

Rechnerkommunikation

Felix Leidl

2. August 2023

Inhaltsverzeichnis

Anwendungsschicht	3
Paradigmen	3
Client-Server	3
Wechselnde Rollen	3
Verteilte Anwendung	3
Peer-to-Peer	3
Anforderungen	3
Hypertext Transfere Protocol (HTTP)	4
Ablauf	4
Format der Anfragen	4
Anfragenachricht	4
Format der Antworten	5
Antwortnachricht	5
HTTP-Ablauf	6
Antwortzeit	6
Dynamische Inhalte	6
Caching	7
HTTP/2	7
File Transfere Protocol (FTP)	8
Simple Mail Transfer Protocol (SMTP)	8
Vertraulichkeit und Datenintegrität	9
Netzwerkmanagement	10
Simple Network Management Protocol (SNMP)	10
Management Information Base (MIB)	12
Domain Name Service (DNS)	13
Domain-Struktur	13
Hierarchie von Name-Servern	13
Ressource Records	13
DNS-Protokoll	14
Anfragen	14
Transportschicht	15

Netzwerkschicht	15
Sicherungsschicht	15
Physikalische Schicht	15

Anwendungsschicht

Netzwerkanwendung:

- Anwendungsprozesse auf verschiedenen Hosts
- kann direkt unter Verwendung der Dienste der Transportschicht implementiert werden
- standardisiert Anwendung benutzen ein Anwendungsprotokoll, das das Format der Nachrichten und das Verhalten beim Empfang festlegt

Paradigmen

Client-Server

Server stellt Dienst zur Verfügung, der vom Client angefragt wird

Wechselnde Rollen

Hosts übernehmen mal die eine, mal die andere Rolle

Verteilte Anwendung

Besteht aus mehreren unabhängigen Anwendungen, die zusammen wie eine einzelne Anwendung erscheinen (z.B. WebShop mit Web-Server, Applikations-Server und Datenbank), Koordination ist zwar verteilt, findet aber für das Gesamtsystem statt

Peer-to-Peer

Vernetzung von Gleichen:

- dezentrale Architektur (z.B. Bitcoin)
- Hybridarchitektur: Initialisierung findet über zentrale Architektur statt, Anwendung dezentral zwischen Hosts

Anforderungen

- Verlust
- Bitrate
- Verzögerungszeit

Hypertext Transfere Protocol (HTTP)

Ablauf

1. Benutzer gibt URI (Uniform Resource Identifier) in Web-Browser ein
2. URI enthält Host-Namen eines Web-Servers und den Pfad zu einem Objekt
3. Web-Browser stellt Anfrage an Web-Server für dieses Objekt
4. Web-Server liefert Objekt an Web-Browser zurück
5. Web-Browser stellt Objekt für Nutzer lesbar da

