Vorlesung Kommunikationssysteme
Wintersemester 2024/25

Protokolldesign und das Internet

Christoph Lindemann

Comer Buch, Kapitel 1, 2, 3

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 die Probleme komplexer Netzwerkprotokolle
- Funktionen der Schichten des TCP/IP Stacks und deren Zusammenspiel verinnerlichen

Themen:

- Einführung
- Design von Netzwerkprotokollen
- TCP/IP
- Internet
  - Kommunikation
  - Rollen

Einleitung

Definition

- Netze verbinden
  - **Computer verschiedenster Hersteller und Architekturen**
- sowohl
  - **in der Nähe als auch über große Distanzen**
- mittels
  - **Netzequipment und Netzwerkprotokollen unterschiedlicher Hersteller**
- und ermöglichen **beliebige Anwendungen**

Aufbruchsstimmung

- Seit den 1970ern entwickelten sich viele Netze zunächst inkompatibel zueinander und entsprachen nicht dem gerade genannten Ideal
- Abgeschottete Netze (sowohl Hardware als Software) von
  - Telekommunikationsunternehmen
  - Computerherstellern
  - Netzwerkausrüstern
- **Keinerlei Interoperabilität**

Forschungsprojekte

- Innerhalb der Forschungscommunity wurden diese abgeschotteten Netze abgelehnt
- Forschungsprojekte entstanden in den 1980ern :
  - *Ethernet* am Xerox Palo Alto Research Center
  - *Token passing ring networks* am MIT
  - *ARPANET* am Department of Defense

Grundlagen des Wachstums

- **Offene Netze sind für ein Wachstum eines Netzes auf Welt-umspannende Größe notwendig**

- **Offene Netze:**
  - Mehrere Gruppen arbeiten zusammen an einer neuen Technologie
  - Es wird eine Spezifikation geschaffen; diese ist für jeden frei zugänglich
  - Produkte orientieren sich an dieser Spezifikation

- Im Gegensatz zu geschlossenen Netzen:
  - Jeder Hersteller entwickelt seine private Technologie

Wichtige Gremien

- IEEE (*Institute of Electrical and Electronics Engineers*)
- IETF (*Internet Engineering Task Force*)
- ITU (*International Telecommunications Union*)
- ISO (*International Organization for Standardization*)
- W3C (*World Wide Web Consortium*)
- …

Problem: Komplexität (1)

- Als Einsteiger wirken Rechnernetze zunächst einschüchternd und komplex
- Es existieren viele Spezifikationen / Standards / Technologien
- Die Technologien entwickeln sich laufend weiter
- Viele Technologien können miteinander verbunden und genutzt werden
- Keine alles umfassenden Theorien / Konzepte, die erklären wie Netze gebaut und verwaltet werden sollen

Problem: Komplexität (2)

- Zum vollständigen Verständnis muss ein umfangreiches Wissen auf mindestens fünf Gebieten gesammelt werden
  - **Netzwerkanwendungen und –programmierung**
    - Entwicklung von robusten, fehlerfreien und korrekten Netzwerkanwendugen
  - **Signalverarbeitung und Kommunikation**
    - Das physikalische Medium, dessen Eigenschaften und Übertragungsverfahren

Problem: Komplexität (3)

- Zum vollständigen Verständnis muss ein umfangreiches Wissen auf mindestens fünf Gebieten gesammelt werden
  - **Packet Switching und Netztechnologien**
    - Feste Leitung wie im Telefonnetz werden durch Paketvermittlung ersetzt → geteiltes Netz
  - **TCP/IP**
    - Es gibt keine One-Fits-All Packet Switching Technologie
    - TCP/IP bietet das Konzept des globalen Internets und überlässt anderen Protokollen (z.B. Ethernet) den physikalischen Transport oder auch das Routing (z.B. OSPF, RIP, BGP)

Problem: Komplexität (4)

- Zum vollständigen Verständnis muss ein umfangreiches Wissen auf mindestens fünf Gebieten gesammelt werden
  - **Allgemeine Netzwerkkonzepte und -technologien**
    - Streaming von Sprache und Bild
    - Netzwerk-Monitoring
    - Software Defined Networking
    - …

