um zwischen Programmen zu unterscheiden
- UDP-Nachricht: Ziel-Port, Quell-Port
- UDP nutzt IP: unzuverlässig, verbindungslos, keine Überprüfung, keine Flusskontolle

Das UDP (II)

- Pakete:
  - gehen verloren
  - werden dupliziert
  - kommen ungeordnet an
  - kommen schneller als Verarbeitung möglich
- Programme haben Verantwortung
- Oft bei Programmierung vernachlässigt
- Tests in hochverfügbaren Netzen

UDP-Format



- Header: 4 Felder à 16 Bit
- Dest-Port 16 Bit
  Source-Port optional
- Length in Octets (min 8)
- Prüfsumme optional
  0 = nicht berechnet

UDP-Format

- IP berechnet keine Prüfsumme über Daten
- UDP-Prüfsumme einzige Möglichkeit Daten auf Integrität zu prüfen.
- Berechnete Prüfsumme 0? (1111111111111111)
- Gleicher Algorithmus wie IP:
  16-Bit Worte -> 1'er Komplemente
  Summe der Einerkomplemente -> Einerkomplement

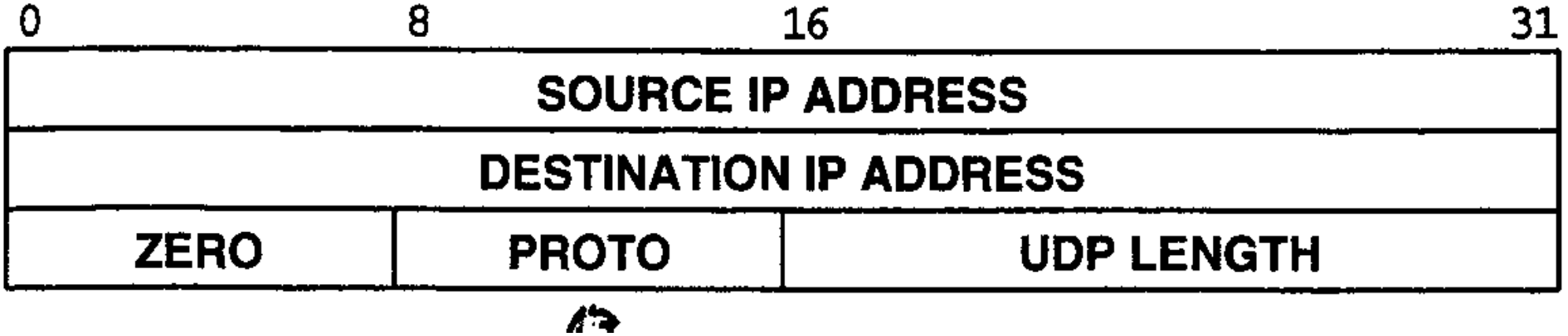
UDP-Format

- Also:
  nicht berechnet: 0000000000000000 = 0
  berechnet: 1111111111111111 = 0
- Näheres zum 1'er Komplement in Technische Informatik I

UDP-Pseudo-Header

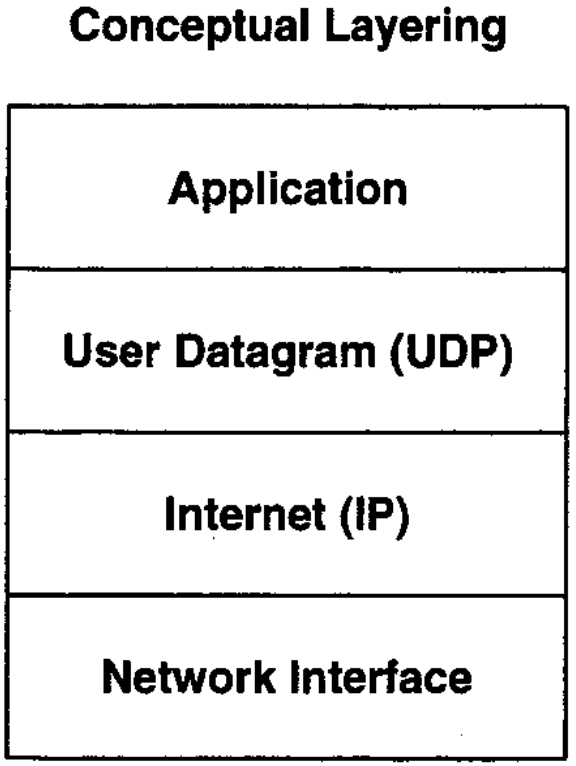
- Pseudo-Header wird vor den eigentlichen Header gehängt
- Bei ungerader Anzahl von 16 Bits werden 0 angehängt
- Wird nicht übertragen und auch nicht in Length eingetragen
- Berechnung: 0 in Checksumme dann 16 Bit-Komplement-summe aus gesamten Objekt (P-Header, Header, Data)