Format der Anfragen

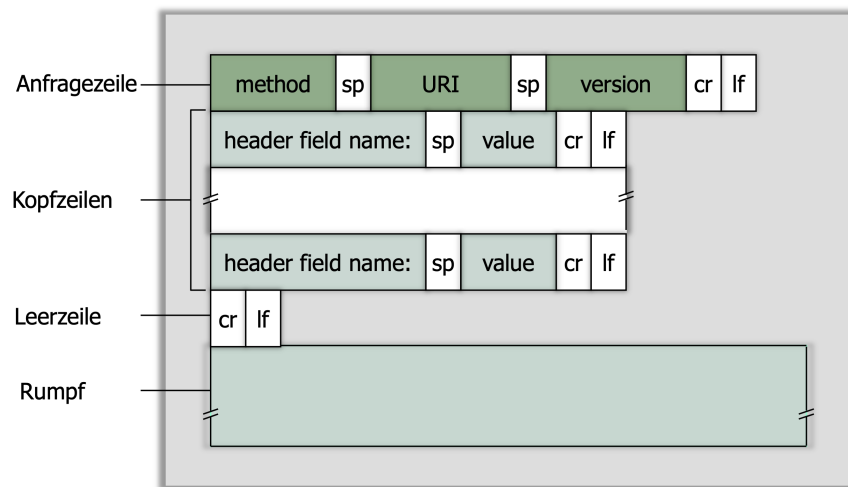


Abbildung 1: Anfrage

Anfragenachricht

- Methoden:
 - GET: Abruf eines Dokuments, besteht aus Methode, URI, Version
 - HEAD: Abruf von Metainformationen eines Dokuments
 - POST: Übergabe von Informationen an Server
 - PUT
 - DELET
- Kopfzeilen:
 - Typ/Wert-Paare, Typen: Host, User-agent, ...
- Rumpf:

- leer bei GET, kann bei POST Inhalt haben

GET:	/somedir/page.html HTTP/1.1
HOST:	www.someschool.edu
User-agent:	Mozilla/4.0
Connection:	close
Accept-language:	de-de

Abbildung 2: Anfragenachricht

Format der Antworten

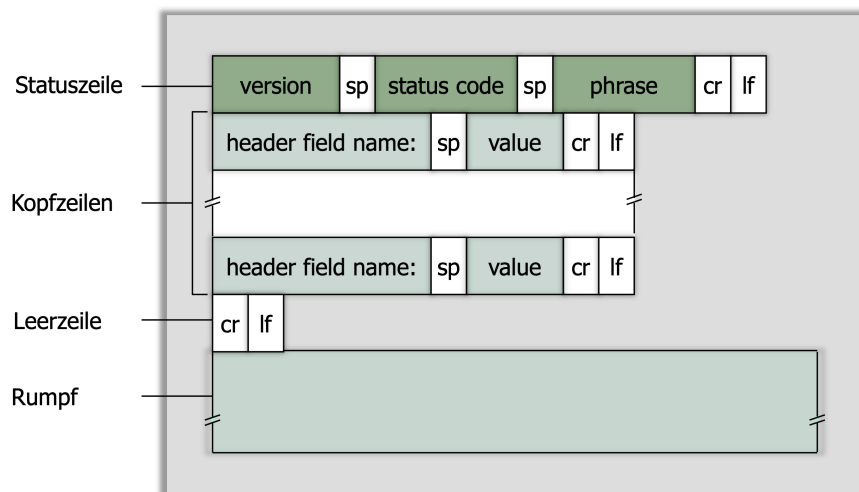


Abbildung 3: HTTP Antworten

Antwortnachricht

HTTP/1.1	200 OK
Connection:	close
Date:	Thu, 06 Aug 1998 12:00:15 GMT
Server:	Apache/1.3.0 (Unix)
Last-Modified:	Mon, 22 Jun 1998 ...
Content-Length:	6821
Content-Type:	text/html
data	data

Abbildung 4: Antwortnachricht

HTTP-Ablauf

Nicht-persistentes HTTP:

Für jedes Objekt wird eine einzelne TCP-Verbindung aufgebaut. Entweder Basisseite und eingebettete Objekte sequentiell oder parallele Verbindung für eingebettete Objekte

Persistentes HTTP:

Server lässt Verbindung bestehen, alle Objekte werden über eine TCP Verbindung gesendet. Ohne Pipelining wird jedes Objekt einzeln Angefragt, mit alle auf einmal

Antwortzeit

Basisseite: Aufbau der TCP-Verbindung (1x RTT) + Anfrage hin und Antwort zurück (1x RTT) \Rightarrow 2RTT + Zeit zum Senden + weitere Wartezeiten durch TCP

