KURSBUCH



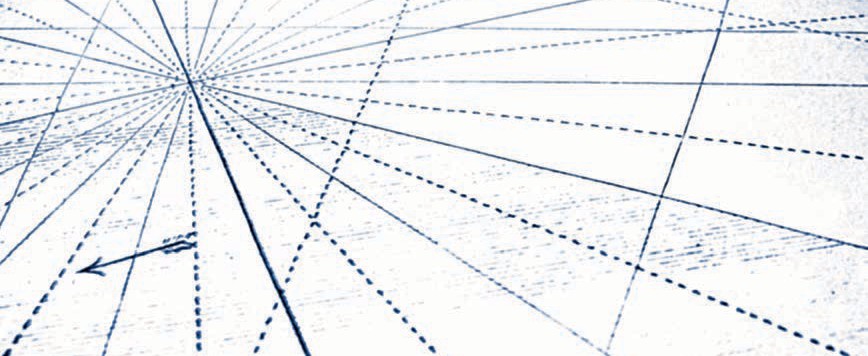
## Netzwerke und verteilte Systeme

DLMCSNDS01



Lernziele

##### Einführung 9



Computernetzwerke werden basierend auf ihrer Größe und Architektur sowie auf den bereitgestellten Diensten in verschiedene Kategorien untergliedert. Allen gemeinsam ist jedoch, dass es sich im Wesentlichen um eine Reihe von Computinggeräten handelt, die durch die Übermittlung von digitalen Daten miteinander kommunizieren. Diese Daten werden dabei als digitale Signale übermittelt, welche über verschiedene Arten von physikalischen Medien weitergegeben werden. Im Lehrbuch „Netzwerke und verteilte Systeme“ werden die unterschiedlichen Arten von Computernetzwerken, die Konzepte der digitalen Übertragung, die Grundlagen der Kommunikationstechnik sowie verschiedene Arten physikalischer Schichten erläutert.

Bei der Datenübertragung zwischen sendenden und empfangenden Computern werden die Daten in einer bestimmten Abfolge übermittelt. Dabei werden verschiedene Schichten durchlaufen, die jeweils eigene Funktionalitäten und Dienste bereitstellen. Diese sind durch Schichtprotokolle standardisiert. Nach einer Einführung in die einzelnen Netzwerkschichten werden die Dienste und Protokolle dieser Schichten erläutert.

Mit mehr als fünf Milliarden Geräten weltweit ist das Internet die größte Art eines Computernetzwerks. In diesem Kursbuch werden das für das Internet verwendete TCP/IP-Referenzmodell (Transmission Control Protocol/Internet Protocol) und der Protokollstapel beschrieben. Da proportional zur kontinuierlich steigenden Anzahl an Diensten und Nutzern auch die Sicherheitsrisiken und Schwachstellen im Internet steigen, werden auch die für das Internet relevanten Sicherheitsaspekte dargelegt.

Eine weitere Lektion ist den unterschiedlichen Architekturen gewidmet, die die Grundlage von Computernetzwerken bilden. Dabei werden die Algorithmen und Protokolle für verschiedene Typen verteilter Architekturen beschrieben. Abschließend wird auf eine Reihe innovativer Technologien eingegangen, die bei Computernetzwerken eine immer größere Rolle spielen. Dazu gehören u. a. das Internet der Dinge (Internet of Things, IoT), Pervasive Computing, Mobile Computing, Ubiquitous Computing sowie Distributed-Ledger-Technologien. Auch diese aktuellen Themen werden ausgeführt.

[www.iubh.de](http://www.iubh.de/)



# Lektion 1

## Computernetzwerke

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… die Konzepte von Computernetzwerken sowie der verschiedenen Arten von Computernetzwerken, Netzwerktopologien und Verbindungen zu erläutern

… die Konzepte der digitalen Datenübertragung, der Kommunikationstechnik und der Codierungstheorie zu beschreiben

… die Konzepte von Medien der physikalischen Schicht sowie von Datenübertragungsmethoden anzuwenden

… die Netzwerkleistung zu bestimmen, zu analysieren und zu bewerten

DL-E-DLMCSNDS01-U01

1. Computernetzwerke

### Einführung

In unserer modernen Gesellschaft hängen immer mehr Aspekte unseres Alltags von Computernetzwerken wie dem Internet ab. In den verschiedensten Bereichen — vom Bildungs- und Gesundheitswesen über Sicherheit, tägliche Transaktionen und Bankwesen bis hin zu Unterhaltung und Kommunikation — bilden diese Netzwerke die Grundlage für alltägliche Aufgaben und Abläufe. Dabei werden verschiedene Arten von Netzwerken implementiert, die sich im Hinblick auf die Architektur, die Reichweite, den Geltungsbereich und die bereitgestellten Dienste unterscheiden. Unabhängig davon, um welche Art von Netzwerk es sich handelt und welche Dienste bereitgestellt werden, wird ein Computernetzwerk in erster Linie eingesetzt, um Daten zwischen Geräten zu übertragen. Dieser als Datenübertragung oder Datenkommunikation bezeichnete Vorgang umfasst eine Reihe von Schritten: das Konvertieren von analogen Informationen in digitale Daten, die Datencodierung, die Komprimierung, die Verschlüsselung, die Konvertierung von Daten in Signale, die Modulation, die Signalübertragung und die Datenweitergabe. Die Dienstqualität (Quality of Service, QoS) eines Computernetzwerks hängt von der Leistung der Datenkommunikation ab, die sich anhand von Verzögerung, Durchsatz und Bitfehlerrate ermitteln lässt.

### Grundlagen der digitalen Datenübertragung

Ein Computernetzwerk ist eine Sammlung von Computinggeräten, die miteinander kommunizieren. Für diese Kommunikation werden digitale Daten über verschiedene zwischengeschaltete Verbindungsgeräte sowie über kabelgebundene oder kabellose Medien zwischen Sender und Empfänger übertragen. Ein Netzwerkgerät wird üblicherweise als Knoten und das Medium als Verbindung bezeichnet (Forouzan, 2013). Die Kommunikation zwischen den Knoten umfasst verschiedene Schritte, Regeln oder Mechanismen. Diese Mechanismen werden durch Standardisierungsorganisationen definiert und als Protokolle bezeichnet. Die digitale Datenübertragung umfasst also die folgenden fünf Hauptelemente (Forouzan, 2013):

1. die „Nachricht“, also die Daten oder Informationen, die zwischen den Geräten übermittelt werden sollen.
2. den „Sender“, also das Gerät, das die Nachricht übermittelt.
3. den „Empfänger“, also das Gerät, das die Nachricht empfängt.
4. die „Verbindung“, also das physikalische Medium, über das die Nachricht zwischen Sender und Empfänger übertragen wird.
5. „Protokolle“, also standardisierte Mechanismen, die für die Kommunikation eingesetzt werden.

###### Die Schritte der digitalen Datenübertragung

Ein Computer ist ein digitales Gerät, das Daten als Sammlung binärer Zahlen speichert. Zwischen zwei kommunizierenden Knoten werden also im Grunde eine Reihe von Nullen und Einsen ausgetauscht. Dabei werden die Daten in erster Linie als elektrische oder elektromagnetische Wellen übertragen, die als Signal bezeichnet werden. Aus diesem Grund sollte ein Mechanismus vorhanden sein, um die digitalen Daten als elektrische oder elektromagnetische Wellen (das digitale Signal) zu codieren. Dieser Mechanismus wird als Leitungscodierung oder digitale Modulation im Basisband bezeichnet. Abhängig von der Art des physikalischen Mediums werden jedoch verschiedene Knoten benötigt, um das Signal mit unterschiedlichen Frequenzen zu übertragen. Zu diesem Zweck wird das digitale Signal unter Verwendung eines analogen Signals

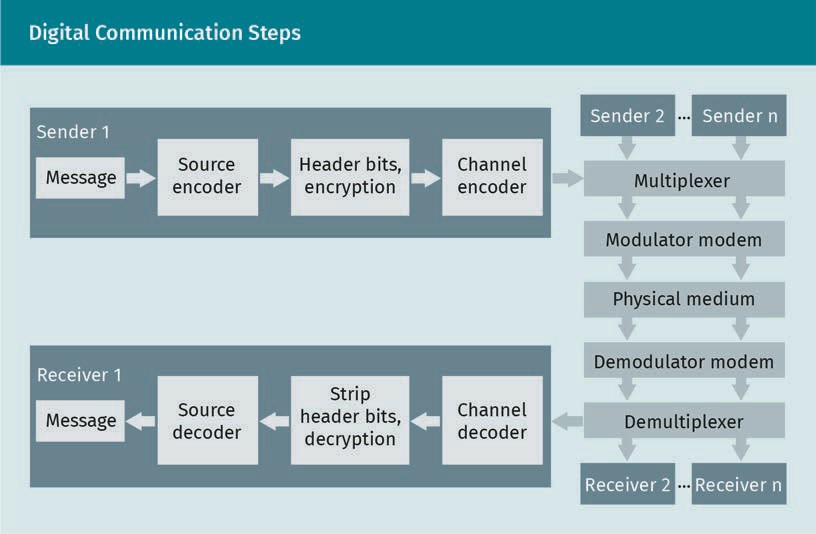
Computernetzwerke

mit der gewünschten Frequenz moduliert. Dieses Signal wird als Träger oder Carrier bezeichnet. Das für die Modulation und Demodulation eingesetzte Gerät ist das sogenannte Modem. Der Begriff „Modem“ setzt sich aus den Begriffen „Modulator“ und „Demodulator“ zusammen und bezieht sich beispielsweise auf ein DSL-Modem (Digital Subscriber Line) oder ein Kabelmodem (Tanenbaum & Wetherall, 2014). DSL-Modems werden eingesetzt, um Computer oder Router mit Zugriff auf Telefonleitungen mit dem Breitbandinternet zu verbinden. Kabelmodems hingegen nutzen Kabelfernsehnetze, um einen Breitbandinternetzugriff für Netzwerkknoten bereitzustellen.

In einem Computersystem müssen Nachrichten oder Informationen digital dargestellt werden. Die Konvertierung der analogen Informationen in digitale Daten erfolgt über die sogenannte Quellencodierung (Comer, 2015). Wenn ein Signal über das physikalische Medium weitergegeben wird, kommt es aufgrund von Pfadverlust, Fading oder Übertragungsdämpfung zu einer Verringerung der Signalstärke (Dämpfung). Das Signal kann auch aufgrund von Rauschen, Interferenzen, Mehrwegeeffekten oder Abschattungen (Shadowing) verzerrt werden. Durch Dämpfung und Verzerrung kann ein Wert „0“ in einen Wert „1“ bzw. ein Wert „1“ in den Wert „0“ umgekehrt werden. Dies wird als Bitfehler bezeichnet (Kurose & Keith, 2017). Bitfehler werden mithilfe der Kanalcodierung ermittelt und korrigiert (Comer, 2015). Um die übermittelten Daten gegen mögliche Angriffe zu schützen, wird auf den Kommunikationsknoten ein Mechanismus zur Verschlüsselung und Entschlüsselung implementiert.

Der sendende Knoten benötigt für die Übermittlung der Daten eine gewisse Zeit. Diese Zeitdauer wird als Übertragungsverzögerung bezeichnet. Sie ist proportional zur Größe der Daten (Kurose & Keith, 2017). Der Senderknoten verwendet die sogenannte Quellencodierung, um die Daten vor der Konvertierung in ein Signal zu komprimieren. (Comer, 2015). Nach der Datenkomprimierung muss der Senderknoten einige „zusätzliche Bits“ zu den eigentlichen Datenbits hinzufügen. Diese zusätzlichen Bits heißen „Headerbits“ und enthalten die Adressen der an der Kommunikation beteiligten Knoten. Die eigentliche Nachricht besteht aus den Datenbits.

Da eine Netzwerkverbindung über mehrere Kanäle verfügen kann, kann ein physikalisches Medium die Signale mehrerer Senderknoten gleichzeitig übertragen. Der Mechanismus zum Kombinieren der Signale aus mehreren Quellen wird als Multiplexing bezeichnet. Beim Demultiplexing wird ein kombiniertes Signal erneut in mehrere Signale entbündelt (Comer, 2015; Kurose & Keith, 2017).



###### Leistung der digitalen Kommunikation

Die Gesamtleistung eines digitalen Kommunikationssystems oder Computernetzwerks wird als Dienstqualität (Quality of Service, QoS) bezeichnet. Die Dienstqualität hängt von einer Reihe von Faktoren wie der Übertragungsrate, der Verzögerung, der Kanalkapazität, dem Durchsatz und der Bitfehlerrate (Bit Error Rate, BER) sowie weiteren Aspekten ab, die nachfolgend beschrieben sind:

* Die „Übertragungsrate“ ist die Anzahl an Bits pro Sekunde, die ein Sender über das physikalische Medium überträgt bzw. weitergibt.
* Die „Verzögerung“ ist die erforderliche Dauer, um eine Nachricht vom Sender an den Empfänger zu übermitteln.
* Die „Übertragungsverzögerung“ ist die Zeit, die ein Sender benötigt, um ein Bit zu übermitteln bzw. weiterzugeben.
* Die „Kanalkapazität“ ist die maximale Anzahl an Bits, die pro Sekunde über eine Verbindung übertragen werden können. In der Regel entspricht die Übertragungsrate des Senders der Kanalkapazität.
* Die „Auslastung der Verbindung“ ist das Verhältnis der aktuell übertragenen Bits zur Kanalkapazität. Angenommen, die Verbindungskapazität beträgt 2 Megabit pro Sekunde (MBit/s) und der Sender überträgt mit einer Geschwindigkeit von 1 MBit/s. In diesem Fall beträgt die Verbindungsauslastung 50 Prozent. Wenn die Übertragungsrate 2 MBit/s beträgt, dann ist die Verbindung zu 100 Prozent ausgelastet.
* Der „Durchsatz“ ist die Anzahl an Bits, die pro Sekunde erfolgreich vom Empfänger bzw. von den Empfängern empfangen werden. Wenn die Kanalkapazität z. B. 2 MBit/s und die Übertragungsrate 1 MBit/s beträgt, dann ist der Durchsatz 1 MBit/s. Wenn die Übertragungsrate auf

Computernetzwerke

2 MBit/s erhöht wird, dann ist auch der Durchsatz 2 MBit/s. Wenn die Übertragungsrate nun weiter auf 3 MBit/s erhöht wird, ist der Durchsatz weiterhin 2 MBit/s, da die maximale Kapazität des Kanals bei 2 MBit/s liegt.