UDP-Pseudo Header



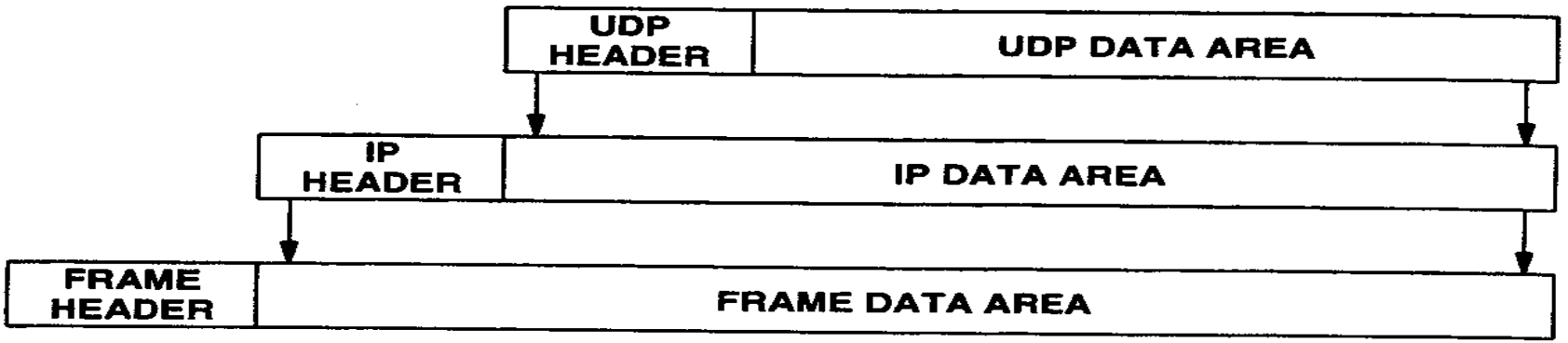
- IP-Adressen
- Proto: Protokoll Typ = 17 (UDP)
- Length: Länge des Datagrams ohne Pseudo-Header
- Empfänger muss Felder aus IP-Header extrahieren und Checksumme berechnen

UDP im Schichtenmodell



- Applikationen greifen auf UDP zu
- UDP benutzt IP zum senden und empfangen
- UDP Nachricht in IP gekapselt

UDP im Schichtenmodell



- Jede Schicht stellt einen Header voran
- Empfänger: jede Schicht entfernt einen Header
- IP zuständig für Transport zwischen zwei Hosts; UDP zuständig für Unterscheidung von Quellen und Zielen auf einem Rechner

Schichten und die UDP Checksumme

- Widerspruch:
  Pseudo-Header hat Felder aus IP?
- IP muss bekannt sein, für UDP-Versand. User?
- Source IP hängt von der Route ab, die IP wählt
- 1. UDP->IP->P-Header->Checksumme
  ->verwerfen->IP
  2. UDP->UDP-Datagramm->IP-Datagramm->
  Felder speichern->Checksumme->IP