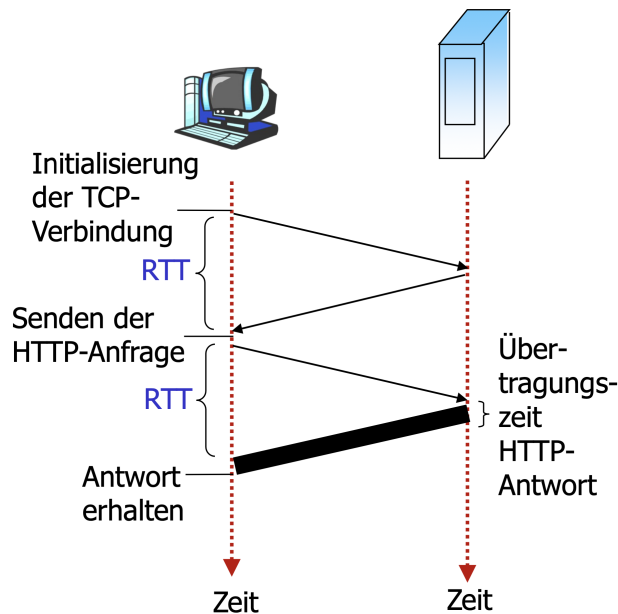


Abbildung 5: Antwortzeit

Dynamische Inhalte

Common Gate Interface (CGI) verarbeitet als externer Prozess die Information und liefert neue HTML-Seite an Server

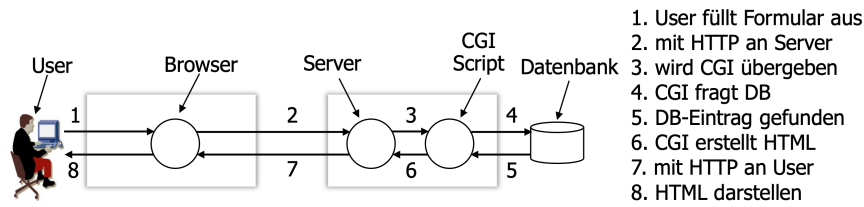


Abbildung 6: CGI

Scripting: Durch Interpretation von eingebetteten Skripten können dynamische Inhalte erzeugt werden.

Serverseitig: im HTML ist Code eingebettet, der vom Server interpretiert wird und dabei HTML erzeugt, z.B. PHP

Clientseitig: im HTML ist Code eingebettet, der vom Client interpretiert wird, z.B. JavaScript

Caching

Cache (Proxy Server) ist Server für Web-Browser und Client für Web-Server, der als Zwischenspeicher zur Verringerung der Wartezeit des Nutzers und des Netzverkehrs dient

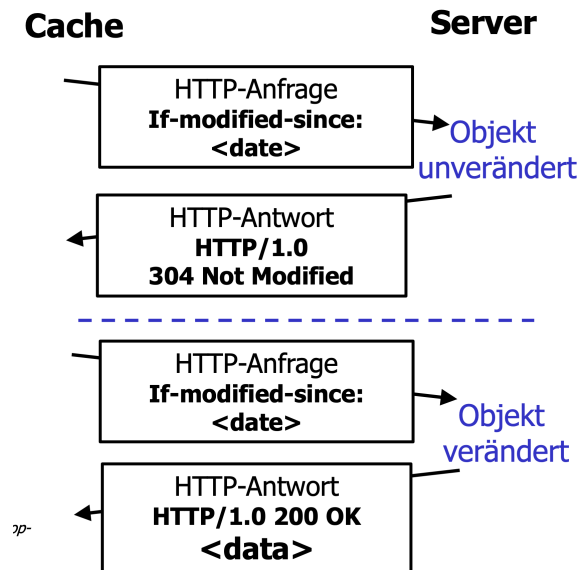


Abbildung 7: Cache Server

HTTP/2

Wesentliche Elemente:

- gleiche Methoden
- binäres statt textbasiertes Format
- Multiplexing verschiedener Ströme über eine TCP-Verbindung, Vermeidung von Head-of-Line (HOL) Blockierung durch große Objekte durch Aufteilung in kleinere Frames und Interleaving
- Header-Kompression
- Server-Push

File Transfere Protocol (FTP)