* Die „Bitfehlerrate“ (Bit Error Rate, BER) ist das Verhältnis von Bitfehlern zur Gesamtzahl an übertragenen Bits. Wenn insgesamt 10.000 Bits übertragen wurden und die Anzahl an Bitfehlern 10 beträgt, so liegt der BER-Wert bei 0,001 bzw. 10-3.

### Netzwerktopologien und Verbindungen

Vor dem Bau eines Hauses müssen Architekten und Ingenieure die Gebäudekonstruktion sorgfältig planen. Genauso müssen Netzwerktechniker das Netzwerklayout präzise definieren, bevor die Verbindungen und Knoten (Geräte) eines Netzwerks bereitgestellt werden. Und dieses Layout wird als Netzwerktopologie bezeichnet. Dabei wird zwischen einer physikalischen und einer logischen Topologie unterschieden. Eine physikalische Topologie ist die physikalische Abbildung des Netzwerks, in der die dargestellten Positionen der Netzwerkknoten in Übereinstimmung mit den physikalischen Werten und Eigenschaften der Netzwerkeinheiten geplant werden. Im Gegensatz dazu ist ein logisches Netzwerk eine logische Abbildung des Netzwerks, die zeigt, wie die Netzwerkknoten miteinander verbunden sind. Die eigentlichen Positionen der Knoten werden dabei nicht berücksichtigt. Während eine physikalische Topologie für den Entwurf, die Bereitstellung und die Analyse eines bestimmten Netzwerks wichtig ist, ist eine logische Topologie für die Untersuchung und Analyse eines allgemeinen Netzwerks von Bedeutung.

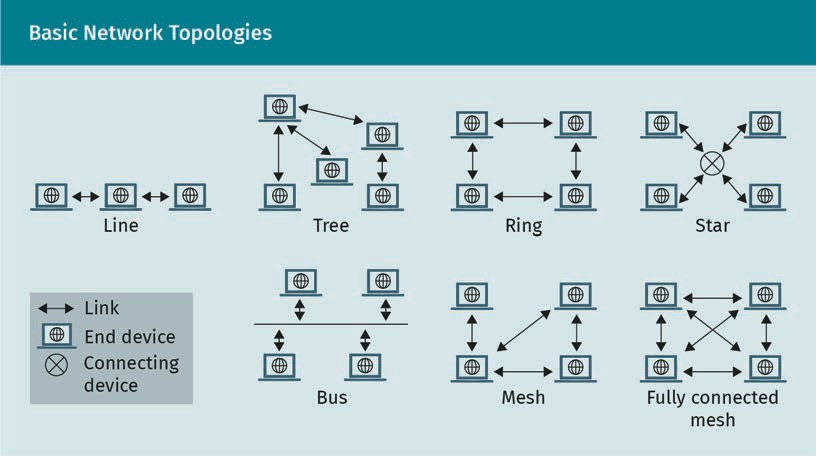
###### Netzwerktopologien

Die einfachste Art von Netzwerktopologie umfasst lediglich einen Sender und einen Empfänger. Wenn die Anzahl an Netzwerkknoten steigt, wird diese Topologie komplex. Abhängig von der zugrunde liegenden Topologie kann selbst bei Netzwerken mit identischen Ressourcen die Dienstqualität unterschiedlich ausfallen. Im Folgenden sind einige der gängigsten Netzwerktopologien aufgeführt:

* In einer „Bustopologie“ sind alle Endknoten an eine gemeinsame Leitung bzw. einen gemeinsamen Bus angeschlossen, die bzw. der als Backbone bezeichnet wird.
* Die Knoten einer „Linientopologie“ sind hintereinander geschaltet miteinander verbunden. Es gibt zwei Endknoten, die jeweils nur über eine Verbindung verfügen. Alle anderen Knoten in einer Linientopologie verfügen über zwei Verbindungen.
* Eine „Ringtopologie“ ähnelt einer Linientopologie, allerdings sind in diesem Fall keine Endknoten vorhanden.
* Bei einer „Sterntopologie“ sind alle Endknoten mit einem zentralen Verbindungspunkt verbunden.
* In einer „Mesh-Topologie“ bzw. einem vermaschten Netzwerk sind die Knoten willkürlich miteinander verbunden. Ist jeder Knoten mit jedem anderen Knoten in diesem Netzwerk verbunden, wird dies als vollständig vermaschtes Netzwerk bezeichnet.
* Eine „Baumtopologie“ ist eine Kombination aus einer Bus- und einer Sterntopologie. Folglich werden diese Netzwerke auch als Stern-Bus-Topologie bezeichnet.
* Bei einer „Hybridtopologie“ verschmelzen mehrere Topologien zu einem Netzwerk. Diese Art von Topologie ist die gängigste Topologie großer Netzwerke.

Topologie

Eine Netzwerktopologie ist die logische oder physikalische Abbildung der Verbindungen zwischen Netzwerkknoten (Geräten).



Jede dieser Topologien bietet Vor- und Nachteile. Welche Topologie sich für ein bestimmtes Netzwerk eignet, hängt vom Zweck des Netzwerks, den QoS-Anforderungen und der Ressourcenverfügbarkeit ab.

Bustopologie

Die Bustopologie weist die einfachste Architektur auf. Es wird ein einziger Bus bzw. ein einziges Kabel benötigt, um sämtliche Endknoten zu verbinden. Da zudem keine Verbindungsgeräte oder -knoten erforderlich sind, ist die Installation kostengünstig. Darüber hinaus ist die Installation zusätzlicher Geräte recht simpel. Und beim Ausfall eines Knotens wird die Verbindung zwischen anderen Knoten nicht beeinträchtigt. Da das Backbone jedoch von allen Knoten gemeinsam verwendet wird, bricht das gesamte Netzwerk beim Ausfall des Backbones zusammen. Zudem kann die Leistung der Knoten beeinträchtigt werden, wenn weitere Knoten hinzugefügt werden.

Linientopologie

In einem Netzwerk mit Linientopologie aus n Knoten werden n − 1 Verbindungen benötigt. Im Gegensatz zum Busnetzwerk können Daten über mehrere Repeater übertragen werden, da die Knoten über andere Knoten verbunden sind. Beim Ausfall eines Knotens wird das Netzwerk in zwei Liniennetzwerke unterteilt, die nicht miteinander verbunden sind. Fällt ein Endknoten aus, wird die Verbindung zwischen anderen Knoten jedoch nicht beeinträchtigt. Die Installation neuer Geräte ist in einer Linientopologie nicht ganz so einfach wie bei der Bustopologie.

Ringtopologie

Jeder Knoten einer Ringtopologie verfügt über zwei Verbindungen. Wenn die Endknoten einer Linientopologie miteinander verbunden werden, entsteht eine Ringtopologie. Gleichermaßen entsteht durch den Ausfall eines Knotens in einer Ringtopologie eine Linientopologie. Ein Ringnetzwerk verfügt über eine identische Anzahl von Knoten und Verbindungen. Ähnlich wie bei der Linientopologie können Daten über mehrere Knoten übertragen werden. Bei einer Ringtopologie können die Daten bidirektional übertragen werden. Tritt bei einem Knoten ein Problem auf, können die anderen Knoten weiterhin verbunden sein.

Computernetzwerke

Sterntopologie

Jedes Paar aus Endgeräten ist über zwei Verbindungen miteinander verbunden. Die erste Verbindung verbindet den Sender mit dem Verbindungspunkt, die zweite den Verbindungspunkt mit dem Empfänger. Beim Ausfall des zentralen Verbindungspunkts wird die Verbindung aller anderen Knoten unterbrochen, sodass das gesamte Netzwerk ausfällt. Im Vergleich zu anderen Netzwerktopologien lässt sich ein Sternnetzwerk recht einfach erweitern. Die Knoten zweier separater Sternnetzwerke lassen sich problemlos miteinander verbinden, indem die zentralen Verbindungspunkte der beiden Topologien miteinander verbunden werden. Für ein Sternnetzwerk mit n Endknoten sind n Verbindungen erforderlich.

Baumtopologie

Eine Baumtopologie setzt sich aus Knoten und Verbindungen zusammen, die in diesem Fall auch als Kanten bezeichnet werden. Für ein Baumnetzwerk mit n Knoten sind n − 1 Verbindungen erforderlich. Vergleichbar mit Stern- und Bustopologien lässt sich auch ein Netzwerk mit Baumtopologie recht problemlos erweitern. Der wesentliche Nachteil einer Baumtopologie ist, dass beim Ausfall eines übergeordneten Knotens mehrere getrennte Netzwerke entstehen können.

Mesh-Topologie

Bei dieser Topologie sind die Knoten durch mehrere Pfade miteinander verbunden. Beim Ausfall eines Knotens wird die Verbindung zwischen anderen Knoten also nicht unterbrochen. Bei einem vollständig vermaschten Netzwerk mit n Knoten werden n n − 1 /2 Verbindungen benötigt. Ein vollständig vermaschtes Netzwerk bietet die höchste Fehlertoleranz. Die Installationskosten können bei einer großen Anzahl von Knoten (n) jedoch sehr hoch sein.

###### Verbindungen

Abhängig davon, um welchen Typ es sich handelt, kann der Umfang eines Computernetzwerks stark variieren. Die Anzahl an Knoten kann dabei von einigen wenigen Geräten bis hin zu einer Milliarde Geräte reichen. Bei Netzwerkknoten unterscheidet man zwischen Endknoten und Knoten, die als Verbindungspunkte zwischen diesen Endknoten positioniert sind. Anwendungsprogramme werden auf Endknoten ausgeführt (z. B. auf Laptops, Handys und Servern). Wenn zwei Endknoten miteinander kommunizieren, kommunizieren also die beiden Anwendungsprogramme miteinander, die auf den beiden Endknoten ausgeführt werden. Auf Verbindungspunkten können keine Anwendungsprogramme ausgeführt werden. Diese Knoten werden eingesetzt, um die Endknoten innerhalb eines Netzwerks zu verbinden. Außerdem können sie Geräte aus verschiedenen Netzwerken miteinander verbinden.

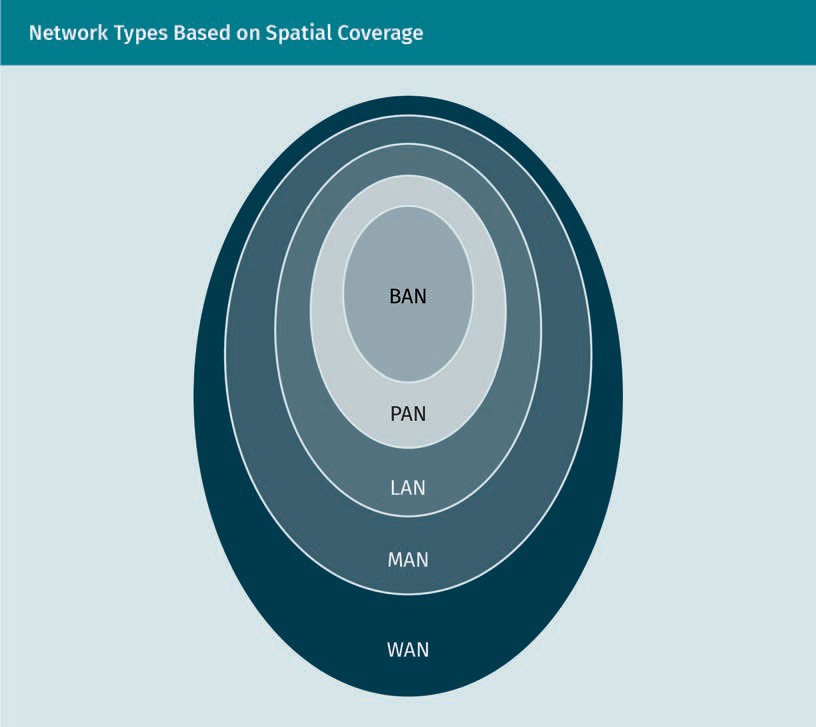
Netzwerktypen

Netzwerke werden basierend auf ihrer räumlichen Ausdehnung oder Reichweite in verschiedene Kategorien untergliedert. Im Folgenden sind die wichtigsten Kategorien aufgeführt:

* BAN-Knoten (Body Area Network) befinden sich am Körper einer Person.
* Die Reichweite eines Personal Area Network (PAN) ist üblicherweise gering. Diese Art von Netzwerk deckt z. B. den Arbeitsplatz einer Person ab (üblicherweise einen einzigen Raum).
* Ein LAN (Local Area Network) ist im Allgemeinen ein Heim- oder Büronetzwerk.
* Das Metropolitan Area Network (MAN) erstreckt sich auf eine Stadt oder ein Stadtgebiet.
* Mit seiner globalen Abdeckung bietet ein WAN (Wide Area Network) die größte Reichweite. Das bekannteste Beispiel für ein WAN ist das Internet.

Basierend auf dem Diensttyp lassen sich Netzwerke in die folgenden Kategorien untergliedern:

* + Ein Intranet ist ein privates Netzwerk innerhalb einer Organisation.
  + Das Extranet ist eine Erweiterung des Intranets, das es einer begrenzten Anzahl von autorisierten externen Nutzern ermöglicht, auf das private Netzwerk einer Organisation zuzugreifen. Bei diesen externen Nutzern handelt es sich meist um Partner, Lieferanten oder Dienstleister, mit denen die Organisation zusammenarbeitet.
  + Das Internet ist eine globale Plattform aus miteinander verbundenen Netzwerken. Da die Knoten unterschiedlicher Netzwerke über das Internet miteinander kommunizieren können, entsteht ein Netzwerk aus unzähligen Netzwerken.



Verbindungsgerät Das Verbindungsgerät (z. B. ein Router) zwischen zwei Netzwerken wird als Gateway bezeichnet (Kurose &

Keith, 2017).

Verbindungsgeräte

Verbindungsgeräte verfügen über mehrere Eingangs- und Ausgangsports. Wenn das Gerät ein Paket empfängt, wählt es den passenden Ausgangsport, um die Daten an den gewünschten Empfänger zu übermitteln. Aus diesem Grund werden Verbindungsgeräte auch als paketvermittelnde Switches bezeichnet. Es gibt eine Vielzahl verschiedener Verbindungsgeräte. Die gängigsten Geräte sind Repeater, Bridges, Hubs, Switches und Router. Repeater werden eingesetzt, um das eingehende Signal zu verstärken. Bridges sind Repeater mit einem Filter für eingehende Pakete. Mit einem Hub lassen sich mehrere Endknoten an ein gemeinsames Netzwerk anschließen. Vergleichbar mit Hubs werden auch Switches zum Verbinden mehrerer Geräte eingesetzt. Switches bieten die Möglichkeit, eingehende Pakete basierend auf der MAC-Adresse (Media Access Control) zu filtern (Kurose & Keith, 2017).

