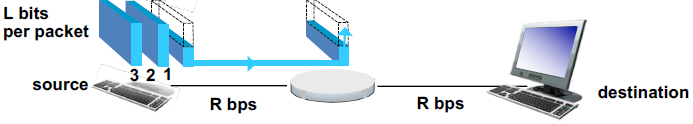
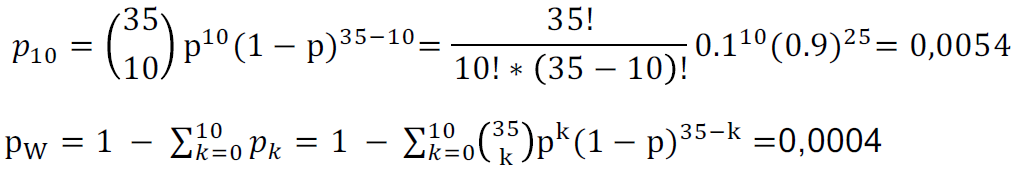
Rechnernetze Zusammenfassung

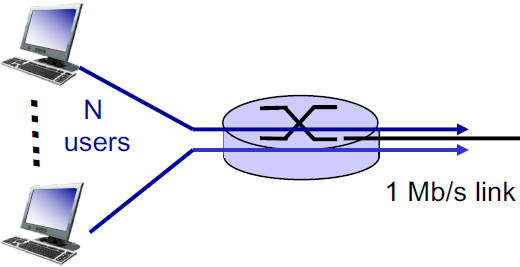
* Allgemeines
  + Schichtenmodell:
    - Bitübertragung: Bits via physikalisches Medium
    - Sicherung: Rahmen (Frame) via PPP, Ethernet, WLAN, …
    - Netzwerk: Datagramm via IP, OSPF, BGP, …
    - Transport: Segment via TCP, UDP, …
    - Anwendung: Nachricht via http, SMTP, FTP, …
  + Eine Netzwerkschicht stellt den höheren Schichten eigene Dienste zur Verfügung
    - Sicherheit: Verschlüsselung, Datenintegritätssicherung, Authentifikation, …
    - Zugriff über APIs
  + Zwei weitere Schichten im ISO/OSI-Modell: Sitzungsschicht, Darstellungsschicht
  + Grundlegende Protokollfunktionen
    - Fehlerkontrolle
    - Flusskontrolle
      * Vermeiden von Überlastung eines Knotens
    - Überlastkontrolle
      * Vermeiden von Überlast des Netzwerks
    - Segmentierung und Reassemblierung
    - Multiplexen
      * Gemeinsame Nutzung einer einzigen Verbindung durch mehrere gleiche Verbindungen einer höheren Schicht
    - Verbindungsaufbau/-abbau
  + Paketvermittlung
    - Store-and-Forward:
      * Übertragung eines Pakets pro Link:



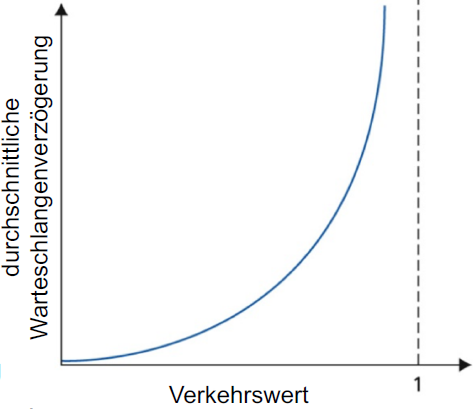
* + - * Wahrscheinlichkeit, dass ein User etwas sendet:
      * Wahrscheinlichkeit, dass genau (mit ) User zeitgleich aktiv sind:



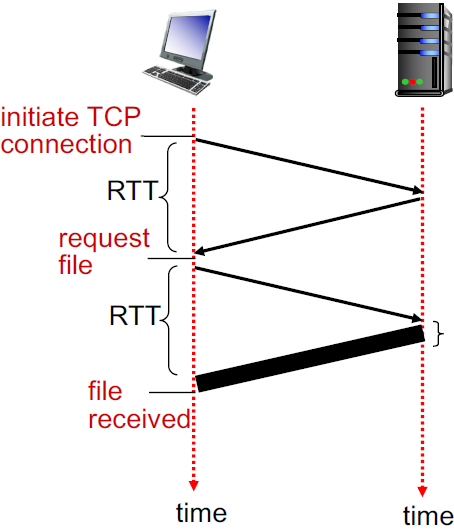
* + Leitungsvermittlung
    - Pro User wenn er aktiv ist
    - Max. Anzahl User:



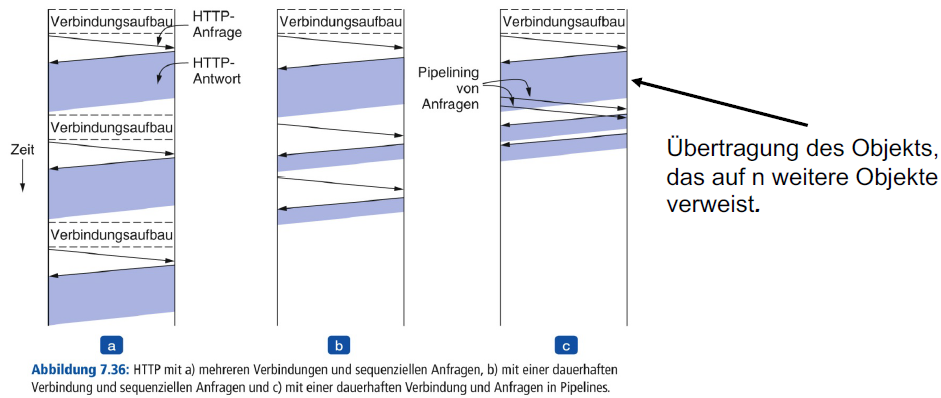
* + Paketvermittlung vs. Leitungsvermittlung
    - Paketvermittlung erlaubt bessere Auslastung des Netzes
      * Keine „Stillephasen“
      * Dynamische Aufteilung der Netzwerkressourcen
      * Kein Overhead für Leitungsreservierung
    - Bei Paketvermittlung kann
      * Paketverzögerung und -verlust auftreten
      * Zuverlässiger Datentransfer und Überlastkontrolle müssen ggf. von höheren Schichten geleistet werden
      * Keine garantierten Paketzustellzeiten garantieren
    - Lösung: „virtuelle Kanäle“
  + Paketverlust
    - * Einzelne Bits verfälscht durch z.B. geringe Signalstärke im WLAN
      * Empfänger muss dies erkennen
      * Teilweise können Fehler korrigiert werden
  + Puffer-Überlauf
    - Warteschlange einer Leitung hat endlich viele Pufferplätze
    - Verwerfen von Paketen wenn Puffer vollständig belegt
  + Verzögerungen
    - Processing
      * Prüfung auf Bitfehler, Routing-Entscheidung; wenige Mikrosekunden
    - Queuing
      * Wartezeit im Puffer der Ausgangleitung; Hängt von Verkehrslast ab
    - Transmission
      * Zeit für das Senden von Bits (Übertragungsverzögerung)
        + = Bandbreite der Leitung [bit/s]
        + = Paketgröße [bit]
    - Propagation
      * Laufzeit eines Bits
      * Ausbreitungsverzögerung eines Bits:
      * = Länge der physikalischen Leitung bzw. Entfernung [m]
      * = Ausbreitungsgeschwindigkeit:
    - Verkehrswert