- Übertragung zwischen zwei Hosts
- eine TCP-Verbindung (Port 21) zur Steuerung
- lesbare Kommandos: USER username, PASS password, LIST, RETR filename, STOR filename, ...
- jeweils eine TCP-Verbindung (Port 20) zur Übertragung einer Datei
- 'out-of-band-controll'

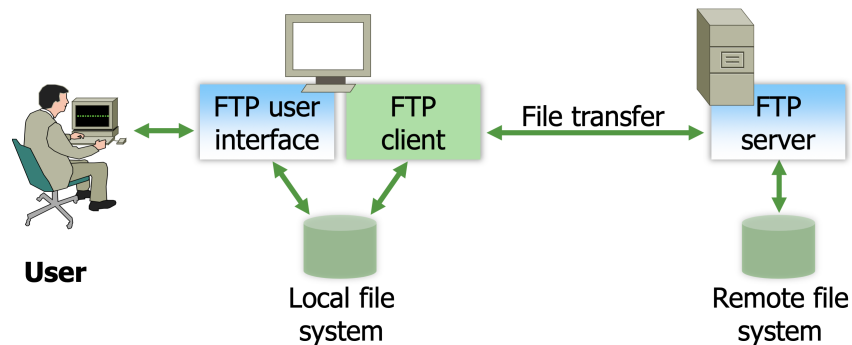


Abbildung 8: FTP

Simple Mail Transfer Protocol (SMTP)

- Nachrichten im ASCII-Format, Kopf, Rumpf
- andere Daten werden in ASCII umgewandelt angehängt
- Versenden mit SMTP über TCP (lesbar)
- Abholen mit POP3, IMAP, HTTP (lesbar)

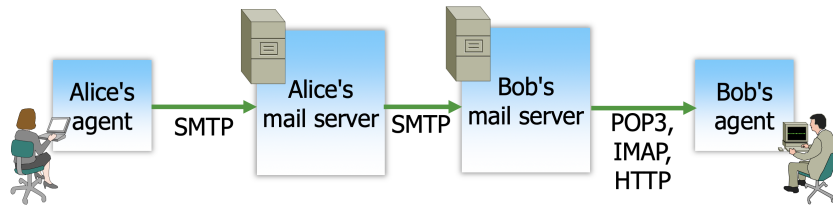


Abbildung 9: SMTP

- nutzt TCP (Port 25)
- direkte Übertragung: vom Sendenden zu empfangendem Server
- drei Phasen der Übertragung:
 - Handshake
 - Nachrichtenübertragung
 - Abschlussphase
- Interaktion mittels Befehlen und Antworten
 - Befehle: ASCII-text
 - Antworten: Statuscode und Text
- Nachrichten müssen 7-bit ASCII-text sein

Vertraulichkeit und Datenintegrität

1. Erzeugung eines Hashwerts der E-Mail
2. Signierung mit privatem Schlüssel K_A^- von Alice
3. Verschlüsselung der Mail und der Signatur mit K_S
4. Asymmetrische Verschlüsselung von K_S mit dem öffentlichen Schlüssel K_B^+ von Bob

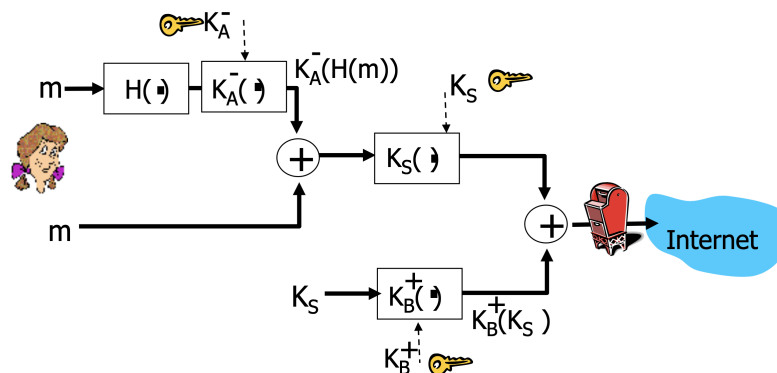


Abbildung 10: Vertraulichkeit und Datenintegrität

Netzwerkmanagement

Überwachung und Verwaltung eines Netzes = komplexes HW/SW Gebilde
FCAPS-Modell nach ISO:

- Fault Management: Monitoring, Dokumentation, Reaktionsmaßnahmen
- Configuration Management: Übersicht über Geräte und deren HW/SW-Konfigurationen
- Accounting Management: Verwendung von Netzwerkressourcen, Festlegung, Kontrolle, Dokumentation des Zugangs von Benutzern und Geräten
- Performance Management: Monitoring von Auslastung, Durchsatz, Antwortzeiten, Dokumentation, Reaktionsmaßnahmen
- Security Management: Monitoring und Kontrolle des Zugangs, Schlüsselverwaltung, Firewalls, Intrusion Detection

Simple Network Management Protocol (SNMP)

Organisationsmodell:

- Managing Entity: Prozess auf zentraler Management Station, Client
- Managed Device
- Managed Object: HW oder SW im Managed Device
- Managed Agent: Prozess auf Managed Device, kann lokal ausgeführt werden
- Anfrage/Antwort-Protokoll zwischen Manager und Agent über UDP
- Manager basierend auf Regeln, was zu tun ist

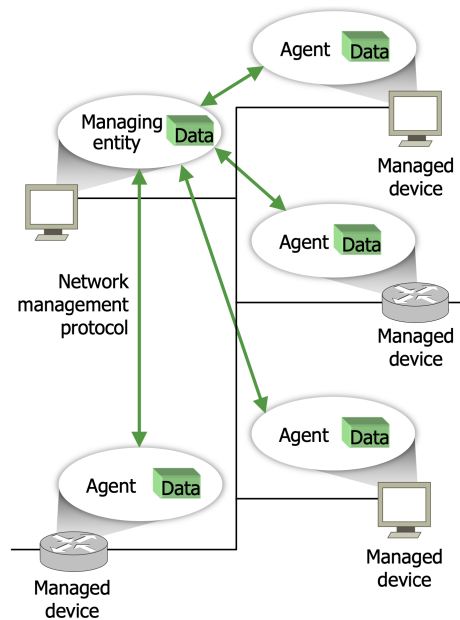


Abbildung 11: SNMP

SNMP Messages:

- **Get:** Manager an Agent, um Daten von Agent zu erhalten
- **Get-Next:** Manager an Agent, für nächsten Datensatz, Zugriff auf sequentielle Datensätze
- **Get-Bulck:** Manager an Agent, für mehrere Datensätze auf einmal
- **Set:** Manager an Agent, initialisiert oder ändert den Wert eines Datensatzes
- **Response:** Agent an Manager, Antwort auf Get- und Set-Nachrichten
- **Trap:** Agent an Manager, unaufgeforderte Nachricht über Fehlersituation

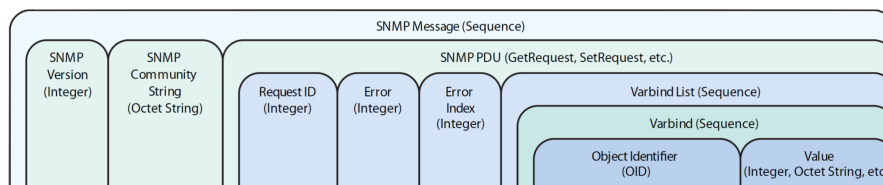


Abbildung 12: Format von SNMP Nachrichten

Management Information Base (MIB)