Computernetzwerke

###### Verzögerungen

Bei der Übertragung von Datenpaketen über Verbindungsgeräte können vier Arten von Verzögerungen auftreten (Newpaltz, ohne Jahresangabe). Um die gewünschte Dienstqualität sicherzustellen, muss die Verzögerung bei der Festlegung der Netzwerkparameter präzise bestimmt werden.

Übertragungsverzögerung

Die Übertragungsverzögerung ist die Zeitdauer, die ein Knoten zum Übermitteln eines Datenpakets benötigt. Bei einer Paketlänge L und einer Datenübertragungsrate R lässt sich diese Art von Verzögerung mit der folgenden Formel berechnen:

Tt’ = L

R

Die Übertragungsverzögerung für eine n-Hop-Verbindung ist nL/R.

Ausbreitungsverzögerung

Nachdem die Übermittlung eines Pakets ausgelöst wurde, wird es durch das physikalische Medium an den Empfänger weitergegeben. Die Weitergabezeit wird als Ausbreitungsverzögerung bezeichnet und ist proportional zur Entfernung zwischen Sender und Empfänger. Unter Verwendung der Entfernung ( und der Ausbreitungsgeschwindigkeit ) lässt sich die Ausbreitungsverzögerung wie folgt ausdrücken:

Tp’ = (

)

n-Hop

Eine n-Hop-Verbindung verfügt über *(n-1)* Verbindungsgeräte und *n* Verbindungen.

Bei einem drahtlosen Medium entspricht die Geschwindigkeit der Lichtgeschwindigkeit.

Verarbeitungsverzögerung

Sobald das Paket am Eingangsport eines Verbindungsgeräts empfangen wird, wird es verarbeitet, um über den geeigneten Ausgangsport an das Ziel weitergeleitet zu werden. Diese Verarbeitungszeit wird als Verarbeitungsverzögerung bezeichnet. Sie hängt von der Verarbeitungsgeschwindigkeit und Paketgröße ab. Unter Verwendung der Verarbeitungsgeschwindigkeit + und der Paketgröße L lässt sich die Verarbeitungsverzögerung mit der folgenden Formel berechnen:

Tp,’ = L

+

Warteschlangenverzögerung

Wenn ein Paket am Verbindungsgerät empfangen wird, muss es an der Pufferwarteschlange warten, bis alle vorherigen Pakete verarbeitet wurden. Diese Wartezeit an der Pufferwarteschlange wird als Warteschlangenverzögerung bezeichnet. Die Bestimmung dieser Verzögerung ist nicht ganz einfach. Mit der folgenden Formel lässt sich die durchschnittliche Warteschlangenverzögerung jedoch in etwa berechnen. Unter Verwendung der Paketankunftsrate am Puffer > und der Dienstrate . ist die durchschnittliche Warteschlangenverzögerung

T = 1

/’

. − >

Die Gesamtverzögerungszeit am Verbindungspunkt kann mit der folgenden Formel berechnet werden:

T = Tt’ + Tp’ + T,p’ + T/’

### Grundlagen der Kommunikationstechnik und Codierungstheorie

Computer sind digitale Systeme und in einem Computernetzwerk werden Daten digital gespeichert, dargestellt und übertragen. Dabei werden die übermittelten Daten jedoch als analoges Signal über das physikalische Medium weitergegeben. Um die Vorgänge bei der digitalen Kommunikation zu verstehen, ist es wichtig, sich mit den Eigenschaften digitaler und analoger Signale auseinanderzusetzen.

###### Signaltypen

Ein Signal ist eine Funktion, die Informationen zu einem physikalischen Phänomen bereitstellt. In der Nachrichtentechnik ist ein Signal eine zeitvariante Spannungs-, Strom- oder elektromagnetische Welle, mit der Informationen übermittelt werden. Im Allgemeinen ist die Spannung (0) die abhängige Variable eines Signals und die Zeit (t) die unabhängige Variable. Signale lassen sich basierend auf bestimmten Merkmalen klassifizieren [Monolithic Power Systems (MPS), ohne Jahresangabe].

Kontinuierliche und diskrete Signale

Kontinuierliche und diskrete Signale werden auch als kontinuierliche und diskrete Zeitsignale bezeichnet. Bei einem kontinuierlichen Signal wird die unabhängige Variable Zeit (t) kontinuierlich gemessen. Ist das Signal diskret, sind für die unabhängige Variable Zeit (t) ausschließlich diskrete Werte zulässig.

Periodische und aperiodische Signale

Wenn die Signalamplitude sich nach einer bestimmten Zeit wiederholt, wird das Signal als periodisches Signal bezeichnet. Die Amplitude eines aperiodischen Signals hingegen wiederholt sich nicht regelmäßig. Ein kontinuierliches periodisches Signal lässt sich als 0 t = 0 t + 1 · T ausdrücken, ein diskretes periodisches Signal als 0 n = 0 n + 1 · 2 . Dabei gilt 1 = 1, 2, 3, ….

Analoge und digitale Signale

Bei einem analogen Signal ist der Wert der abhängigen Variable kontinuierlich. Die Signalamplitude kann also eine beliebige reelle Zahl sein. Bei der Amplitude eines digitalen Signals hingegen kann es sich nur um feste diskrete Werte aus einem vordefinierten Satz an Werten handeln. Allgemein sind bei diskreten Signalen sowohl die Zeit als auch die Amplitude diskret.

Zusammengesetzte Signale, modulierende Signale, Trägersignale und modulierte Signale

Für ein zusammengesetztes Signal werden mehrere Sinuswellen kombiniert. Bei einem modulierenden Signal handelt es sich um ein digitales Signal, das die zu übermittelnde Nachricht bzw. die zu übermittelnden Daten darstellt. Das modulierende Signal wird mit einer reinen Sinuswelle mit der gewünschten Frequenz kombiniert, um das modulierte Signal zu erzeugen. Die Sinuswelle wird als Trägersignal bzw. Träger bezeichnet.

Computernetzwerke

Gerade und ungerade Signale

Ein Signal 0 t ist ein gerades Signal, wenn es die Eigenschaft 0 t = 0 −t aufweist. Für ein ungerades Signal hingegen gilt v(t) = -v(-t).

###### Codierungstheorie

Bei der digitalen Kommunikation kann die Codierung für verschiedene Zwecke genutzt werden. Dazu zählen u. a. die Komprimierung, die Verschlüsselung, die Fehlererkennung und -korrektur, die Konvertierung von analogen in digitale Signale sowie die Datenübertragung. Das Codierverfahren zum Konvertieren eines analogen Signals in ein digitales Signal wird als Quellencodierung bezeichnet. Bei der Kanalcodierung werden redundante Bits zu den Datenbits hinzugefügt, um Bitfehler zu erkennen und zu korrigieren. Das für die Datenübermittlung verwendete Codierverfahren, also das Verfahren zum Konvertieren von digitalen Daten in digitale Signale, wird als Leitungscodierung bezeichnet.

Quellencodierung

Bei der Quellencodierung werden Daten oder Signale in einen Bitstrom umgewandelt. Das einfachste Beispiel der Quellencodierung ist die Puls-Code-Modulation (PCM). Bei der Puls-Code-Modulation werden die folgenden drei Schritte ausgeführt, um ein analoges Signal in einen digitalen Bitstrom umzuwandeln:

1. Bei der „Abtastung“ wird die Amplitude des analogen Signals periodisch gelesen. Dieser Schritt erfolgt, um das zeitkontinuierliche Signal in ein zeitdiskretes Signal umzuwandeln.
2. Bei der „Quantisierung“ werden die Abtastwerte in ganzzahlige Werte umgewandelt, die auf einem vordefinierten Satz an ganzzahligen Werten basieren. Dieser Schritt erfolgt, um die analogen Amplitudenwerte in diskrete oder digitale Werte zu konvertieren.
3. Bei der „Codierung“ werden quantisierte Werte als Binärwerte dargestellt. Wenn die Quantisierung über vier Stufen verfügt, sind die quantisierten Werte 0, 1, 2 und 3. Die entsprechenden Binärwerte lauten 00, 01, 10 und 11. Bei 2 Quantisierungsstufen sind log2 2  Bits zur Darstellung eines quantisierten Werts erforderlich.

###### Wichtige Begriffe und Konzepte

Bandbreite

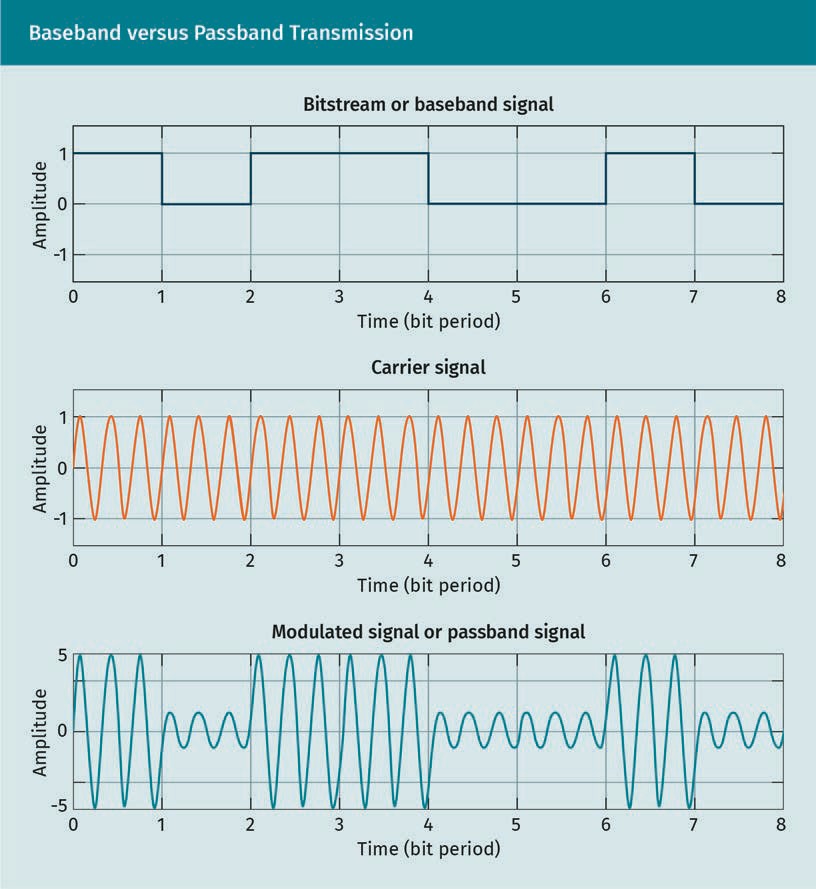
Bei der digitalen Kommunikation setzt sich das übertragene Signal aus mehreren Sinuswellen zusammen. Der Frequenzbereich der Sinuswellen wird als Frequenzbandbreite bzw. Bandbreite des übertragenen Signals bezeichnet. Die Maßeinheit der Frequenzbandbreite ist Hertz 34. In Computernetzwerken ist die Bandbreite die Kapazität einer Verbindung, also die maximale Anzahl an Bits, die pro Sekunde über diese Verbindung übertragen werden können. Die Netzwerkbandbreite wird auch als digitale Bandbreite bezeichnet.

Basisband- und Durchlassbandsignale

Das Basisband ist die Bandbreite des übertragenden Signals, bevor dieses mithilfe eines Trägers moduliert wird. Mit anderen Worten: Das Basisband ist die Bandbreite des digitalen Signals des Informationssignals. Das Durchlassband ist die Bandbreite des übertragenen zusammengesetzten Signals nach der analogen Modulation.

Basisband

Diese Art der Übermittlung wird in Ethernet-LANs verwendet. Im WLAN findet die Durchlassbandübertragung Anwendung.



Rauschen

Rauschen ist ein unerwünschtes Signal, durch welches das gewünschte Signal beeinträchtigt wird. Die Amplitude dieses Signals ist für einen bestimmten Zeitraum indeterministisch. Das additive weiße gaußsche Rauschen (Additive White Gaussian Noise, AWGN) ist die einfachste Modellierung des Rauschens bei der digitalen Kommunikation. Das weiße Rauschen weist bei allen Frequenzen dieselbe Intensität und die Gauß-Verteilung auf.

Shannon-Hartley-Gesetz

Das Shannon-Hartley-Gesetz besagt, dass sich die Kanalkapazität (6) in Bits pro Sekunde bei einer Bandbreite des Durchlassbands von 5 mithilfe der folgenden Gleichung bestimmen lässt:

6 = 5log2 1 + )/2 = 5log2 1 + )2R

Computernetzwerke

Dabei ist ) die Signalstärke und 2 die Rauschleistung. Der Signal-Rausch-Abstand (Signal-to-Noise Ratio, SNR) wird üblicherweise in Dezibel angegeben ’5 (Tanenbaum & Wetherall, 2014).

Nyquist-Theorem

Das Nyquist- oder Abtasttheorem besagt, wie oft ein analoges Signal abgetastet werden sollte, um es in ein digitales Signal umzuwandeln. Laut diesem Theorem muss die Abtastrate mindestens doppelt so hoch sein wie die maximale Frequenz des zusammengesetzten Signals (Tanenbaum & Wetherall, 2014).

Übertragungsmodi

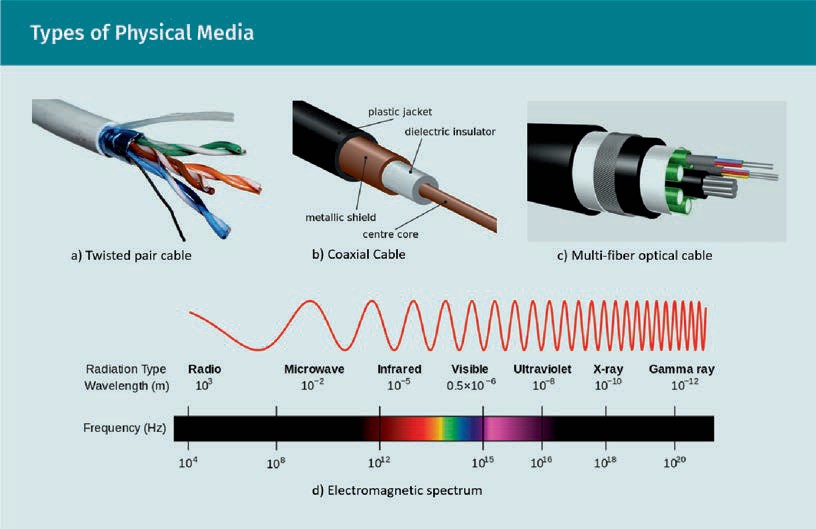
Systeme, bei denen Daten gleichzeitig in beide Richtungen übertragen werden können, werden als Vollduplexsysteme bezeichnet. Wenn die Datenübertragung jeweils nur in eine Richtung möglich ist, spricht man von einer Halbduplexverbindung. Bei einem Simplexsystem wird die Datenübertragung lediglich in eine Richtung unterstützt (Forouzan, 2013).