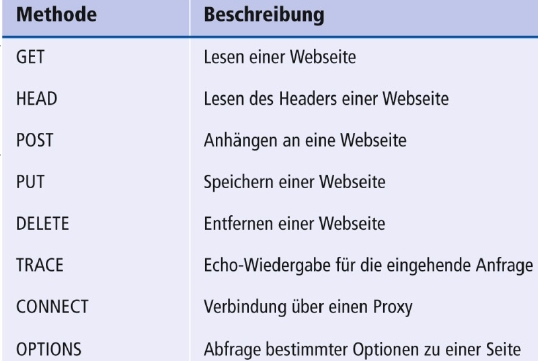
* + - Ende zu Ende Latenz: Summe aus
      * Verarbeitungszeit
      * Warteschlangenverzögerung
      * Übertragungszeit
      * Ausbreitungsverzögerung
* Anwendungsschicht
  + Peer-to-Peer (P2P)
    - Keine Server
    - Beliebige Endsysteme („Peers“)
      * Bieten und fordern Dienste bei anderen Peers an
      * Bringen neue Services ein
      * Dynamische IP
    - Beispiele: „Skype“, Instant Messaging
  + API - „Application Programming Interface“
    - Schnittstelle zwischen Anwendungsschicht und Transportschicht
    - Internet API: Socket
    - Identifizieren eines Partnerprozesses auf anderem Rechner
      * IP des anderen Rechners
      * Portnummer
  + Protokolle
    - TCP: Transmission Control Protocol
      * Stellt zuverlässige, reihenfolge-erhaltende Byte-Übertragung („pipe“) zwischen Client und Server bereit
      * Client kontaktiert Server durch
        + Erzeugen eines TCP-Sockets
        + Spezifizierung der IP + Portnummer des Server Prozesses
        + Verbindungsaufbau durch 3-Wege-Handshake
        + Server erstellt Socket für die Kommunikation
      * Liefert keine Security Features
        + Früher SSL, heute TLS
        + #TODO
    - UDP: User Datagram Protocol
      * Überträgt über unzuverlässigen Datentransfer eine Gruppe von Bytes (Datagramme) zwischen Client und Server
      * Vertauschung und Verlust von Segmenten möglich
      * Verwaltungsaufwand
        + + Weniger Overhead für direkte Kommunikation
        + – Client fügt jeder Nachricht Ziel-IP und Portnummer an
        + – Server muss IP und Portnummer des Clients aus jeder Nachricht extrahieren, wenn er Antwort adressieren möchte
    - SMTP: Simple Mail Transfer Protocol
      * Weiterleitung von Mails
      * Verwendet TCP über Port 25
      * Keine Sicherheit
        + ESMTP: Extended Simple Mail Transfer Protocol
        + Authentifikation
        + Verschlüsselung der TCP-Verbindung
      * Format
        + Header: From, To, CC, Subject
        + Body: Eigentliche Nachricht
      * MIME: Multimedia Mail Extension
        + Zusatzinformationen über den Typ der Daten der Mail
    - HTTP(S): Hypertext Transfer Protocol (Secure)
      * Webseiten: Skalierbarkeit, Effiziente Übertragung, Security, …
      * Verwendet TCP
      * Zustandslos: Keine Speicherung von vergangenen Anfragen
      * Nicht persistente Verbindung



* + - * Persistente Verbindung



* + - * Request
        + GET, POST, HEAD, PUT, DELETE, TRACE, CONNECT commands



* + - * + Header lines
        + Carriage return (Ende des Headers \r\n)
        + ASCII
        + Statuscodes

200 OK

400 Bad Request

404 Not Found

* + - * Cookies
        + Speichern von Zuständen über mehrere Transaktionen
        + Identifikation des Benutzers
        + Auf dem Client des Users gespeichert
      * Cache
        + Client-Seite

Expires Datum existiert und liegt in der Zukunft 🡪 Seite wird direkt aus dem Cache geholt, kein Netzwerkzugriff

If-modified-since Datum wurde abgespeichert

* + - * + Server-Seite

Proxy Server: Server, der im Namen eines Webservers antwortet

Geringere Antwortzeit

Geringerer Verkehr ins Internet

Sicherheitsvorteile für geschl. Netze

Client sendet alle Anfragen an Proxy Server

Falls Objekt im Web-Cache des Proxy Servers vorhanden 🡪 Gib gecachtes Objekt zurück, ansonsten 🡪 Fordere Orginal-Server an