Private Netze (1)

- Das Internet ist ein **öffentliches Netz** auf Basis von TCP/IP
- Zugang zu öffentlichen Netzen wird von Internet Service Providern (ISP) gekauft
- Netz bzw. Netzzugang ist ein Service
- **Private Netze** werden von einer einzige Gruppe genutzt
  - Können auch Netze sein, die von ISPs gemietet wurden

Private Netze (2)

- Klassen von privaten Netzen
  - **Privatkunden**: Heimnetz mit wenigen Computern
  - **Small Office / Home Office (SOHO)**: Firmennetz mit Computern, Druckern, Bezahlterminals, …
  - **Small-to-Medium Business (SMB)**: SOHO + zweiten Standort oder Produktionsstätte
  - **Large Enterprise**: Mehrere Standorte, viele Router

Interoperabilität (1)

- Basis der Kommunikation innerhalb eines Netzes ist das **Netzwerk-Protokoll**
  - **Enthält Regeln, die alle Aspekte der Kommunikation festlegen**
    - **Z.B. elektrische Spannung, Bedeutung von Zeichenfolgen**
- **Ein funktionierendes Netzwerk greift auf viele Protokolle zurück, die unterschiedliche Aspekte der Kommunikation regeln**
  - **Z.B. physikalische Übertragung vs. Transportprotokoll (TCP)**

Interoperabilität (2)

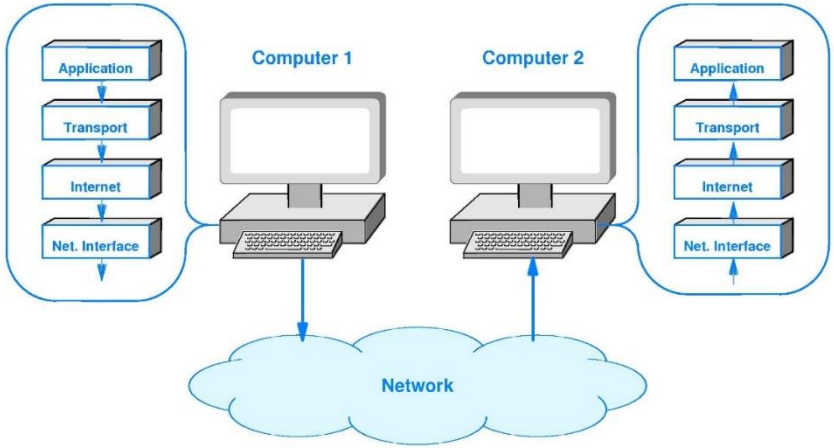
- Protokolle beschreiben
  - **Syntax**
    - Format und Darstellung von Nachrichten
    - Kodierung
  - **Semantik**
    - Bedeutung von Nachrichten
    - Verfahren um Nachrichten auszutauschen
    - Fehlerbehandlung

Protokolldesign (1)

- Ziele des Protokolldesigns:
  - **Effizienz**
  - **Vollständigkeit** (alle Szenarien abgedeckt)
  - **Vermeidung von Redundanz** → Jedes Protokoll löst ein anderes Problem der Netzwerkkommunikation
  - **Interoperabilität** → Schnittstellen zwischen den Protokollen
- Lösung: Protokolle werden zusammen als **Protokoll-Familie** entworfen (z.B. TCP/IP Protokollstack)