### Die physikalische Schicht: Übertragungsmethoden und -medien

In einem digitalen System werden alle Daten in Form von binären Zahlen dargestellt. Beim Austausch von Daten zwischen zwei Geräten werden also im Wesentlichen eine Reihe von Nullen und Einsen ausgetauscht. Diese beiden Geräte können über ein geführtes oder ungeführtes Medium miteinander verbunden sein, das als physikalische Schicht bezeichnet wird. Bei der Ausbreitung über die physikalische Schicht werden die Daten als Energie übertragen. Abhängig von den Merkmalen des Mediums können die übermittelten Daten auf unterschiedliche Weise dargestellt werden. Auch die Dienstqualität des Netzwerks wird direkt durch die Merkmale der physikalischen Schicht (z. B. Ausbreitungsgeschwindigkeit, Dämpfung und Rauschen) beeinflusst. Die Dienstqualität umfasst Verzögerung, Jitter, Bitfehlerrate, Kanalkapazität und Durchsatz.

###### Arten von physikalischen Schichten

Bei der Übertragung eines Pakets vom Sender zum Empfänger kann das Paket eine Reihe unterschiedlicher physikalischer Medien durchlaufen. Die Medien der physikalischen Schicht lassen sich in zwei Kategorien untergliedern: geführte und ungeführte Medien. Geführte Medien sind in erster Linie Twisted-Pair-Kupferkabel, Koaxialkabel und Glasfaserkabel. Zu den ungeführten Medien zählt üblicherweise die Freiraumübertragung, also die Weitergabe der Daten als elektromagnetische Wellen mit unterschiedlichem Frequenzbereich (Funkwellen, Mikrowellen und Infrarotstrahlung). Die Installation geführter Medien ist in der Regel kostengünstiger als die Verwendung von ungeführten Medien. Nachfolgend ist eine Darstellung der verschiedenen Arten physikalischer Medien gezeigt.



Twisted-Pair-Kabel

Twisted-Pair-Kabel, also Kabel mit verdrillten Adernpaaren, werden aus isolierten Kupferdrähten hergestellt, über die Daten als elektrische Signale oder Spannungswellen übertragen werden. Über eine Ader wird die Spannungswelle übertragen, die andere Ader dient der Erdung. Durch die Spannungsdifferenz zwischen den beiden Adern werden dann die Daten dargestellt (als Nullen oder Einsen). Kupferdrähte können durch externe Rauschquellen wie Hitze oder elektromagnetische Wellen beeinflusst werden. Wenn sich diese Quellen ungleich auf die beiden Adern auswirken, kann die Spannungsdifferenz der Adern zur Ausgabe falscher Daten führen. Um diesen Effekt zu verhindern, werden die Adern miteinander verdrillt. Twisted-Pair-Kabel können geschirmt oder ungeschirmt sein. Geschirmte Twisted-Pair-Kabel sind mit einer Metallschicht ummantelt, um elektromagnetisches Rauschen zu reduzieren. Diese Art von Kabel wird üblicherweise für ein LAN verwendet. Die Datenrate variiert allgemein zwischen 100 MBit/s und 1 GBit/s mit einer Bitfehlerrate (BER) von 10-7 bis 10-9. Der Nachteil dieser Kabel ist, dass sie relativ leicht „angezapft“ werden können (Solomon & Kim, 2021; Tanenbaum & Wetherall, 2014).

Koaxialkabel

Der Aufbau eines Koaxialkabels unterscheidet sich vom Aufbau eines Twisted-Pair-Kabels. Der Kern eines Koaxialkabels ist im Wesentlichen ein isolierter Kupferdraht, der das Signal überträgt. Dieses Kabel ist mit einer Metallfolie ummantelt, die der Erdung dient. Außerdem schützt eine Metallummantelung das Kabel gegen externes Rauschen. Koaxialkabel bieten im Vergleich zu anderen Kabeltypen eine höhere Frequenzbandbreite. Gleichzeitig kommt es jedoch auch zu höheren Dämpfungseffekten, sodass die Signalstärke bei der Langstreckenkommunikation mithilfe von Repeatern verstärkt werden muss. Koaxialkabel werden häufig von Telefon-, TV- und Internetanbietern eingesetzt. Die Datenrate liegt bei bis zu 1 GBit/s mit einer Bitfehlerrate von 10-9 (Solomon & Kim, 2021; Tanenbaum & Wetherall, 2014).

Computernetzwerke

Glasfaserkabel

Bei Glasfaserkabeln werden Daten als optische Signale weitergegeben. Der Aufbau eines Glasfaserkabels ähnelt dem eines Koaxialkabels. Die beiden Kabeltypen unterscheiden sich jedoch darin, dass das Glasfaserkabel nicht aus Metall, sondern aus flexiblen Glasfasern besteht. Der Kern des Kabels ist für einen geringeren Brechungsindex ummantelt, sodass das Licht ins Innere geleitet wird. Glasfaserkabel zeichnen sich durch ein geringes Gewicht und eine äußerst geringe Dämpfung sowie eine Bitfehlerrate von 10-11 bis 10-13 aus. Darüber hinaus können sehr hohe Datenraten von mehreren Hundert GBit/s erreicht werden. Zudem kann die Übertragung nicht durch elektromagnetisches Rauschen beeinträchtigt werden und die Kabel können nur schwer „angezapft“ werden. Da sich Glasfaserleitungen für die Langstreckenkommunikation eignen, werden sie häufig als Backbone in Internet- und Telefonnetzen verwendet. Der Nachteil dieser Verbindungen ist, dass sie leicht beschädigt werden können, mit einem höheren Wartungsaufwand einhergehen und die Installation kostenintensiv ist (Solomon & Kim, 2021; Tanenbaum & Wetherall, 2014).

Funkwellen

Bei der Kommunikation über Funkwellen werden Daten in erster Linie über elektromagnetische Wellen im Frequenzbereich zwischen 3 Kilohertz (kHz) und 300 Gigahertz (GHz) übertragen. Wellen mit einer Frequenz zwischen 3 kHz und 1 GHz werden als Funkwellen bezeichnet. Ein wesentliches Merkmal der Kommunikation über Funkwellen ist, dass die Daten ungerichtet über drahtlose Medien, also im Freiraum weitergegeben werden. Funkwellen mit geringer Frequenz (also im Bereich zwischen 3 kHz und 3 MHz) werden als Bodenwellen bezeichnet. Der Grund dafür ist, dass sich diese Wellen an der Erdoberfläche ausbreiten und ihrer Krümmung folgen. Im Bereich von 3 bis 30 MHz werden Funkwellen als Raumwellen bezeichnet. Diese Wellen werden zwar vom Boden absorbiert, gelangen durch eine Brechung an der Ionosphäre jedoch wieder zur Erde zurück. Man spricht in diesem Fall auch von „Skip“-Ausbreitung. Der Ausbreitungsmodus einer Funkwelle mit hoher Frequenz (über 30 MHz) wird als Sichtkommunikation oder Sichtausbreitung bezeichnet. Dies bedeutet, dass es aufgrund von Hindernissen zwischen dem Sender und dem Empfänger zu einer hohen Dämpfung oder Blockierung des übertragenen Signals kommen kann (Tanenbaum & Wetherall, 2014).

Mikrowellen

Bei elektromagnetischen Wellen im Frequenzbereich zwischen 1 GHz und 300 GHz spricht man von Mikrowellen. Der Ausbreitungsmodus dieser Wellen ist die Sichtausbreitung. Eine Reihe von Frequenzbereichen im Mikrowellenbereich sind als ISM-Bänder (Industrial-Scientific-Medical) für Industrie, Wissenschaft, Medizin reserviert. Die am häufigsten verwendeten Frequenzen des ISM-Bands sind die WLAN-Frequenzen 2,4 GHz und 5 GHz. Für Bluetooth-Verbindungen und die Nahfeldkommunikation werden ebenfalls ISM-Frequenzen verwendet. Mikrowellen im Frequenzbereich zwischen 3 GHz und 300 GHz heißen Millimeterwellen (mmWave), da die Wellenlänge zwischen 10 mm und 1 mm liegt. Da Millimeterwellen häufig durch Feuchtigkeit, Regen und Sauerstoff absorbiert werden, eignen sich diese Wellen für die Nahbereichskommunikation. Bei der Kommunikation im Nahbereich wird die Frequenzwiederverwendung deutlich verbessert. Die Datenübertragungsrate von Millimeterwellen kann bis zu mehreren GBit/s betragen (Tanenbaum & Wetherall, 2014).

Infrarotwellen

Der Frequenzbereich von Infrarotwellen liegt zwischen 300 GHz und 400 Terahertz (THz). Da Infrarotwellen keine Wände durchdringen können, eignen sie sich für gesicherte Sichtverbindungen im Nahbereich (Tanenbaum & Wetherall, 2014). Ein gängiges Beispiel für die Infrarotkommunikation ist eine Fernbedienung.

###### Moderne Breitbandtechnologien

DSL- (Digital Subscriber Line) und Kabelverbindungen sind die aktuell am häufigsten verwendeten kabelgebundenen Breitbandtechnologien. Im Vergleich zu Kupfertelefonleitungen bieten DSL-Verbindungen eine höhere Datenübertragungsrate. Die DSL-Technologie ist in einer Vielzahl von Varianten verfügbar. Am häufigsten wird ADSL (Asymmetric Digital Subscriber Line) genutzt. Kabelbreitbandverbindungen bieten eine schnelle Datenübertragung über TV-Koaxialkabel aus Kupfer. Eine weitere, glasfaserbasierte Breitbandlösung ist FTTH (Fiber To The Home). Auch diese Technologie ist in verschiedenen Varianten verfügbar, die unter dem Sammelbegriff Fiber To The X (FTTX) zusammengefasst werden. LTE (Long-Term Evolution) ist eine drahtlose Lösung für Breitbandverbindungen, die eine schnelle Datenübertragung über Funkwellen im Mobilfunknetz ermöglicht. In der folgenden Tabelle werden die Eigenschaften der physikalischen Schicht verschiedener Breitbandlösungen verglichen (Kurose & Keith 2017; Oksman et al., 2016; Conformance Speciﬁcation Radio, 2011).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Breitbandtechnologien: physikalische Schichten im Vergleich | | | | |
|  | DSL | Kabel | LTE | FTTH |
| Physikalische Schicht | Twisted-Pair-Kupferdraht | Koaxialkabel, Hybrid Fiber Coax | Funkwelle 410 -  2.495 MHz | Glasfaserkabel |
| Downlink-Bitrate | Allgemein 256 KBit/s -  100 MBit/s;  Maximal 1 GBit/s | Bis zu  42,8 MBit/s | Bis zu 100 MBit/s | Durchschnittlich 20 MBit/s bis hin zu 10 GBit/s |

Zusammenfassung

Ein Computernetzwerk ist eine Sammlung von Computinggeräten, die miteinander kommunizieren. Zur Verbindung dieser Geräte werden Kupferkabel, Glasfaserkabel und elektromagnetische Wellen sowie Verbindungsgeräte wie Router, Hubs, Bridges, Switches und Repeater eingesetzt. Die Darstellung und der Austausch von Daten erfolgen in digitaler Form. Analoge Informationen werden mithilfe der Quellencodierung in digitale Daten umgewandelt. Dieser Vorgang umfasst drei Phasen: Abtastung, Quantisierung und Codierung. Bei der Codierung werden Daten mithilfe binärer Bits dargestellt. Dieser Vorgang kann auch eine Komprimierung und Verschlüsselung umfassen. Die Schritte zum Konvertieren von digitalen Daten in ein Signal werden als digitale Modulation oder Leitungscodierung bezeichnet. Wird ein Signal mit einer Hochfrequenzträgerwelle kombiniert, so spricht man von Modulation. Dieses kombinierte Signal ist ein zusammengesetztes Signal aus mehreren Sinuswellen. Die Bandbreite des digitalen Signals heißt Basisband, die Bandbreite des zusammengesetzten Signals nennt man Durchlassband.

Computernetzwerke

Das Nyquist-Theorem besagt, dass die Abtastrate mehr als das Doppelte der Signalbandbreite betragen sollte. Während der Signalausbreitung wird das Signal durch Stör- oder Rauschsignale beeinträchtigt, die Bitfehler verursachen können. Die Kapazität eines rauschgestörten Kanals kann mithilfe des Shannon-Hartley-Gesetzes berechnet werden. Die Dienstqualität eines Netzwerks hängt von verschiedenen Parametern wie der Bitfehlerrate, dem Durchsatz und der Verzögerung ab. Es gibt vier wesentliche Verzögerungsarten: Ausbreitungsverzögerung, Übertragungsverzögerung, Warteschlangenverzögerung und Verarbeitungsverzögerung.



# Lektion 2

## Kommunikationsprotokolle

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… die übermittelten und empfangenen Pakete sowie das Protokoll zu analysieren

… die Konzepte von Netzwerkschichten, Diensten und Protokollen anzuwenden.

… die Grundlagen von Diensten, Adressierung und Protokollen für die Bitübertragungs-, Vermittlungs- und Transportschicht darzulegen.