* + - DNS: Domain Name System
      * Verteilte Datenbank: Hierarchie von Name-Servern
      * Abbildung von Hostnamen auf IPs
      * Client-Server Paradigma
      * Verwendet UDP, Port 53
      * Mehrere Aliasnamen für Hostname
      * Lastverteilung bei replizierten Servern
      * Kein zentraler DNS Server, da
        + Single Point of Failure
        + Verkehrslast
        + Entfernung der zentralen Datenbank
        + Wartung / Aktualisierung
        + Keine Skalierbarkeit
      * DNS-Caching
* Transportschicht
  + Aufgaben
    - Logische Kommunikation zwischen Anwendungsprozessen
    - Identifikation der zugehörigen Prozesse über Portnummer
    - Flusskontrolle, Überlaufkontrolle
  + Protokolle
    - laufen nur auf den Endgeräten
    - Internet-Transportprotokolle: TCP, UDP
    - Pipeline-Protokolle
      * Erlaub mehrere Pakete, die noch bestätigt werden müssen
      * Erhöhter Nutzungsgrad
      * Arten
        + Go-Back-N

Bis zu N nicht bestätigte Pakete

Kumulatives ACK (bestätigt alle Pakete bis einschließlich Sequenznummer)

Timer für das älteste, nicht bestätigte Paket

Paket korrekt:

Sende ACK für das empfangene Paket 🡪 Sequenznummer im ACK

Paket nicht korrekt:

Verwerfen 🡪 Kein Puffer beim Empfängers

Sende ACK mit Sequenznummer des letzten korrekten Pakets

* + - * + Selective Repeat

Individuelle Bestätigung aller korrekten Pakete

Empfänger puffert Pakete

Sender wiederholt nur Pakete ohne ACK

Sender braucht Timer für jeden unbestätigte Paket

Sendefenster: N Pakete mit aufeinanderfolgenden Sequenznummern

* + Multiplexen
    - Einsammeln der Daten von mehreren Prozessen
    - Einpacken der Daten (Segmente) mit Steuerinformationen
  + Demultiplexen
    - Abliefern der empfangenen Segmente beim richtigen Prozess
    - Verbindungsorientiert (TCP)
      * Verbindung wird eindeutig durch Quell-IP-Adresse, Quellportnummer, Ziel-IP-Adresse, Zielportnummer (Parameter) identifiziert
      * Beim Empfangen
        + Auswerten der Parameter, eigene IP anhängen
        + Bei Verbindungsanfrage an Server-Socket weiterleiten
        + Weiterleiten des Segments zum Socket mit den passenden Parametern
    - Verbindungslos (UDP)
      * Sockets eindeutig durch IP, Portnummer identifiziert
      * Beim Empfangen
        + Auswerten der Parameter
        + Weiterleitung an den Socket mit der Portnummer
  + Portnummern
    - Definiert durch die IANA (Internet Assigned Numbers Authority)
  + Transfer über zuverlässigen Kanal
    - ACK
      * Empfänger 🡪 Sender
      * Quittung für das Empfangen von Paketen ohne Fehler
    - NAK
      * Empfänger 🡪 Sender
      * Quittung für das Empfangen von Paketen mit Fehlern
      * Sender schickt Paket erneut
  + TCP \*1)
    - Sequenznummer
      * Bytes im Strom sind durchnummeriert
      * Sequenz-Nummer: Kleinste Bytenummer, die im Paket gesendet wird
      * Acknowledgement-Nummer: Nächstes Byte, das erwartet wird
    - RTT: Round Trip Time
      * Gemessene Zeit vom Versenden bis zur Bestätigung durch ACK
      * Ändert sich dynamisch, jedoch sollte RTT sich nur langsam ändern
        + Benutzung mehrerer Messungen, nicht nur letzter Wert
    - Verwendet Timeouts, um verlorene Segmente oder ACKs wiederherzustellen
      * Timeout-Wert sollte größer als die RTT sein
      * Timeout wird an Schwankungen der RTT angepasst
      * Tritt ein Timeout auf, wird das Timeout verdoppelt 🡪 Einfache Form der Überlastkontrolle / Staukontrolle
    - Fast Retransmit
      * Verlust eines Segments erhöht Ende-zu-Ende Latenz
      * Wenn Sender 3 Duplikate eines ACKs erhält, nimmt er an, dass das Sagment verloren gegangen ist
      * Segment erneut schicken, bevor der Timer abläuft
      * Deutliche Performancesteigerung
    - Flusskontrolle
      * Kontrolle, dass der Puffer zwischen Anwendung und OS nicht überläuft
      * Anpassung der Sendegeschwindigkeit des Senders an die Konsumgeschwindigkeit des Verbrauchers
      * Empfänger informiert den Sender explizit über die aktuelle Größe des freien Puffers 🡪 RcvWindow-Feld im TCP-Header
      * Viele OS passen RcvWindow dynamisch an
    - Überlastkontrolle
      * Netzwerkgesteuerte Kontrolle
        + Router liefert Feedback an Endgeräte
        + 1 Bit zeigt Stau an (SNA, DECNet, ATM, TCP-ECN)
        + Explizite Datenrate, die ein Sender noch produzieren darf
      * Ende-zu-Ende Kontrolle
        + Jeder Sender stellt seine Übertragungsrate in Abhängigkeit von der von ihm wahrgenommenen Überlast im Netz ein
    - 3-Wege-Handshake (open)
      * Client sendet Segment mit SYN-Flag = 1 und initialer Client-Seq-Nr.
      * Server empfängt SYN, belegt Puffer und antwortet mit SYN+ACK-Segment und initialer Server-Seq-Nr.
      * Client empfängt SYN+ACK, legt Puffer an und antwortet mit ACK 🡪 Dieses Segment darf bereits Daten beinhalten
    - 4-Wege-Handshake (close)
      * Client sendet TCP FIN Kontrollsegment
      * Server empfängt FIN, antwortet mit ACK, schließt die Verbindung und sendet FIN
      * Client empfängt FIN, antwortet mit ACK und sendet ACK auf weitere empfangene FINs
      * Server empfängt ACK; Verbindung ist geschlossen
* Netzwerkschicht
  + Transport der Pakete vom sendenden zum empfangenden Host
  + Aufgaben
    - Routing
    - Switching: Eingang des Routers 🡪 Richtiger Ausgang des Routers
    - Verbindungsaufbau
  + Forwarding & Routing
    - Verbindungslos: Kein Verbindungsaufbau auf der Netzwerkebene
      * Schwierige Überlastkontrolle, Flexibilität hoch
      * Datagramm-Netzwerke
      * Pakete durch Verwenden der Zieladresse geroutet
    - Verbindungorientiert
      * Einfache Überlastkontrolle, Flexibilität gering
      * Virtuelle Kanäle
      * Verbindungsaufbau vor dem Transport der Daten
      * Paket trägt ID des virtuellen Kanals
      * Ressourcen für die Verbindung müssen reserviert werden
    - Output Port Funktion (Paketqueue ist voll)
      * Drop-Tail: Ankommende Pakete werden verworfen
      * AQM: Active Queue Management
        + Paket Markieren, sodass Sender von der Überlast erfährt
      * RED: Random Early Detection Algorithm (AQM-Strategie)
        + Schwellwerte und für die Queuelänge
        + Aktuelle Queuelänge < oder >

Verwerfe Paket

* + - * + Aktuelle Queuelänge > und <

Verwerfe / Markiere Paket mit einer Wahrscheinlichkeit, die von der durchschnittlichen Queuelänge und den Schwellwerten abhängt