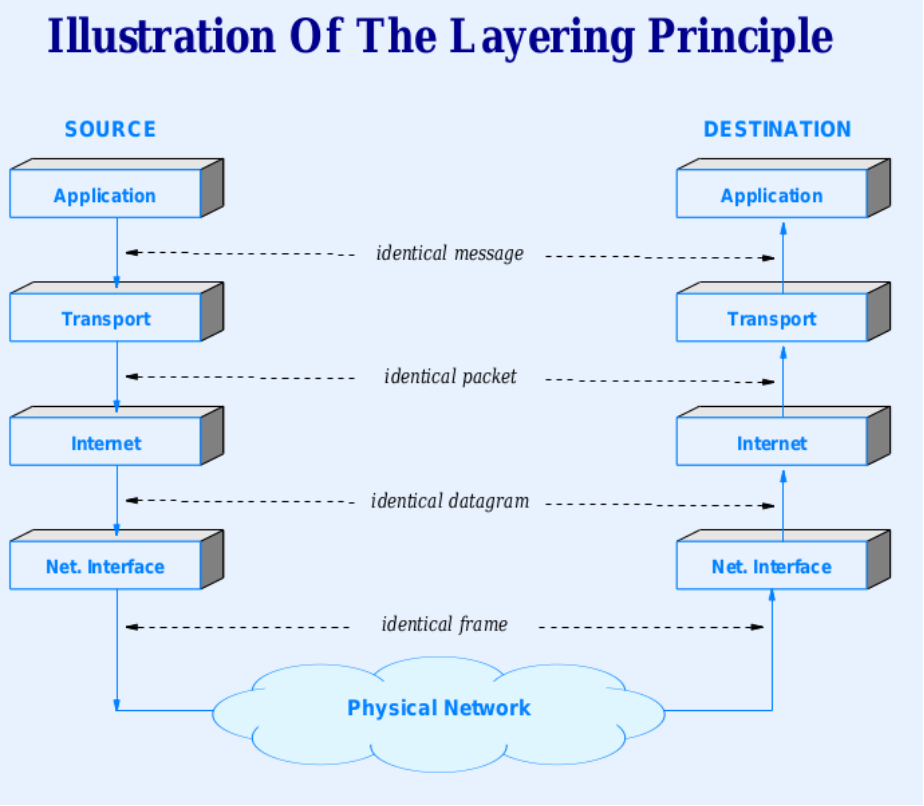
Protokolldesign (2)

- Die Protokoll-Familie teilt das Problem der Kommunikation in Schichten auf → **Schichtmodell** (Layer)
- Für jede Schicht kann es mehrere Protokolle geben
- Für die praktische Netzwerkkommunikation durchlaufen zu transportierende Daten **linear jede Schicht** und werden dort von **genau einem Protokoll** verarbeitet
- **Der Output einer höheren Schicht dient als Input der darunter liegenden (**beim Senden; beim Empfang umgekehrt)



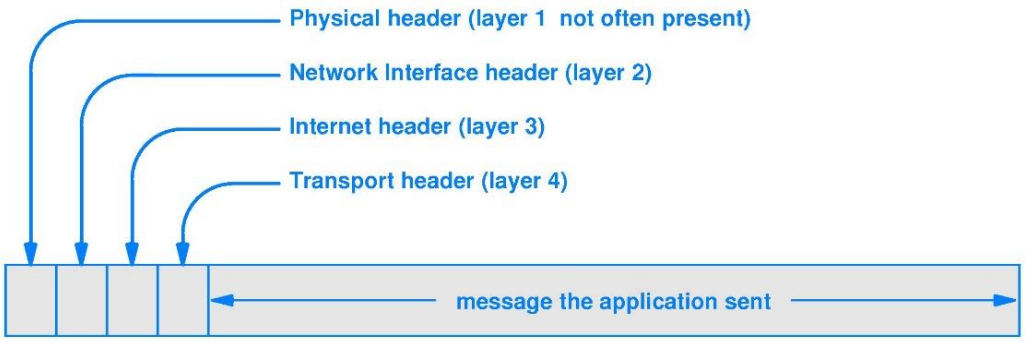
Protokolldesign (3)

- Pakete die beim Sender von Schicht A erzeugt werden, werden beim Empfänger von der gleichen Schicht A gelesen