DL-E-DLMCSNDS01-U02

1. Kommunikationsprotokolle

### Einführung

Wenn man Informationen senden möchte, z. B. in Form eines Briefs, so besteht das primäre Ziel des Senders bzw. Absenders darin, die Informationen an den Empfänger zu übermitteln. Und zum Übermitteln dieser Informationen müssen eine Reihe von Schritten ausgeführt werden. Zunächst werden die logischen Informationen in physische Informationen umgewandelt, indem sie auf Papier gebracht werden. Dabei schreibt der Absender nicht nur die zu übermittelnden Informationen auf, sondern fügt auch den Namen des Empfängers, seinen eigenen Namen, eine Unterschrift, das Datum und den Ort hinzu. Nachdem der Brief verfasst wurde, kann die Nachricht nicht direkt abgeschickt werden. Stattdessen muss das Schreiben zunächst gefaltet und in einen Umschlag gelegt werden, um den Brief zu schützen und zu verhindern, dass Dritte die Inhalte lesen. Anschließend müssen die Adressen des Absenders und des Empfängers außen auf den Umschlag geschrieben werden. Diese Informationen umfassen nicht nur die Namen, sondern eine Reihe weiterer Angaben, die in festgelegter Reihenfolge aufgeführt werden (z. B. Name, Zimmer- oder Wohnungsnummer, Hausnummer, Straße, Ort und Land). Außerdem muss möglicherweise noch eine Briefmarke aufgeklebt werden.

All das beschreibt lediglich den ersten Schritt beim Versenden eines Briefs. Anschließend sind Postangestellte, Postboten, Fahrer, Piloten und verschiedene weitere Personen involviert, die jeweils eine bestimmte Abfolge von Arbeitsschritten einhalten müssen, um den Brief zuzustellen. Dieser Ablauf wiederum lässt sich in eine Reihe von Schritten bzw. Schichten unterteilen. In jeder dieser Schichten werden unterschiedliche Aufgaben ausgeführt, um den Brief an das Ziel zu transportieren. Und für jede dieser Aufgaben gelten andere Regeln und Vorgaben.

Ein Computernetzwerk und die digitale Kommunikation sind mit diesem Ablauf für den Briefversand vergleichbar. Die Router in einem Netzwerk sind mit den Postfilialen vergleichbar, die Netzwerkverbindungen mit den Straßen und Autobahnen zwischen den Postfilialen. Die Endgeräte entsprechen den Häusern und Gebäuden, eine E-Mail oder ein Editor sind das Gegenstück zu Papier und Tintenfüller. Das Falten des Briefs steht für die Datenkomprimierung und der Umschlag für die Kryptografie. Und auch bei der digitalen Kommunikation muss eine bestimmte Adressenhierarchie beachtet werden. Wie der Versand eines Briefs umfasst auch der Vorgang zum Senden von Daten mehrere Schritte oder Schichten, in denen unterschiedliche Aufgaben bzw. Dienste ausgeführt und Regeln eingehalten bzw. Protokolle verwendet werden.

### Das ISO/OSI-Referenzmodell

Die Internationale Organisation für Normung (International Organization for Standardization) hat ein standardisiertes OSI-Modell (Open System Interconnection) definiert, das die folgenden sieben Schichten für die Kommunikation und Datenübertragung in Computernetzwerken umfasst:

* Schicht 7: Anwendungsschicht
* Schicht 6: Darstellungsschicht
* Schicht 5: Sitzungsschicht
* Schicht 4: Transportschicht
* Schicht 3: Vermittlungsschicht

Kommunikationsprotokolle

* Schicht 2: Sicherungsschicht
* Schicht 1: Bitübertragungsschicht

Die oberste Schicht des ISO/OSI-Modells ist die Anwendungsschicht, die unterste Schicht heißt Bitübertragungsschicht. Wenn ein Sender eine Nachricht sendet, wird diese durch alle Schichten weitergegeben und verarbeitet: von der Anwendungsschicht über die Sicherungsschicht bis hin zur Bitübertragungsschicht. Auf Empfängerseite durchläuft die Nachricht die Schichten von der Sicherungsschicht bis zur Anwendungsschicht. Diese sieben Schichten existieren nur auf Endgeräten. Auf den Verbindungsgeräten sind die oberen vier Schichten nicht vorhanden. Softwareanwendungen werden nicht auf Verbindungsgeräten, sondern ausschließlich auf Endgeräten installiert. Router verfügen z. B. lediglich über die unteren drei Schichten (Vermittlungs-, Sicherungs- und Bitübertragungsschicht), Switches nur über zwei Schichten (Sicherungs- und Bitübertragungsschicht).

###### Schicht 7: Anwendungsschicht

Die Anwendungsschicht stellt Funktionen für Softwareanwendungen bereit und arbeitet mit eigenen Protokollen. Endbenutzer kommunizieren über Softwareanwendungen wie E-Mail-Programme, Webbrowser und Messenger-Dienste für Audio- oder Videonachrichten miteinander. Abhängig von der Art des Diensts kommunizieren die auf Endgeräten installierten Anwendungen über verschiedene Protokolle miteinander. Für Webdienste wird z. B. HTTP (Hypertext Transfer Protocol) verwendet, für E-Mail-Dienste SMTP (Simple Mail Transfer Protocol). Die Anwendungsschicht dient in erster Linie der Kommunikation zwischen Endbenutzern. Dazu werden die auf den Geräten der Endbenutzer installierten Softwareanwendungen miteinander verbunden und Protokolle der Anwendungsschicht verwendet. Darüber hinaus werden basierend auf den Anwendungsdiensten verschiedene Dienste bereitgestellt (z. B. zur Benutzeranmeldung, zum Hoch- und Herunterladen von Nachrichten und zur Dateiübertragung).

###### Schicht 6: Darstellungsschicht

Auf Senderseite empfängt die Darstellungsschicht die Nachricht von der Anwendungsschicht und formatiert sie mithilfe verschiedener Codierungsverfahren mit der gewünschten Datenstruktur. Bei dieser Codierung werden die Daten zudem für die nachfolgende Schicht komprimiert. Auf Empfängerseite werden in dieser Schicht Daten von der Sitzungsschicht empfangen, dekomprimiert, decodiert und für die Darstellung in der Anwendungsschicht formatiert. Beispiele für Protokolle der Darstellungsschicht sind SSL (Secure Sockets Layer), HTTP/HTML (Agent), FTP (File Transfer Protocol), AppleTalk Filing Protocol und Telnet (Knipp et al., 2002).

###### Schicht 5: Sitzungsschicht

Die Sitzungsschicht ist für das Aufbauen, Aufrechterhalten und Beenden von Kommunikationssitzungen zwischen Endbenutzern verantwortlich. Darüber hinaus werden die Endbenutzer in dieser Schicht für verschiedene Kommunikationsmodi (Simplex, Halbduplex oder Duplex) synchronisiert. So kann Knoten A z. B. Daten an Knoten B übertragen, eine Übertragung von Daten in die umgekehrte Richtung ist jedoch nicht möglich. Halbduplexverbindungen unterstützen die Kommunikation in beide Richtungen, sodass sowohl Knoten A als auch Knoten B Daten an den jeweils anderen übertragen kann. Allerdings ist diese Übertragung nicht gleichzeitig möglich. Bei der Duplexkommunikation kann die Übermittlung in beide Richtungen

gleichzeitig erfolgen (Forouzan, 2013). Ein gängiges Beispiel für einen Dienst der Sitzungsschicht ist die Benutzeranmeldung bei einem Server über einen Webbrowser. Die Sitzungsschicht stellt die Verbindung her, erfasst Sitzungszeiten und schließt die Verbindung, sobald der Benutzer sich abmeldet oder die Sitzung abläuft.

###### Schicht 4: Transportschicht

In dieser Schicht werden Dienste für eine zuverlässige Datenübertragung bereitgestellt. Auf Senderseite empfängt die Transportschicht Daten von der nächsthöheren Schicht, unterteilt die Daten in Segmente und fügt zusätzliche Bits hinzu, um eine zuverlässige Datenübertragung sicherzustellen. Die Dienste für eine zuverlässige Datenübertragung umfassen Fehlerkorrekturverfahren, Flusskontrolle, Überlastungskontrolle sowie die Erkennung von Datenverlust und die erneute Übermittlung dieser Daten.

###### Schicht 3: Vermittlungsschicht

Die Vermittlungsschicht empfängt Daten von der Transportschicht und fügt Routinginformationen zu diesen Daten hinzu. Die beiden wichtigsten Dienste dieser Schicht sind Paketweiterleitung und Routing. Bei der Paketweiterleitung werden Daten in erster Linie an den geeigneten Ausgangsport des Routers weitergeleitet. Der Routingdienst bestimmt den besten Pfad zwischen den Endbenutzern.

###### Schicht 2: Sicherungsschicht

Die Sicherungsschicht ermöglicht den Übergang von logischen zu physikalischen Daten. In dieser Schicht werden Daten als logische Bits von der Vermittlungsschicht empfangen. Diese logischen Bits werden in physikalische Bits umgewandelt (in Form von elektrischen oder elektromagnetischen Signalen). Außerdem bietet diese Schicht eine Zugriffssteuerung (Media Access Control, MAC) und Synchronisierung zwischen Sender und Empfänger. Wie bei der Transportschicht werden zudem Mechanismen für Fehlererkennung und Flusskontrolle bereitgestellt.

###### Schicht 1: Bitübertragungsschicht

Die Bitübertragungsschicht ist das verbindende Element zwischen Netzwerkknoten. In dieser Schicht werden Daten als elektrische, elektromagnetische oder optische Signale übertragen. Die beliebtesten physikalischen Verbindungen sind Koaxialkabel, Twisted-Pair-Kabel, Glasfaserkabel, WLAN-Verbindungen, Mikrowellensignale und Funksignale.

###### Protokolldateneinheit und Adressierung

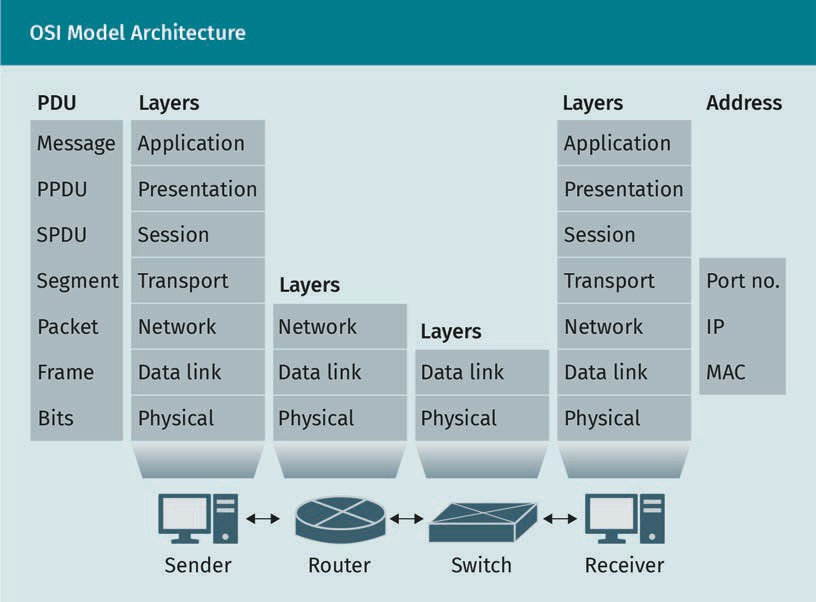
Wenn ein Sender Daten überträgt, durchlaufen diese Daten alle sieben Schichten. In jeder Schicht werden die Daten, die von der nächsthöheren Schicht empfangen wurden, um zusätzliche Bits erweitert. Die Dateneinheit einer Schicht wird üblicherweise als Protokolldateneinheit (Protocol Data Unit, PDU) bezeichnet. In den verschiedenen Schichten werden unterschiedliche Elemente als PDU verwendet, die sich aus je zwei Komponenten zusammensetzen: Header und Payload. Die von der nächsthöheren Schicht empfangenen Daten werden als Payload bezeichnet, die zu den empfangenen Daten hinzugefügten Bits sind die Headerbits bzw. der Header. Die Headerbits umfassen die

Kommunikationsprotokolle

Dienstinformationen sowie die Schichtadressen von Sender und Empfänger. Ein Header der Transportschicht umfasst z. B. die Portnummern, bei der Vermittlungsschicht sind die IP-Adressen enthalten und der Header der Sicherungsschicht enthält die MAC-Adressen von Sender und Empfänger. Diese Elemente sind im Folgenden näher beschrieben:

* Eine Portnummer ist eine 16 Bit große Nummer zur Adressierung eines Anwendungsprogramms, das auf einem sendenden oder empfangenden Endknoten installiert ist. Der Port 80 wird beispielsweise für HTTP-Webdienste (Hypertext Transfer Protocol) und der Port 993 für sichere Mailübertragungen eingesetzt. Eine Liste mit Portnummern der Transportschicht finden Sie auf der Website der Internet Assigned Numbers Authority (Internet Assigned Numbers Authority (IANA), ohne Jahresangabe).
* Eine IP-Adresse (Internet Protocol, Internetprotokoll) ist eine 32 Bit lange Nummer, die einem Endgerät zugewiesen ist. Genauer gesagt wird mit dieser Nummer das Betriebssystem des Endknotens gekennzeichnet. Jedes Gerät (z. B. ein Laptop, Smartphone oder Router), das mit dem Internet oder einem LAN (Local Area Network) verbunden ist, erhält eine IP-Adresse.
* Eine MAC-Adresse ist eine Nummer aus 48 Bits, die das Gerät eines Netzwerkknotens in der Sicherungsschicht kennzeichnet.
* Die Socket-Adresse ist eine Kombination aus einer IP-Adresse und einer Portnummer (Forouzan, 2013).

Die folgende Abbildung zeigt die Architektur des ISO-Modells, die Schichtnamen, die Namen der Protokolldateneinheiten der einzelnen Schichten sowie die zugehörigen Schichtadressen.



### Sicherungsschicht: Standards und Technologien

Die Sicherungsschicht ist die zweite Schicht und befindet sich unterhalb der Vermittlungsschicht und oberhalb der Bitübertragungsschicht. Diese Schicht wird auf einer Netzwerkkarte (Network Interface Controller, NIC) implementiert. Eine NIC ist eine Hardwarekomponente, mit der die Sicherungsschicht und die Bitübertragungsschicht verbunden werden. Zwei häufig verwendete NICs sind Ethernet- und WLAN-Netzwerkkarten. Oberhalb der Sicherungsschicht werden Daten logisch dargestellt. In der Bitübertragungsschicht werden Daten jedoch als elektrische oder elektromagnetische Wellen dargestellt. Über die Sicherungsschicht werden die logischen und physikalischen Schichten miteinander verbunden. Die Protokolldateneinheit dieser Schicht ist ein Frame. Im Frameheader wird das Datagramm oder Paket der Vermittlungsschicht gekapselt. Die Sicherungsschicht ist in zwei Unterschichten unterteilt: Logical Link Control (LLC) und Media Access Control (MAC). Die LLC-Unterschicht bietet Dienste für Fehlererkennung und Flusskontrolle, die MAC-Schicht Dienste für eine MAC-basierte Paketweiterleitung und Mehrfachzugriffskontrolle.