* + IPv4
    - Adressklassen
      * A: 1.0.0.0 bis 127.255.255.255
      * B: 128.0.0.0 bis 191.255.255.255
      * C: 192.0.0.0 bis 223.255.255.255
      * D: 224.0.0.0 bis 239.255.255.255
      * Ineffiziente Nutzung des Adressraums
        + Klasse-B-Netz belegt 65.536 Adressen, nutzt aber nur 2000
        + Oftmals C zu klein und B zu groß
    - CIDR: Classless Interdomain Routing
      * Netzwerk-Teil einer IP-Adresse beliebig
      * Aufteilung Netzwerk/Hostteil wird über Subnetzmaske angegeben
    - ICANN: Internet Corporation for Assigned Names and Numbers
      * Vergabe von IP-Adressbereichen an ISPs und große Organisationen
      * Verwaltung DNS-Top-Level-Domains und Betrieb DNS-Root Server
    - ICMP: Internet Control Message Protocol
      * Austauschen von Steuerungsinformationen auf Netzwerkebene
      * Fehlermeldungen
      * Statusmeldungen
      * In IP-Datagrammen transportiert
    - NAT: Network Address Translation
      * Private IP-Adressen im Intranet
      * Umsetzung der virtuellen IP-Adresse in eine öffentliche IP-Adresse bei Verbindung zum Internet
      * Vorteile
        + Nur eine öffentliche IP-Adresse für das gesamte Subnetz
        + Adressen im Subnetz ohne Konsequenzen änderbar
        + Hosts im Netzwerk nicht direkt adressierbar (von außen)
      * Nachteile
        + Unzulässige Vermischungen der Protokollschichten
        + Widerspricht dem End-to-End Gedanken
        + NAT ist eine Zwischenlösung auf dem Weg zu IPv6
  + IPv6
    - 128 Bit (16 Byte)
    - Header: Feste Länge vom 40 Byte
    - Header-Chaining außerhalb des festen Headers (Extension Header)
  + Tunneling
    - Übergang von IPv4 auf IPv6
    - IPv6 Paket wird als Payload in IPv4 Paket gepackt und zwischen IPv4 Routern verschickt
  + Paketfilterung
    - Firewall
      * Grenze zwischen unsicherem und sicherem Netz
      * Komponenten
        + Paketfilter: Anwendungsunabhängig

Filterung von Informationen auf Netzwerk- und Transportschicht

Zustandslos: Untersuchung unabhängig von anderen Paketen

Zustandsbasiert: Untersuchung abhängig von anderen Paketen

* + - * + Applikationsfilter: Proxy, Anwendungsspezifisch
      * DMZ: Demilitarisierte Zone, Bereich zwischen mehreren Firewalls
      * Hosts in DMZ heißen Bastion Hosts
  + Routing
    - Global: Alle Router kennen die komplette Topologie
      * Dijkstra-Algorithmus
    - Dezentral: Jeder Router kennt die physikalisch direkt verbundenen Nachbarn
      * Austausch der Informationen mit dem direkten Nachbarn
      * „Distanz-Vektor“-Algorithmus
    - Statisch: Router ändern sich langsam mit der Zeit
    - Dynamisch: Router ändern sich häufig
    - AS: Autonome Systeme, Zusammenschluss von Routern zu Regionen
      * Intra-AS Routing Protokolle (innerhalb AS)
        + RIP: Routing Information Protocol

Distanzvektor-Algorithmus

Pfadkosten: Anzahl Hops

Alle 30s Austausch von Routinginformationen via RIP Response Message

* + - * + OSPF: Open Shortest Path First

Nachfolger von RIP

Nutzt „Link State”-Algorithmus

Zustandsinformationen über periodische „Link State Broadcast“-Nachrichten verteilt

Jeder Router kennt gesamte Topologie des AS

Routenberechnung via Dijkstra

Ausgetauschte OSPF-Nachrichten werden authentifiziert und über TCP-Verbindung gesendet

Mehrere Pfade mit gleichen Kosten parallel verwendet 🡪 Lastverteilung

* + - * + IGRP

CISCO-Protokoll

RIP-Nachfolger

Distanzvektor-Protokoll

Verwendet TCP

Updateinformationen nur bei Veränderungen ausgetauscht, nicht periodisch

DUAL: Distributed Updating Algorithm

Berechnung schleifenfreier Routingpfade

* + - * Inter-AS-Routing Protokolle (zwischen AS)
* Sicherungsschicht
  + Kapseln der Datagramme in Rahmen (Frames), fügt Header und Trailer Hinzu
  + Problem: Erkennung von Rahmengrenzen im Bitstrom
  + Aufgaben
    - Fehlererkennung
      * Signalabschwächung, Störungen
      * Signalisiert Sender das Paket neu zu verschicken oder verwirft es
    - Fehlerkorrektur
      * Korrigieren von Bitfehlern ohne neue Paketanforderung
    - Flusskontrolle
      * Ausgleich der Datenrate zwischen Sender und Empfänger
      * Z.B. CRC: Cyclic Redundancy Check, UDP-Prüfsumme, …
* Netzwerksicherheit

1. TCP Segment Struktur