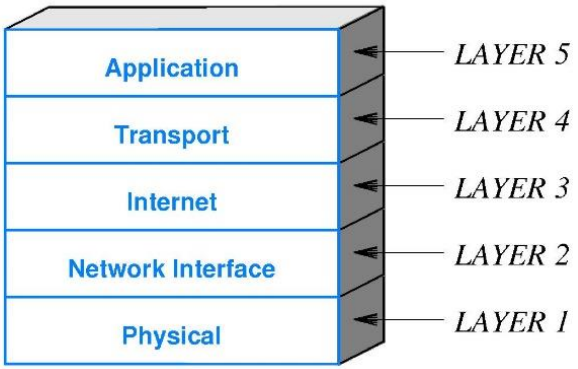
Protokoll Header

- Jedes Protokoll jeder Schicht ergänzt beim Senden **Metadaten** zu den zu transportierenden Daten
  - Prüfsummen
  - Adressen (Ports, IP-Adressen, MAC-Adressen)
  - Kontrollinformationen (Paketnummern, Segmentierung)
- **Header mit diesen Daten wird in jeder Schicht den Daten voran gestellt**
- Beim Empfänger extrahiert jedes Protokoll seinen entsprechenden Header und wertet Daten aus



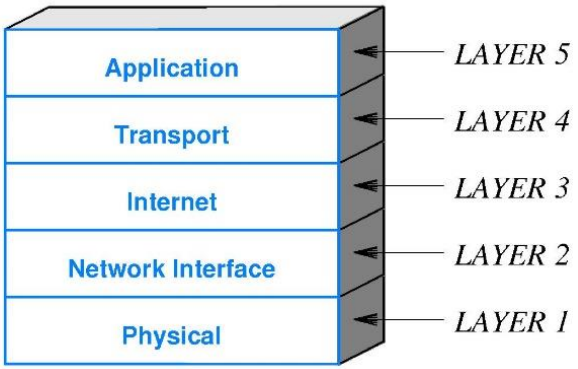
TCP/IP Stack (1)

- **Physical Layer**
  - Übertragung der Daten als Signal auf dem physikalischen Medium
- **Network Layer (Medium Access Control, MAC)**
  - Steuert die Kommunikation auf dem Medium zwischen zwei physikalisch verbundenen Geräten
  - Hardwareadressierung, Paketgrößen, Medienzugriff
  - Z.B. Ethernet



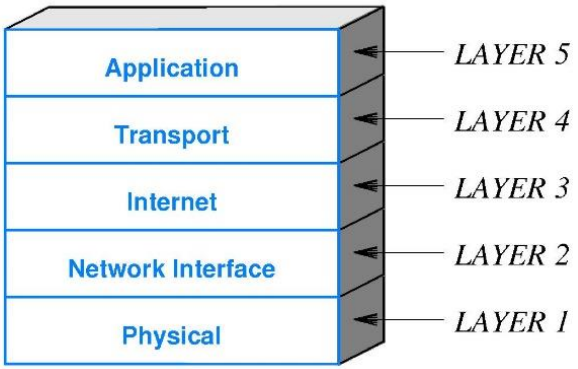
TCP/IP Stack (2)

- **Internet Layer:**
  - Kommunikation zwischen zwei beliebigen Geräten irgendwo im Netz (nicht zwangsweise physikalisch verbunden)
  - Adressierung, Paket-Struktur, Segmentierung, …
  - Z.B. IP
- **Transport Layer:**
  - Kommunikation zwischen Anwendungen auf beliebigen Geräten
  - Überlastkontrolle, Flusskontrolle, Fehlerkorrektur, …



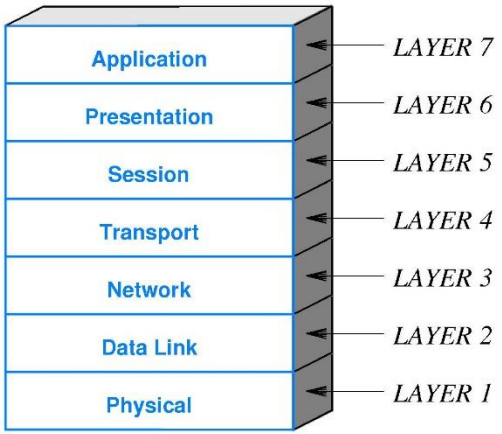
TCP/IP Stack (3)

- **Application Layer:**
  - Protokolle auf Anwendungsschicht
  - Struktur der zu übertragenden Daten
  - Jedes Programm das Daten übertragen möchte implementiert zwangsweise ein eigenes Protokoll
  - Z.B. FTP, Bittorrent, IMAP, …