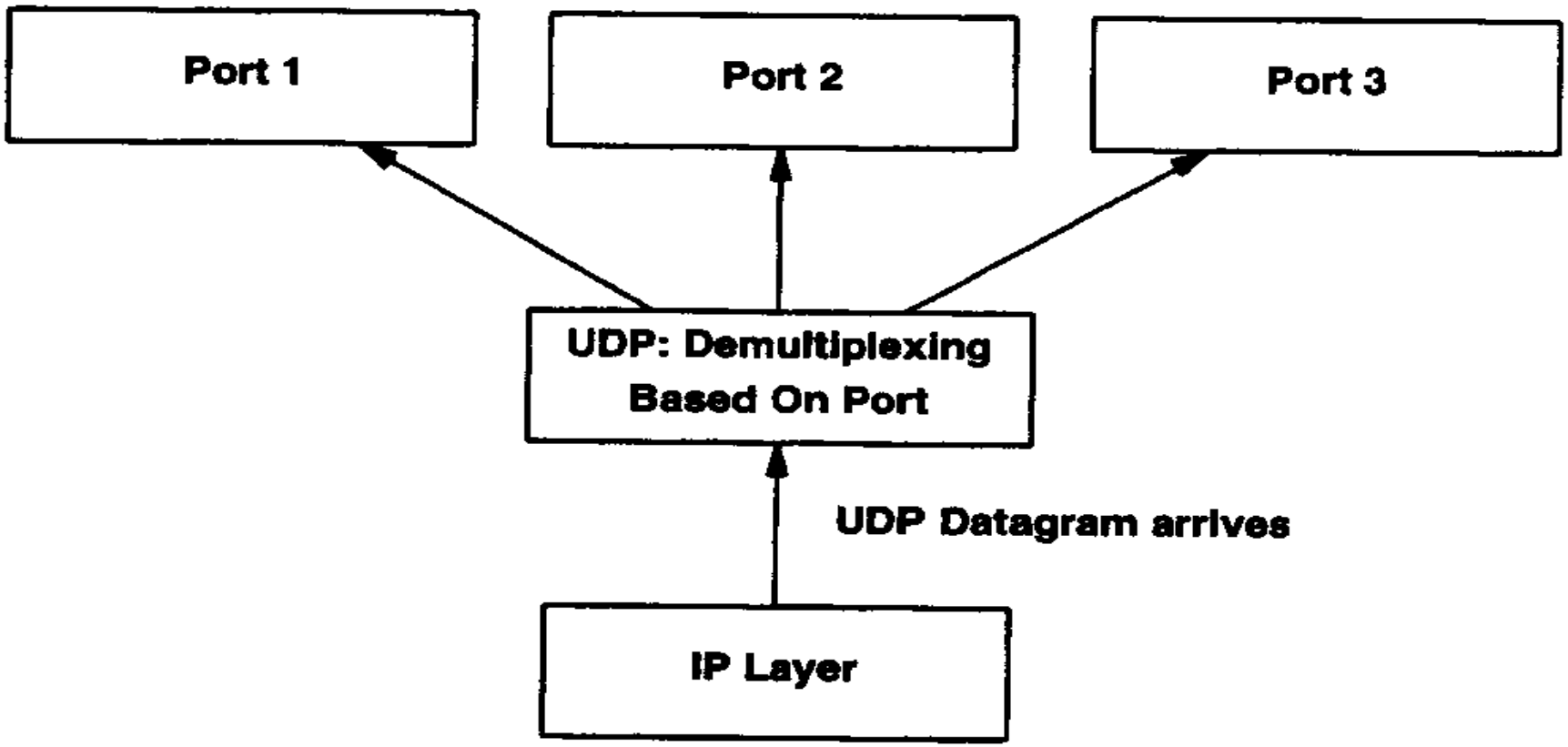
Schichten und UDP-Checksumme

- Keine strenge Trennung von UDP und IP (UDP benötigt IP-Adressen)
- Keine doppelte Erzeugung bzw. Verwendung von Feldern

Multiplexing, Demultiplexing und Ports

- Multiplexing:
  Zuweisung von Anwendungs-Nachrichten zu Netzwerk-Frames
- Demultiplexing:
  Zuweisung ankommender Netzwerk-Frames zum Anwendungsprozess
- Bei UDP über den Port-Mechanismus geregelt
- Jede Applikation bekommt Port vom BS vor dem senden

Multiplexing, Demultiplexing und Ports



Multiplexing, Demultiplexing und Ports

- Warteschlange:
  Jede Applikation bekommt vom BS eine Schlange für Antworten
- UDP Nachricht kommt an:
  UDP Port vorhanden und in Benutzung?
  Nein: ICMP port unreachable
  Ja: Weiterleitung zu entsprechendem Port
- Port voll: Datagramm wird verworfen