###### MAC

Die Adresse der Sicherungsschicht, die sogenannte MAC-Adresse, ist eine Hardwareadresse mit 48 Bits bzw. 6 Bytes. Jede Netzwerkkarte verfügt also über eine eindeutige MAC-Adresse. Die IP-Adresse der Vermittlungsschicht ist eine logische Adresse, die vom Benutzer ausgewählt oder bearbeitet werden kann. Die MAC-Adresse hingegen ist auf der Netzwerkkarte „eingebrannt“ und darf nicht von Benutzern geändert werden. Da ein Gerät jedoch über mehrere Netzwerkkarten verfügen kann, sind auch mehrere MAC-Adresse möglich. Auch bei Verwendung von virtuellen Maschinen (VMs) können mehrere MAC-Adressen für ein einziges Gerät verwendet werden. MAC-Adressen werden im Hexadezimalformat dargestellt. Dabei werden die einzelnen Bytes durch einen Doppelpunkt voneinander getrennt. Eine MAC-Adresse, bei der alle Bits auf 1 festgelegt sind (FF:FF:FF:FF:FF:FF), wird Broadcast-Adresse genannt. Das Address Resolution Protocol (ARP) ist ein Protokoll der Vermittlungsschicht, mit dem sich die MAC-Adresse eines Hosts ermitteln lässt, dessen IP-Adresse bekannt ist. Dieser Vorgang ist im Request for Comments 903 (RFC 903) dokumentiert (Finlayson et al., 1984).

###### Fehlererkennung

In der Sicherungsschicht werden basierend auf den Payload-Bitwerten Fehlererkennungsbits (Error Detection Bits, EDBs) berechnet. Diese EDBs werden im Frameheader hinzugefügt. Auf Empfängerseite werden die EDBs erneut berechnet. Weichen die berechneten Fehlererkennungsbits von denen des Senders ab, wird ein Bitfehler ausgegeben. Stimmen die berechneten Fehlererkennungsbits mit denen des Senders überein, so wird kein Bitfehler ausgegeben. In der Sicherungsschicht werden Bitfehler mithilfe einer zyklischen Redundanzprüfung (Cyclic Redundancy Check, CRC) ermittelt. Dabei wählt der Sender zunächst die Anzahl von Bits in EDBs aus (dargestellt mit r). Anschließend wird eine r+1-Bitzahl G *ausgewählt*, der sogenannte Generator. Die Daten D werden dann mit r Nullen aufgefüllt. D’ steht für die mit Nullen aufgefüllten Daten D. Anschließend wird D’ durch G dividiert, um den Rest R zu berechnen. Der Rest R wird in rBits dargestellt, bei denen es sich um die gewünschten EDBs handelt. //Um den Vorgang zu beschleunigen, wird die Berechnung des Restwerts jedoch vereinfacht, indem Subtraktionen mithilfe von XOR-Funktionen (Exklusiv-ODER) durchgeführt werden. Beispiel: Wenn D = 11001010, G = 1011, dann gilt R = Rest von (D/G).

Kommunikationsprotokolle

1011 ) 11001010000 ( 11101

1011

1111010

1011

100010

1011

01110

1011

0101 sodass R = 101

In der Praxis verwendet ein Frame der Sicherungsschicht CRC-Bits mit einer Länge von 32 Bits bzw. 4 Bytes. Die Sicherungsschicht bietet im Allgemeinen keinen Mechanismus zur Fehlerkorrektur. Wenn auf Empfängerseite ein Bitfehler ermittelt wird, wird der Frame einfach verworfen (Tanenbaum & Wetherall, 2014).

###### Mehrfachzugriffsprotokolle

Ein wichtiger Aspekt der Sicherungsschicht ist die Möglichkeit, dass mehrere Benutzer Daten über eine einzige physikalische Ressource übertragen können. Es gibt eine Reihe von Mechanismen für den Mehrfachzugriff, die sich allgemein in die folgenden drei Kategorien einteilen lassen: Ressourcenpartitionierung, abwechselnder Zugriff und wahlfreier Zugriff (Random Access). Bei der Ressourcenpartionierung wird die Ressource (z. B. ein Zeitfenster oder eine Frequenz) in kleinere Abschnitte unterteilt und auf mehrere Benutzer aufgeteilt. Bei Verfahren für einen abwechselnden Zugriff erhalten die Benutzer abwechselnd Zugriff auf eine gemeinsame Ressource. Diese Verfahren werden zentralisiert eingesetzt, d. h., die Ressourcenverteilung wird von einem zentralen Knoten verwaltet. Random-Access-Mechanismen hingegen sind dezentralisiert implementiert, was bedeutet, dass die einzelnen Benutzer nach dem Zufallsprinzip Zugriff auf die gemeinsame Ressource erhalten.

Random-Access-Protokolle

Das CSMA-Protokoll (Carrier Sense Multiple Access) ist eine Lösung für den wahlfreien Zugriff. Dieses Verfahren ähnelt einer Unterhaltung zwischen zwei Menschen: Ein Knoten hört zu, bevor er spricht. Falls ein Knoten erkennt, dass ein anderer Knoten Daten überträgt, wartet er (für eine willkürliche Zeit) mit der Übertragung, bevor er die Datenübertragung erneut initiiert. Dennoch kann es bei CSMA zu Konflikten bzw. Kollisionen kommen. Wenn die beiden Knoten A und B gleichzeitig erkennen, dass der jeweils andere Knoten aktuell keine Daten überträgt, starten möglicherweise beide Knoten gleichzeitig eine Datenübertragung, die dann zu einer Kollision führt. Dies wird als Hidden-Node-Problem bezeichnet. Nachfolgend sind zwei optimierte Versionen von CSMA aufgeführt:

* Bei CSMA/CD (Carrier Sense Multiple Access/Collision Detection) wird ein Mechanismus zur Kollisionserkennung implementiert. Dabei wird der Signal-zu-Rausch-Abstand gemessen. Sobald eine Kollision erkannt wird, wird das Binary Exponential Backoff-Verfahren gestartet: Wenn ein Knoten beim Übertragen eines Frames m Mal eine Kollision feststellt, wartet der Knoten für die Übertragungsdauer von 512×K Bits. Dabei ist K eine zufällig ausgewählte Zahl aus

dem Bereich 0, 1, 2, 3, . . . . 27 − 1 . Die Effizienz von CSMA/CD wird mit 1/ 1 + 5’p/’t bestimmt, wobei ’p und ’t die Ausbreitungs- und Übertragungsverzögerung eines übertragenen Frames sind.

* + //CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance) ist eine //proaktive Lösung, um //Hidden-Node-Probleme zu vermeiden. Bei dieser Version werden RTS- (Request to Send, Sendeanfragen) und CTS-Nachrichten (Clear to Send, Sendeerlaubnis) ausgetauscht (Kurose & Keith, 2017; Comer, 2015).

###### Standards der Sicherungsschicht

Abhängig von den Merkmalen der Bitübertragungsschicht finden in der Sicherungsschicht verschiedene Standards Anwendung. Der bei kabelgebundenen LANs am häufigsten verwendete Standard dieser Schicht ist das Ethernet. Beim Ethernet wird zwischen einer Vielzahl von Varianten unterschieden, darunter 10BASE-T, 10BASE-2, 100BASE-T, 1000BASE-LX, 10GBASE-T und 40GBASE-T, die für LANs mit je 10 MBit/s, 10 MBit/s, 100 MBit/s, 1 GBit/s, 10 GBit/s bzw. 40 GBit/s ausgelegt sind (Comer, 2015). Das „T“ steht für Twisted-Pair-Kabel, „BASE“ für Basisbandübertragung. 10BASE-2 ist für dünne Koaxialkabel standardisiert, 1000BASE-LX für Glasfaserkabel. Ethernet-basierte LANs sind durch die Norm IEEE 802.3 standardisiert. In diesen LANs werden das Random-Access-Protokoll CSMA/CD und die 4-Byte-CRC-Fehlererkennung implementiert. Das am zweithäufigsten verwendete LAN ist ein WLAN, das die Mikrowellenfrequenz 2,4 GHz oder 5 GHz nutzt. WLANs sind durch die Norm IEEE 802.11 standardisiert. Bei diesen Netzwerken wird das Random-Access-Protokoll CSMA/CA verwendet. DOCSIS (Data Over Cable Service Interface Specification) ist ein Standard für Hybrid-Fiber-Koaxialkabel (HFC), bei denen Glasfaser- und Koaxialkabel kombiniert werden. In der Norm IEEE 802.15 sind die Bitübertragungs- (PHY) und die Sicherungsschicht für ein WPAN (Wireless Personal Area Network) standardisiert. Für die unterschiedlichen WPAN-Typen sind verschiedene

Versionen des IEEE 802.15-Standards verfügbar. So gilt 802.15.1 beispielsweise für Bluetooth-basierte WPANs, 802.15.3 für WPANs mit hoher Rate (High Rate WPAN) und 802.15.4 für WPANs mit niedriger Rate (6LoWPAN). Diese Norm wird zudem als Standard für das Internet der Dinge (Internet of Things, IoT) betrachtet. Mit den Normen 802.15.5 und

802.15.6 werden WMAN-Mesh-Netzwerke bzw. drahtlose BANs (Body Area Network) standardisiert (Comer, 2015).

###### Switches

Switches sind Layer-2-Geräte. Das bedeutet, dass sie lediglich über zwei Schichten verfügen: die Bitübertragungs- und die Sicherungsschicht. Aus diesem Grund können Switches Daten basierend auf der MAC-Adresse filtern und an den gewünschten Ausgangsport weiterleiten. Hubs hingegen sind Layer-1-Geräte, die Daten nicht basierend auf der MAC-Adresse filtern und weiterleiten können. Stattdessen leitet ein Hub eingehende Daten an alle Ausgangsports weiter, wodurch unnötiger Overhead im Netzwerk entsteht. Bei Switches wird dieses Problem durch eine MAC-basierte Weiterleitung verhindert.

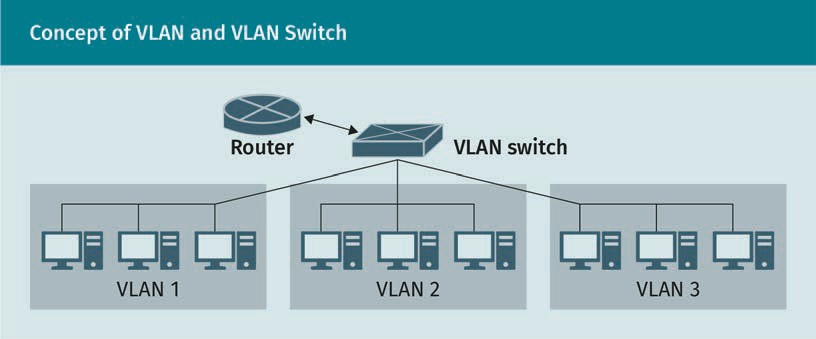
Kommunikationsprotokolle

###### VLAN

Alle Endknoten eines LANs sind üblicherweise mit einem gemeinsamen Switch verbunden. Bei N LANs sollten also auch N Switches vorhanden sein. Darüber hinaus befinden sich alle Knoten eines LANs in derselben geografischen Lage, z. B. auf derselben Etage oder im selben Gebäude. Ein virtuelles LAN (VLAN) hingegen bietet die Möglichkeit, mehrere LANs im selben geografischen Raum mit nur einem Switch aufzubauen. Dieser Switch wird als VLAN-Switch bezeichnet.

Angenommen, ein VLAN-Switch verfügt über 100 Ports. Die Portnummern 2, 3, 6, 10 und 77 bilden gemeinsam

LAN 1, die Portnummern 11, 22, 41, 88 und 50 bilden LAN 2 usw. VLANs sind in der Norm IEEE 802.1Q standardisiert. Dabei wird ein traditioneller Ethernet-Frame um ein VLAN-Tag mit 4 Bytes verlängert (Tanenbaum & Wetherall, 2014; Solomon & Kim, 2021). Ein VLAN darf nicht mit einem virtuellen privaten Netzwerk (VPN) verwechselt werden. Denn im Gegensatz zum VLAN sind die Knoten bei einem VPN nicht unbedingt mit einem gemeinsamen Switch verbunden.

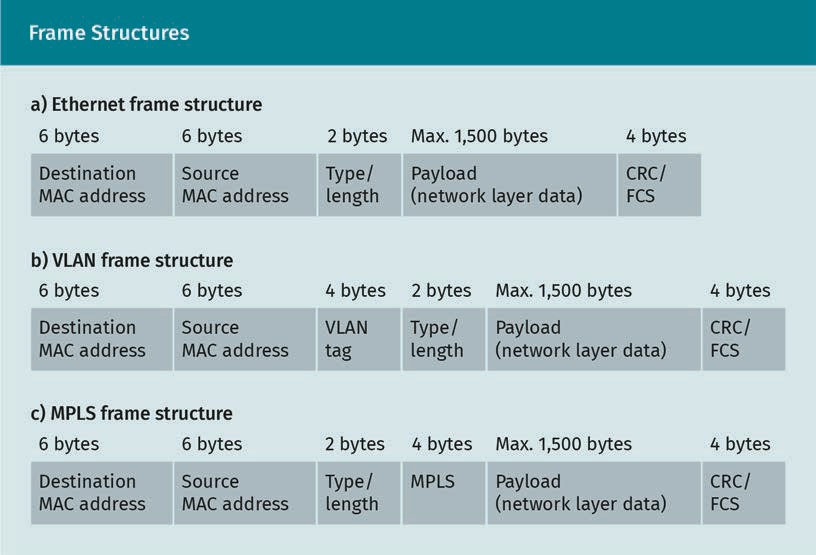


###### MPLS

Das MPLS-Verfahren (Multiprotocol Label Switching) arbeitet zwischen den Schichten 2 und 3. Mit MPLS können Daten basierend auf einem MPLS-Header anstelle einer IP-Adresse über Router weitergeleitet werden. Indem das Lesen des IP-Headers und die Suche in Routingtabellen vermieden werden, beschleunigt dieses Verfahren die Weiterleitung. Zunächst wird ein Routingpfad zwischen der Quelle und dem Ziel gewählt. Wenn in einem traditionellen verbindungslosen Netzwerk ein Paket übermittelt wird, lesen die einzelnen Router die Headerbits und IP-Adressen, wenden den Routingalgorithmus an, durchsuchen Routingtabellen und ermitteln dann den nächsten Hop. Bei diesem Vorgang kommt es zu erheblichen Verarbeitungsverzögerungen. Bei MPLS hingegen wird zunächst ein Pfad zwischen Quelle und Ziel gewählt. Anschließend werden die Daten der Vermittlungsschicht mithilfe eines MPLS-Headers gekapselt, in dem anstelle der langen IP-Adressen ein kurzes Label verwendet wird. Dieses MPLS-Label informiert den Router über den nächsten Hop. MPLS ist in RFC 3031, 6178 und 6790 standardisiert (Rosen et al., 2001).