OSI Schichtenmodell

- Von der ISO und der ITU gemeinsam entworfen
- Konnte sich nicht gegen TCP/IP durchsetzen
- Heute keine Relevanz



Grundlegende Konzepte des Internet

„Das Internet“ (1)

- **Häufiges Missverständnis: Internet == WWW**
  - Völlig Falsch → WWW wird durch Web-Server und Web-Browser implementiert
  - Eigene Protokolle und Standards (z.B. HTTP, HTML)
  - Kommuniziert wird über „das Internet“
- **„Das Internet“ ist ein generischer Kommunikations-Service**
- Keines der bekannten Anwendungen (z.B. WWW, E-Mail) ist Teil des Internets → Reine Applikationen, die den Service „Internet“ nutzen

„Das Internet“ (2)

- **„Das Internet“ bezeichnet im Allgemeinen die Protokolle TCP, IP und UDP** → gemeinsam als TCP/IP Protokoll-Stack bekannt
  - Implementierungen des Internet und Transport Layers im 5 Schichten Modell
- Es existieren APIs mit denen Anwendungen direkt die Dienste des TCP/IP Stack nutzen können → Viele Aspekte der Fehlerkorrektur, Flusskontrolle, Reihenfolge der Daten, … sind vor der Anwendungen vorborgen
  - **Bedeutendste API: Sockets**

Kommunikationsklassen

- TCP/IP unterstützt zwei Arten der Kommunikation:
  - Stream Transport
    - Implementiert von **TCP**
    - Verbindungsorientierte Übertragung von Byte-Strömen
  - Message Transport
    - Implementiert von **UDP**
    - Verbindungsloser Austausch von Nachrichten

Stream Transport (1)

- „Streaming“ von Daten einer Anwendung auf Host A zu einer anderen Anwendung auf Host B
  - **TCP versteht nur einfache Byte Streams** → Interpretation der Daten nur auf Anwendungsschicht
  - **IP** bildet ein **Packet Switched Network** → Auch der TCP Stream wird in einzelne Pakete zerlegt
    - Größe der Pakete abhängig vom Netz, Netzauslastung und Verarbeitungsgeschwindigkeit der Kommunikationspartner
    - Am Ziel so wieder zusammengestellt, dass der Ziel-Anwendung der ursprüngliche Stream geliefert werden kann
- Beispiele: Netflix-Stream, Datei-Download

Stream Transport (2)

- **Verbindungsorientiert**
  - Bevor zwei Anwendungen miteinander kommunizieren können müssen sie eine Verbindung aufbauen
    - Vergleichbar mit einem Anruf über das Festnetz
  - **Verbindung ist bi-direktional** → Daten können in beide Richtung fließen → **zwei Streams**
  - **Nachdem alle Daten ausgetauscht wurden wird die Verbindung wieder abgebaut**

Stream Transport (3)

- Zusammenfassung:
  - **Verbindungsorientiert**
  - **1-zu-1 Kommunikation**
  - **Byte-Stream basiert**
  - **Jede Verbindung erzeugt zwei Streams → Sender zu Empfänger und Empfänger zu Sender**
  - **Byte Streams können eine beliebige Länge besitzen**
- Das am häufigsten verwendete Verfahren

Message Transport (1)

- Bietet Anwendungen einen Dienst zur **Übertragung von Nachrichten**
- Nachrichten werden stets so zugestellt, wie sie abgeschickt wurden
  - Mehrere Nachrichten werden nicht aus Effizienzgründen zusammen aggregiert
  - Nachrichten werden nicht aufgeteilt
- **Keine Garantien zur Zustellung der Daten → Out-of-Order Zustellung, Verlust und Mehrfachzustellung möglich**
- Ermöglicht Broadcast und Multicast → 1:N sowie M:N Kommunikation

Message Transport (2)

- **UDP**
- **Verbindungslos**
- **Max. 64 KB Nachrichten**
- Hauptsächlich für Multimedia eingesetzt → **geringe Latenz** bei VoIP
- Weniger als 5% des gesamten Internet-Traffic