Reservierte und freie UDP Port-Nummern

- Wie werden Ports zugewiesen?
- Beispiel:
  Computer A möchte File von Computer B via FTP. Welcher Port wird benutzt auf A und B?
- 2 Modelle:
  1. Universelle Zuordnung:
     Zentrale Instanz,
     zentrale Liste von Ports
     well-kown-ports

Reservierte und freie UDP Port-Nummern

- 2. Dynamische Verteilung:
  es wird eine Anfrage nach Ports gestellt
  Die Zielmaschine gibt den richtigen Port an
- TCP/IP vereint das Beste aus beiden Welten: manche Ports fest zugeordnet, die meisten frei verfügbar

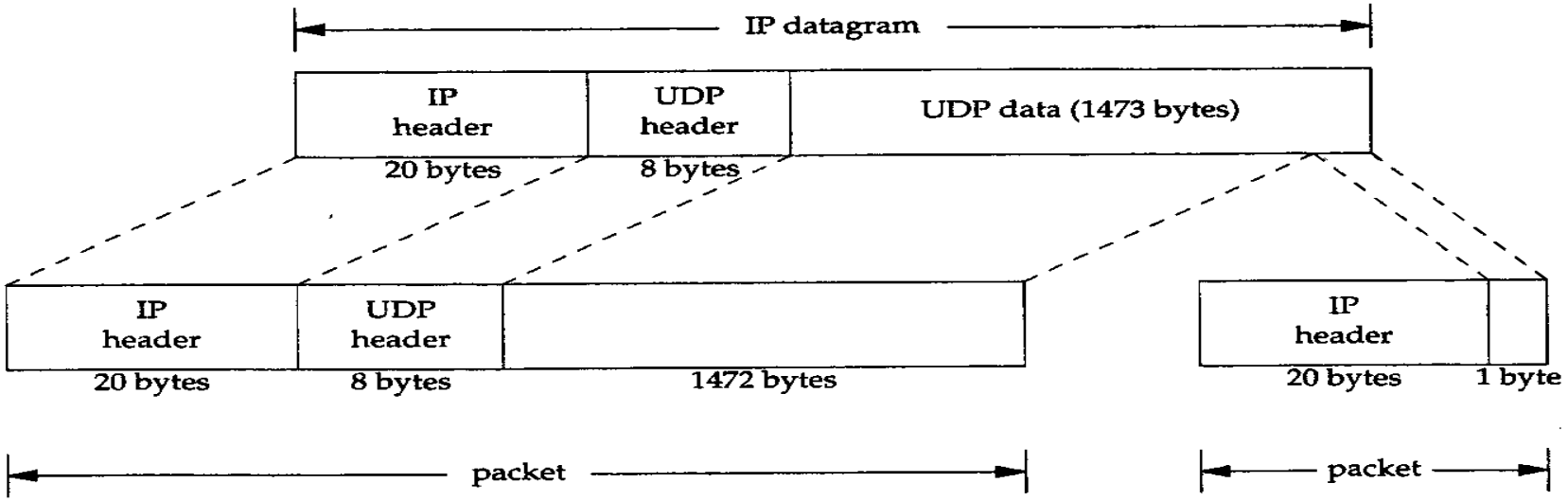
UDP-Ports

Decimal	Keyword	UNIX Keyword	Description
0	-	-	Reserved
7	ECHO	echo	Echo
9	DISCARD	discard	Discard
11	USERS	systat	Active Users
13	DAYTIME	daytime	Daytime
15	-	netstat	Network status program
17	QUOTE	qotd	Quote of the Day
19	CHARGEN	chargen	Character Generator
37	TIME	time	Time
42	NAMESERVER	name	Host Name Server
43	NICNAME	whois	Who Is
53	DOMAIN	nameserver	Domain Name Server
67	BOOTPS	bootps	BOOTP or DHCP Server
68	BOOTPC	bootpc	BOOTP or DHCP Client
69	TFTP	tftp	Trivial File Transfer
88	KERBEROS	kerberos	Kerberos Security Service
111	SUNRPC	sunrpc	Sun Remote Procedure Call
123	NTP	ntp	Network Time Protocol
161	-	snmp	Simple Network Management Protocol
162	-	snmp-trap	SNMP traps
512	-	biff	UNIX comsat
513	-	who	UNIX rwho daemon
514	-	syslog	System log
525	-	timed	Time daemon

Fragmentation:

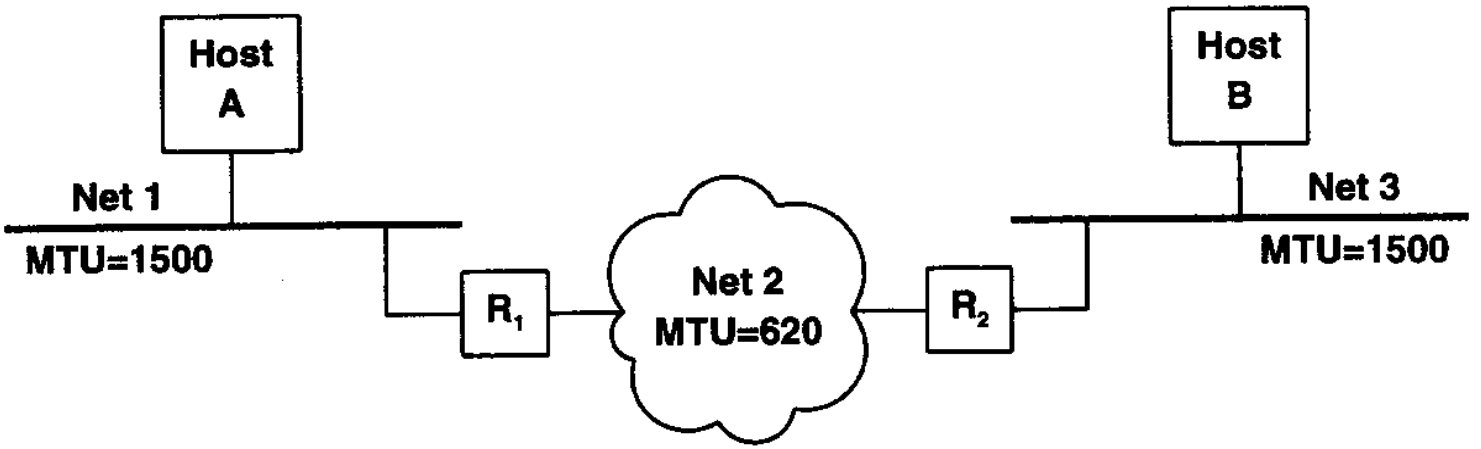
- Idealer Fall: IP Datagramm passt in 1 Frame
- MTU (Maximum Transfer Unit)
- Wichtige IP-Headerfelder: Identification, Flags, Fragment offset, Total length -> Länge des Fragments
- z.B.: „don't fragment“- bit -> Zu großes Paket wird verworfen (ICMP Error)

Fragmentation



- Jedes Fragment eigenen IP-Header
- ABER: UDP-Header nicht dupliziert
- Mindestgröße 20 Bytes (IP-Header)

Fragmentation



- Sender weiß nicht alle MTU's der Netze, durch die geroutet wird
- Fragmentierung vom Sender oder vom Router
- Wo/Wann findet die Zusammenführung der Fragmente statt?

Zusammenfassung

- Die meisten Systeme erlauben simultane Abarbeitung von Programmen-> Prozesse
- UDP unterscheidet zwischen Prozessen an Hand von Protokoll-Ports
- Die Port-Nummern identifizieren die Quelle und das Ziel
- Manche Ports sind fest vergeben und andere frei verfügbar

Zusammenfassung (II)

- UDP gibt Programmen die Möglichkeit über IP zu kommunzieren
- Daher Verlust, Duplizierung und Unordung der Pakete möglich: Die Programme müssen das berücksichtigen
- Viele Programme, die UDP nutzen, arbeiten nicht richtig über ein großes Netz aus diesen Gründen

Zusammenfassung (III)

- In der Protokoll-Schicht liegt UDP in der Transport-Schicht über IP und unter der Anwendungsschicht
- Obwohl eigentlich unabhängig, arbeitet es mit IP zusammen -> Prüfsumme

Quellen

- TCP/IP Illustrated
  W. Richard Stevens
  Addison-Wesley, 1994-1996
  Volume 1: The Protocols
- Internetworking with TCP/IP
  Douglas Comer, David L. Stevens
  Prentice Hall, 2000
  Volume 1: Principles, Protocols and Architectures

Fragen und Manöverkritik

Vielen Dank
für eure
Aufmerksamkeit!