Virtuelles privates Netzwerk (VPN)

Ein VPN ist ein privates Netzwerk, auf das remote zugegriffen werden kann. Es baut auf einem öffentlichen Netzwerk auf, ist für Unbeteiligte jedoch nicht einsehbar.



Übung

Wireshark ist die beliebteste Open-Source-Software zur Überwachung des Datenverkehrs und Analyse von Protokollen. Mit diesem Tool lassen sich Echtzeitpakete erfassen, die von einem Computer gesendet oder empfangen werden. Über eine grafische Benutzeroberfläche können die Headerfelder und Payloadinformationen der einzelnen Pakete angezeigt werden. Es wird dringend empfohlen, Wireshark herunterzuladen und zu installieren, um Pakete zu erfassen und die Paketinhalte zu untersuchen. Eine Vielzahl erfasster Beispielpakete stehen auf der offiziellen Wireshark-Website zur Verfügung. Informationen zum Herunterladen von Beispielpaketen (.pcap- oder .cap-Dateien) für 802.11-WLANs, 802.1Q-VLANs, MPLS und ARP finden Sie auf der Wireshark Foundation-Website (ohne Jahresangabe). Bei der Untersuchung der Pakete sollte insbesondere auf die Struktur der Headerfelder und -werte geachtet werden.

### Die Vermittlungsschicht: Adressierung und Routing

Die Vermittlungsschicht befindet sich zwischen Sicherungs- und Transportschicht und ist in Betriebssystemen implementiert. Zur Adressierung der Vermittlungsschicht wird die IP-Adresse verwendet, die einen Netzwerkknoten kennzeichnet. Genau genommen kennzeichnet diese Adresse das Betriebssystem eines Netzwerkknotens. Wenn auf einem Computer mehrere Betriebssysteme ausgeführt werden (u. a. auch in virtuellen Maschinen), sind mehrere IP-Adressen möglich. Die Vermittlungsschicht ist sowohl in Endknoten als auch in Routern implementiert. Die beiden wichtigsten Dienste dieser Schicht sind Paketweiterleitung und Routing.

Kommunikationsprotokolle

###### Adressierung in der Vermittlungsschicht

Es gibt zwei Arten von IP-Adressen: IPv4 (Internet Protocol version 4) und IPv6 (Internet Protocol version 6). IPv4 ist das traditionelle Format zur Adressierung in der Vermittlungsschicht. Eine IPv4-Adresse verfügt über vier Oktette bzw. 32 Bits. Dabei wird das folgende Format verwendet: 8 . b . c . ’, wobei die einzelnen Oktette durch einen Punkt getrennt sind. Jedes Endgerät eines Netzwerks und jeder Port eines Routers verfügt über eine eindeutige IP-Adresse.

Subnetzmaske

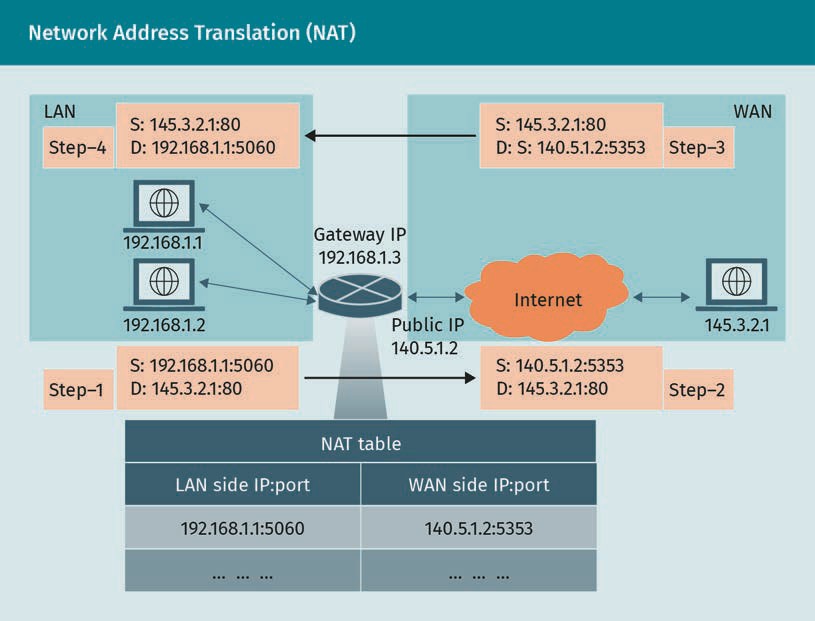
Alle Endknoten oder Hosts, die über einen gemeinsamen Routerport mit dem Internet verbunden sind, weisen ein gemeinsames IP-Muster auf. Dieses gemeinsame Muster wird als Subnetzmaske bezeichnet. Das Netzwerk, das über einen gemeinsamen Routerport mit dem Internet verbunden ist, heißt Subnetz. Die IP-Adresse eines Subnetzhosts wird üblicherweise mit der folgenden

CIDR-Notation (Classless Interdomain Routing) angegeben: 8 . b . c . ’/E.. Dabei gibt e an, dass die Subnetzmaske e Bits enthält und 2(32-e) Hosts in einem Netzwerk zulässig sind. Zur Vereinfachung beträgt die Gesamtzahl von Hosts in einem Subnetz 2(32-e)-2. Bei der Adresse 192.168.0.4/23 stehen die //höchstwertigen 23 Bits beispielsweise für die Subnetzmaske und das Subnetz kann über maximal 29-2 = 510 Hosts verfügen. Der Internetdienstanbieter (Internet Service Provider, ISP) weist eine Subnetzmaske zu, um die Anzahl der Benutzer in einem Netzwerk zu beschränken. Diese Subnetzmaske wird zudem zum Filtern von eingehenden und ausgehenden Paketen in einem Netzwerk verwendet.

Netzwerkadressübersetzung

Bei der Netzwerkadressübersetzung (Network Address Translation, NAT) wird die lokale Host-IP-Adresse im globalen Netzwerk oder Internet verborgen, indem die Portnummer mit der IP-Adresse kombiniert wird. Dieser Schritt wird von einem NAT-Router ausgeführt, der die entsprechende Tabelle zur Neuzuordnung verwaltet. Die folgende Abbildung zeigt ein NAT-Szenario mit einem Router, der über den Port 192.168.1.3 mit einem LAN verbunden ist. Dieses LAN ist zudem über den Port 140.5.1.2 mit dem Internet oder WAN verbunden. Nachfolgend sind die vier Schritte dieses Vorgangs aufgeführt:

1. Der sendende Host mit dem Socket 192.168.1.1:5060 sendet eine Nachricht an den empfangenden Host mit dem Socket 145.3.2.1:80. 192.168.1.1 und 5060 sind die lokale IP-Adresse und die Portnummer des sendenden Hosts.
2. Der NAT-Router übersetzt die lokale IP-Adresse und Portnummer in eine globale (öffentliche) IP-Adresse und einen öffentlichen Port. Die globale IP-Adresse des Senders lautet 140.5.1.2. Dies ist der WAN-seitige Port des Routers. Der Port 5353 wird vom Router willkürlich ausgewählt, um den Socket 192.168.1.1:5060 anzugeben.
3. Vom Quellsocket 145.3.2.1:80 wird eine Nachricht an den öffentlichen Socket 140.5.1.2:5353 gesendet.
4. Anhand der NAT-Tabelle wird die globale Adresse in die lokale Adresse 140.5.1.2:5353 übersetzt.



IPv4

IPv4-Adressen werden traditionell in fünf Klassen unterteilt: A, B, C, D und E. Der erste Oktettbereich für die Klasse A ist 1-126. Bei der Klasse B liegt er zwischen 128

und 191, bei der Klasse C zwischen 192 und 223. Die Klasse D ist für Multicasting reserviert, die Klasse E für die

zukünftige Nutzung.

IPv4

Der IPv4-Adressraum umfasst 32 Bits. Dies bedeutet, dass er insgesamt 232 (ca. 4,3 Milliarden) globale IP-Adressen umfassen kann. Laut dem Cisco Annual Internet Report wird die Gesamtzahl an Internetnutzern bis zum Jahr 2023 bei 5,3 Milliarden Menschen liegen und es werden insgesamt 29,3 Milliarden Geräte verbunden sein (Cisco, 2020). Mit IPv4 kann eine derart große Anzahl an Geräten nicht abgedeckt werden. Um zu verhindern, dass die globalen IP-Adressen vollständig ausgeschöpft werden, wird daher auf die Netzwerkadressübersetzung zurückgegriffen. Denn mithilfe der Netzwerkadressübersetzung können Internetdienstanbieter eine einzige globale IP-Adresse für ein gesamtes Subnetz oder LAN verwenden. Ein weiterer Vorteil von NAT ist, dass die lokalen IP-Adressen im Internet nicht sichtbar sind. Das lokale Netzwerk wird also zusätzlich geschützt. Und wenn ein neuer Internetdienstanbieter für ein LAN ausgewählt wird, können weiterhin dieselben IP-Adressen für die LAN-Geräte verwendet werden.

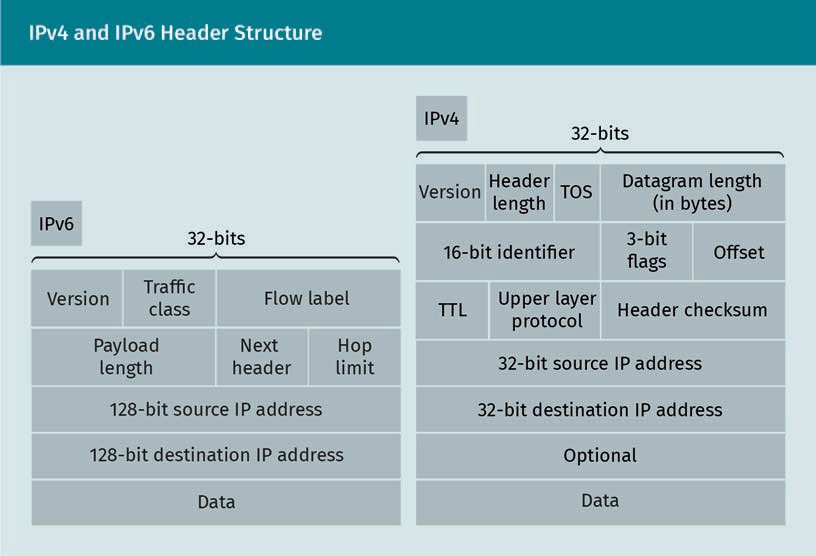
Allerdings geht die Netzwerkadressübersetzung auch mit einigen Nachteilen einher. Das Verbergen lokaler IP-Adressen kann als Sicherheitsrisiko betrachtet werden. Außerdem kommt es durch die Übersetzung zu zusätzlichen Verzögerungen und der gesamte Vorgang wird komplexer. Ein weiterer wichtiger Nachteil ist, dass die Regeln des OSI-Schichtenmodells bei der Netzwerkadressübersetzung nicht eingehalten werden. Denn Router sind Geräte der Vermittlungsschicht und es ist keine Transportschicht vorhanden. Da Portnummern jedoch Adressen der Transportschicht sind, sollten Router eigentlich nicht mit diesen Nummern arbeiten. Außerdem ließe sich das Kapazitätsproblem von IPv4 problemlos mit IPv6 lösen, wodurch die Notwendigkeit der Netzwerkadressübersetzung in Frage gestellt wird.

Kommunikationsprotokolle

IPv6

IPv6 ist eine 128-Bit-Adresse der Vermittlungsschicht, die eingeführt wurde, um das Kapazitätsproblem von IPv4 zu lösen. Mit IPv6 lässt sich die astronomische Zahl von bis zu 2128 = 3,4×1038 eindeutigen IP-Adressen darstellen. Für eine geringere Verarbeitungsverzögerung vereinfacht IPv6 zudem den Datagrammheader der Vermittlungsschicht. Darüber hinaus können Daten basierend auf dem Datenverkehrsfluss gefiltert und weitergeleitet werden.

Doch der Übergang von IPv4 zu IPv6 ist ein langwieriger Prozess. Im April 2021 verwenden mehr als 30 Prozent der Google-Nutzer IPv6 (Google, ohne Jahresangabe). In diesem Zusammenhang soll auch darauf hingewiesen werden, dass die Internet Corporation for Assigned Names and Numbers (ICANN) die IP-Adressen über die Internet Assigned Numbers Authority (IANA) verwaltet und verteilt. Die Rückwärtskompatibilität von IPv6 mit IPv4 wird durch ein als Tunneling oder Tunneln bezeichnetes Verfahren erreicht. Dabei werden IPv4-Header verwendet, um IPv6-Datagramme vollständig zu kapseln.



In der Abbildung oben gibt der TTL-Wert (Time To Live) oder das Hop-Limit die Zeit oder die Hop-Anzahl an, nach der ein Paket abläuft. Der Diensttyp oder die Datenverkehrsklasse gibt den Diensttyp eines Pakets basierend darauf an, ob das Paket für die Warteschlange, das Filtern oder die Weiterleitung priorisiert wird.

DHCP

Wenn ein neues Gerät zu einem Netzwerk hinzugefügt wird, ist dem Gerät zunächst keine IP-Adresse zugeordnet. Das Gerät muss diese Adresse vom Server anfordern. Für diesen Vorgang zum Abrufen einer neuen IP-Adresse wird DHCP (Dynamic Host Configuration Protocol) verwendet. Nachfolgend sind die vier Schritte dieses Vorgangs aufgeführt:

* 1. Ermittlung. Der Client überträgt eine DHCPDISCOVERY-Broadcast-Nachricht mit der Quell-IP-Adresse 0.0.0.0 und Port 68 sowie der Ziel-IP-Adresse 255.255.255.255 und Port 67.
  2. Angebot. Der Server antwortet mit einer DHCPOFFER-Nachricht, die die Ziel-IP-Adresse 255.255.255.255 und den Zielport 68 enthält. Diese Nachricht beinhaltet zudem die IP-Adresse, die dem Client angeboten wird.
  3. Anforderung. Der Client antwortet mit einer DHCPREQUEST-Nachricht, die die Quell-IP-Adresse 0.0.0.0, den Quellport 68, die Ziel-IP-Adresse 255.255.255.255 und den Zielport 67 enthält. Außerdem ist die angeforderte IP-Adresse enthalten. Diese entspricht der IP-Adresse, die vom Server in seiner DHCPOFFER angeboten wurde.
  4. Bestätigung. Der DHCP-Server bestätigt die Anforderungsnachricht, indem er eine DHCPACK-Nachricht mit der Ziel-IP-Adresse 255.255.255.255 und Port 68 sendet.