Rollen im Internet (1)

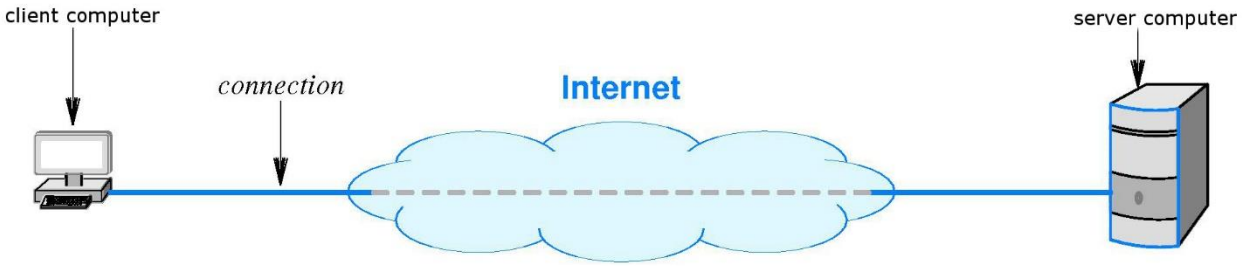
- Bisher gelernt: TCP ist verbindungsorientiert → Probleme
  - **Welcher der beiden Kommunikationspartner baut die Verbindung auf?**
  - **Woher kennen sich die beiden Kommunikationspartner überhaupt?**
- **Lösung: Feste Rollenaufteilung zwischen Server und Client**
  - Server-Anwendung: bietet einen Dienst an und wartet auf Verbindungen
  - Client-Anwendung: greift auf entfernte Dienste zu und baut die Verbindung auf

Rollen im Internet (2)

Server Anwendung	Client Anwendung
Startet zuerst	Start als zweites
Kennt die potentiellen Clients nicht	Muss den Server kennen
Wartet passiv auf Verbindungen von Clients	Baut aktiv die Verbindung zum Server auf
Empfängt Daten vom und sendet Daten zum Client	Sendet Daten zum und empfängt Daten vom Server
Bearbeitet beliebig viele Clients nacheinander und teils parallel	Beendet sich möglicherweise nach erfolgreicher Kommunikation

Rollen im Internet (3)

- Begriff „Server“ wird häufig auch für Hardware verwendet
  - Server sind Computer, die sich mit ihrer starken Hardware (schnelle CPUs, viele Kerne, großer Arbeitsspeicher, schnelle Netzanbindung, …) von Desktop-Rechnern abheben und auf denen Server-Anwendungen laufen



Rollen im Internet (4)

- Client/Server Anwendungen sehen häufig zwei Arten von Nachrichten vor
  - **Requests**
    - Vom Client zum Server geschickt um einen bestimmten Dienst anzufordern
  - **Responses**
    - Vom Server zum Client als Antwort auf den Request
- Pipelined Requests und Responses sind möglich
- Beispiel WWW:
  - Browser verbindet sich zum HTTP-Server und fragt eine HTML-Seite an → Web-Server antwortet

Rollen im Internet (5)

- Computer können gleichzeitig mehrere Client- als auch Server-Anwendungen ausführen
  - E-Mail
  - Chat
  - Videostreaming
- Anwendungen kontaktieren möglicherweise jeweils einen anderen Server
- Clients können mehrfach geöffnet sein (z.B. mehrere Browserfenster)
- Mehrere Server-Anwendungen auf der gleichen Hardware erhöht die Auslastung und senkt Kosten

Rollen im Internet (6)

- Anwendungen implementieren oft sowohl Server- als auch Client-Funktionalität
  - PHP-Server-Anwendung fragt eine MariaDB-Datenbank ab → Client für den SQL-Server
- Zirkuläre Abhängigkeiten müssen vermieden werden
- Komplizierte Performance-Evaluierung

Server Adressierung (1)