MIB-Module enthalten Datenstrukturen für die Managed Devices. Das genaue binäre Format der Übertragung wird mit Basic Encoding Rules (BER) festgelegt. Jedes MIB-Modul wird eindeutig durch die Objekt-Identifizierung (OID) der ASN.1 bezeichnet.

```

UDP-MIB DEFINITIONS ::= BEGIN
...
udp      OBJECT IDENTIFIER ::= { mib-2 7 }
...
udpInDatagrams OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total number of UDP datagrams delivered to UDP users."
    ::= { udp 1 }
...
udpGroup OBJECT-GROUP
    OBJECTS { udpInDatagrams, udpNoPorts,
              udpInErrors, udpOutDatagrams,
              udpLocalAddress, udpLocalPort }
    STATUS      current
    DESCRIPTION
        "The udp group of objects providing for management of UDP
         entities."
    ::= { udpMIBGroups 1 }

END

```

Abbildung 13: MIB-Modul

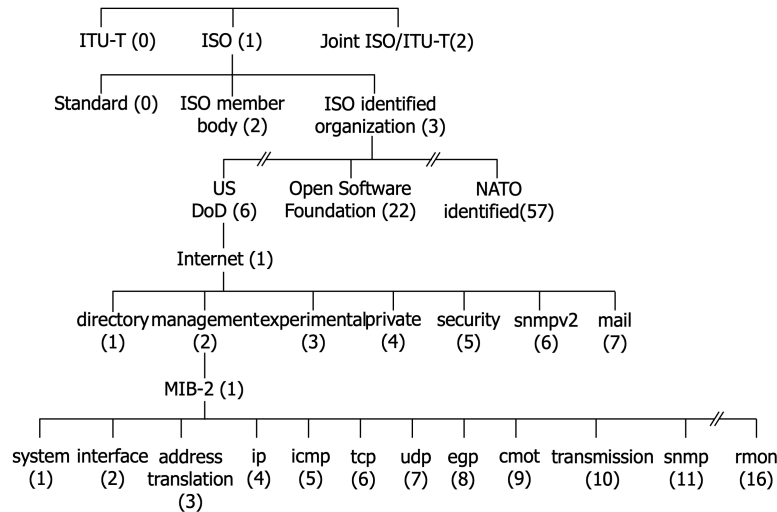


Abbildung 14: OID

Domain Name Service (DNS)

DNS bildet Domain-Namen auf Werte ab und ist als verteilte Datenbank, bestehend aus vielen Name-Servern, repräsentiert

Domain-Struktur

DNS implementiert hierarchischen Namensraum für Internet-Objekte

Hierarchie von Name-Servern

- **Root Name Server:** weltweit
- **Top-level Domain-Server** für com, org, net, ...
- ...
- **autoritativer Name-Server:** unterste Ebene für einzelne Organisationen

Ressource Records

- **TTL:** Time to Live, Dauer der Gültigkeit
- **Type = A:**
 - Wert = IPv4-Adresse
 - AAAA-Einträge für IPv6-Adressen
- **Type = NS:**
 - Wert = Domainname eines Hosts, auf dem ein Server läuft, der Namen in der Domain auflösen kann
- **Type = CNAME:** (Canonical Name)
 - Wert = kanonischer Name eines Hosts, ermöglicht Aliasnamen
- **Type = MX:** (Mail Exchange)
 - Wert = Domain-Name des Hosts, auf dem der Mail-Server läuft

DNS-Protokoll

Identification	Flags
Number of questions	Number of answers RRs
Number of authority RRs	Number of additional RRs
Questions (variable number of questions)	
Answers (variable number of resource records)	
Authority (variable number of resource records)	
Additional information (variable number of resource records)	

Abbildung 15: DNS Protokoll

Anfragen

iterativ:

- Antwort: anderer Server, der Name evtl. auflösen kann (oder keine Antwort)
- NS- und A-Datensatz
- Antwort wird sofort geliefert, es muss keine Information gespeichert werden, gut für hochfrequente Server

rekursiv:

- Antwort: Auflösung des Namens, die u.A. von anderen Servern geholt wird
- A-Datensatz
- bei Anfragen an einen anderen Server muss die Information gespeichert werden

Transportschicht

Netzwerkschicht

Sicherungsschicht

Physikalische Schicht