- Clients müssen sich aktiv zu einer Server-Anwendung verbinden

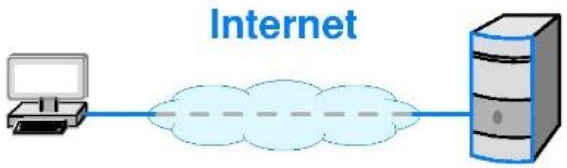
- Identifikation der Server-Anwendung mittels

  - **IP-Adresse**: Adresse des Computers, auf dem die Server-Anwendung läuft
    - Über das Domain Name System (DNS) werden IP-Adressen sprechende Bezeichner zugewiesen, z.B. www.uni-leipzig.de

  - **TCP/UDP – Port:** Auswahl der Server-Anwendung auf dem Ziel-Computer
    - 16-Bit Zahl
    - Standard-Ports für bestimmte Dienste
      - HTTP: Port 80
      - SSH: Port 23

Server Adressierung (2)

- Start after server is already running
- Obtain server name from user
- Use DNS to translate name to IP address
- Specify the port that the service uses, N
- Contact server and interact



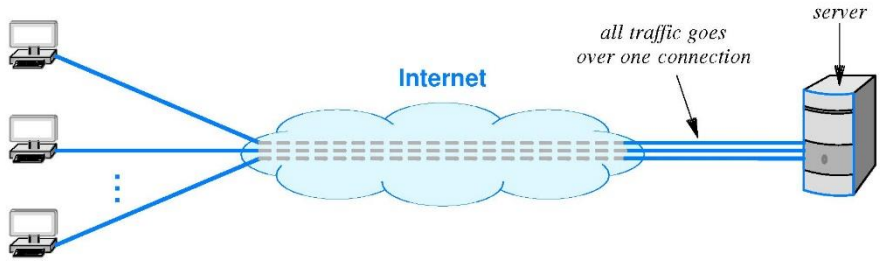
- Start before any of the clients
- Register port N with the local system
- Wait for contact from a client
- Interact with client until client finishes
- Wait for contact from the next client

Parallele Verarbeitung

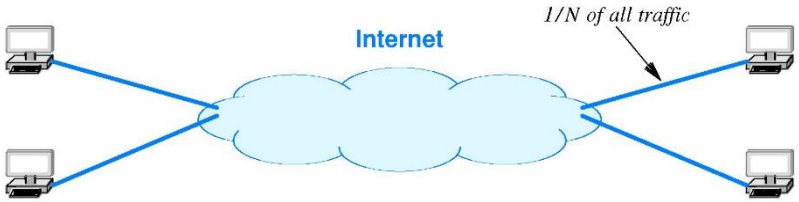
- Server bearbeiten mehrere Verbindungen typischerweise parallel → **mehrere Threads → Clients behindern sich nicht gegenseitig**
  - Ein **Haupt-Thread** zur Annahme von neuen Verbindungen
  - Ein **Handler-Thread** pro akzeptierter Verbindung → Bearbeitet die Daten / Requests des Clients
- Handler-Threads werden einem Thread-Pool entnommen
- Eine Warteschlange für eingehende Verbindungen→ ist der Thread-Pool leer blockiert der Haupt-Thread und nimmt zunächst keine neuen Anfragen mehr an

P2P Kommunikation

- Einzelner Server als Flaschenhals und Single Point of Failure



- Peer-To-Peer (P2P) – Anwendungen trennen nicht mehr zwischen Client und Server
- Jeder Anwendungsinstanz bietet auch Dienste an
  - Z.B. File Sharing mittels BitTorrent



Zusammenfassung (1)

- Für das Internet werden offene Standards und Protokolle benötigt
- Protokolle werden zu Familien zusammengefasst und interagieren in einem Schichtmodell
- TCP/IP ist die Standard Internet-Protokollfamilie
- OSI-Schichtenmodell mit 7 Schichten hat sich nicht durchgesetzt

Zusammenfassung (2)

- „Das Internet“ bezeichnet den TCP/IP Protokoll-Stack
- TCP ist ein Streaming-Protokoll
- UDP ist ein Nachrichten-Protokoll
- Anwendungen nutzen TCP/IP über APIs
- Netzwerk-Anwendungen bestehen aus einem Client und einem Server