###### Dienste der Vermittlungsschicht

In der Vermittlungsschicht werden in erster Linie Dienste für Paketweiterleitung und Routing bereitgestellt. Die Paketweiterleitung erfolgt lokal in einzelnen Routern, das Routing hingegen betrifft das gesamte Netzwerk (alle Router und Verbindungen). Bei der Paketweiterleitung wird ein Paket vom Eingangsport des Routers an den richtigen Ausgangsport weitergeleitet. Routing bedeutet, dass eine geeignete Route bzw. ein geeigneter Pfad zwischen Sender und Empfänger ausgewählt wird. Jeder Router verfügt über eine Weiterleitungstabelle, die mithilfe eines Routingalgorithmus erstellt wird. Dabei stehen für verschiedene Zwecke unterschiedliche Arten von Routingprotokollen zur Verfügung.

###### Arten des Routings

Routing ist die Auswahl eines geeigneten Pfads zwischen dem sendenden und dem empfangenden Knoten. Als „Pfad“ werden eine Reihe von Verbindungen bezeichnet, ein „geeigneter Pfad“ bezieht sich üblicherweise auf den kostengünstigsten Pfad. Für die Kosten einer Route können verschiedene Variablen berücksichtigt werden, z. B. Entfernung, Verzögerungen, Durchsatz und Energie. Sie sind umgekehrt proportional zu Durchsatz und Nutzen. Nachfolgend sind einige der vielen verfügbaren Routingverfahren beschrieben.

Dynamisches und statisches Routing im Vergleich

Beim dynamischen Routing wird die Routingtabelle aktualisiert, sobald sich die Netzwerktopologie oder die Verbindungskosten ändern. Im Gegensatz dazu wird die Routingtabelle beim statischen Routing selten aktualisiert. Bei dieser Variante entscheidet der Netzwerkadministrator, wann die Tabelle geändert wird.

Zentralisiertes und dezentralisiertes Routing im Vergleich

Beim zentralisierten Routing erfasst eine zentrale Entität die Routinginformationen aller Router, wendet den Routingalgorithmus an und leitet die aktualisierte Tabelle an jeden Router weiter. Wird ein dezentralisierter Routingalgorithmus verwendet, wird der Algorithmus in jedem Router einzeln ausgeführt. Anschließend aktualisieren die Router ihre Routingtabellen selbst.

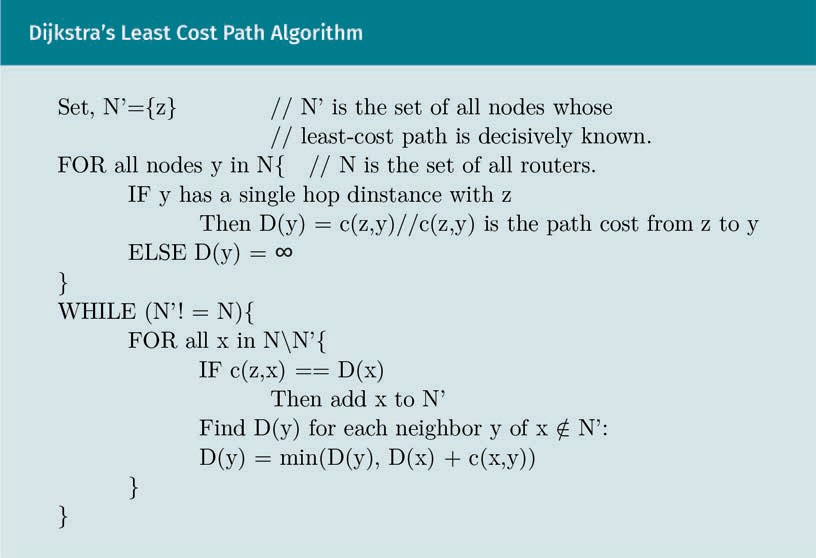
Kommunikationsprotokolle

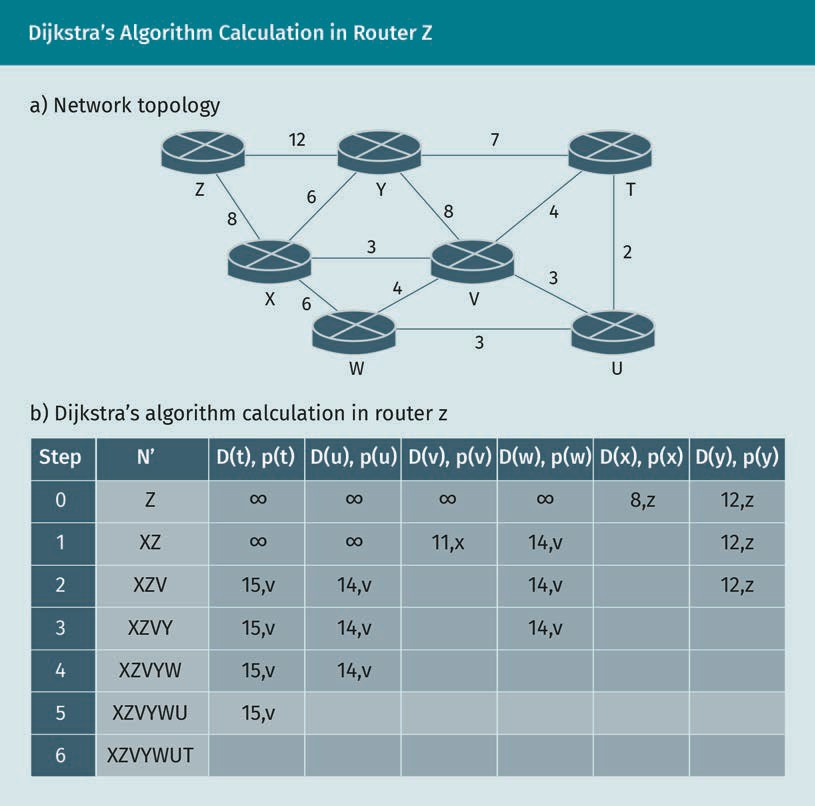
###### Routingalgorithmen

Ein Routingalgorithmus umfasst die einzelnen Berechnungsschritte, um den geeigneten Pfad zu bestimmen. Es gibt zwei Arten von Routingalgorithmen: den Link-State-Algorithmus und den Distanzvektoralgorithmus.

Link-State-Routingalgorithmus

Beim Link-State-Routingalgorithmus tauschen alle Router per Broadcast-Übertragung Routinginformationen untereinander aus. So empfängt letztendlich jeder Router die Routinginformationen zur gesamten Topologie. Dabei wird in jedem Router derselbe Routingalgorithmus angewendet, um einen geeigneten Pfad zu ermitteln und die Weiterleitungstabelle entsprechend zu aktualisieren. Der am häufigsten eingesetzte Link-State-Routingalgorithmus ist der Dijkstra-Algorithmus zur Bestimmung des kostengünstigsten Pfads. Der Dijkstra-Algorithmus ist ein iterativer Algorithmus. Wenn z der Quellknoten ist, D(y) für die Kosten des kostengünstigsten Pfads von z nach y steht, p(y) der Vorgängerknoten von y innerhalb des kostengünstigsten Pfads und N die Menge aller Router ist, dann lässt sich der Dijkstra-Algorithmus zum Ermitteln des kostengünstigsten Pfads wie folgt darstellen.





Distanzvektor-Routingalgorithmus

Im Gegensatz zum Link-State-Algorithmus übertragen die Router die Routinginformationen beim Distanzvektoralgorithmus nicht per Broadcast an alle anderen Router. Stattdessen geben die einzelnen Router die Routinginformationen lediglich an die benachbarten Router weiter. Die Routinginformationen umfassen die Anzahl von Verbindungen sowie die jeweiligen Verbindungskosten. Das Distanzvektorrouting ist ein iterativer, asynchroner und verteilter Algorithmus. Der kostengünstigste Pfad zwischen 4 und < wird mithilfe der Bellman-Ford-Gleichung berechnet:

’4 < = 71n= c z, x + dx y

Bei einer Änderung der Topologie oder Verbindungskosten aktualisiert jeder Router seine Tabelle selbst. Anschließend leitet er die aktualisierten Informationen an seine Nachbarn weiter. Die Nachbarn aktualisieren ihre Tabellen ebenfalls basierend auf den neuen Informationen. Wenn eine Tabellenänderung festgestellt wird, werden die Informationen an die jeweiligen Nachbarn weitergeleitet. Dieser Vorgang wird so lange fortgesetzt, bis alle Router über stabile Routingtabellen verfügen (Kurose & Keith, 2017).

Kommunikationsprotokolle

Routingprotokolle

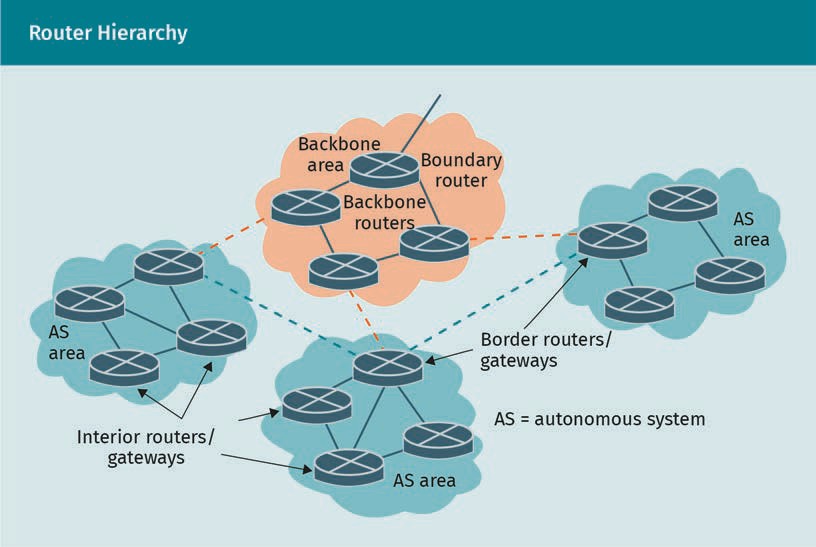
Routingprotokolle definieren die Nachrichtenübertragung und den Informationsaustausch zwischen Routern, um den Routingalgorithmus anzuwenden. Die Router innerhalb eines autonomen Systems (AS) oder Bereichs werden als Routingdomäne bezeichnet. Das zum Austausch von Informationen zwischen den Routern innerhalb einer Domäne verwendete Routingprotokoll heißt Intradomain- oder Intra-AS-Routingprotokoll. Eine weitere Bezeichnung ist Interior Gateway Protocol (IGP).

Jede Domäne verfügt über ein einziges IGP und verschiedene Domänen können unterschiedliche IGPs verwenden. Ein Router (oder Gateway), der die Intradomain-Router mit einer anderen Domäne verbindet, heißt Grenzrouter. Für den Austausch von Informationen zwischen Grenzroutern wird das sogenannte Interdomain-Routingprotokoll oder Exterior Gateway Protocol (EGP) verwendet (Kurose & Keith, 2017). Es gibt verschiedene Arten von Routingprotokollen. Nachfolgend sind eine Reihe von häufig verwendeten Routingprotokollen beschrieben.

OSPF

Open Shortest Path Frst (OSPF) ist ein IGP bzw. Intradomain-Routingprotokoll, mit dem der kürzeste Pfad zwischen Sender und Empfänger ermittelt wird. Dieses Protokoll basiert auf dem Dijkstra-Algorithmus, also auf einem Link-State-Routingalgorithmus. Jeder Router verwaltet eine eigene Routingdatenbank und Routingtabelle. OSPF ist ein dynamischer Algorithmus. Sobald Topologieänderungen erkannt werden, wird die Routingtabelle umgehend neu berechnet und die Routinginformationen werden per Broadcast an andere Router übertragen. Darüber hinaus ermöglicht OSPF eine Nachrichtenauthentifizierung, die Bestimmung mehrerer Pfade mit identischen Kosten und ein Multicasting. OSPF ist in RFC 2328, 5340, 5709, 7474, 6549 und 6860 standardisiert (Kurose &

Keith, 2017).



RIP

Routing Information Protocol (RIP) ist ein traditionelles Interior Gateway Protocol. Dieses Protokoll basiert auf dem Distanzvektor-Routingalgorithmus, der die Pfadkosten mithilfe der Bellman-Ford-Gleichung berechnet. RIP kann für Pfade mit maximal 15 Hops verwendet werden. Eine Anzahl von 16 Hops zeigt an, dass der Zielrouter nicht erreichbar ist. Aufgrund seiner Einschränkungen wird RIP heute selten genutzt. Durch die geringere Anzahl von Overhead-Bits kann das Protokoll für kleinere Netzwerke jedoch nützlich sein. Darüber hinaus unterstützt RIP die kryptografische Authentifizierung. RIP ist in RFC 2453 und 4822 standardisiert (Kurose & Keith, 2017).

BGP

Border Gateway Protocol (BGP) ist ein EGP bzw. Interdomain-Routingprotokoll. BGP ermittelt eine geeignete Route zwischen zwei Knoten, die sich in unterschiedlichen autonomen Systemen oder Domänen befinden. Der Austausch von Informationen erfolgt basierend auf dem TCP-Protokoll der Transportschicht über Port 179. Für BGP sind zwei Arten von TCP-Verbindungen möglich: External BGP (EBGP) und Internal BGP (IBGP). Bei EBGP befinden sich die kommunizierenden Router in zwei unterschiedlichen Domänen. IBGP wird für Verbindungen zwischen Routern verwendet, die sich innerhalb derselben Domäne befinden. Grenzrouter nutzen EBGP für den Austausch von Informationen mit anderen Grenzroutern in anderen Domänen. Für den Informationsaustausch mit Routern innerhalb derselben Domäne verwenden Grenzrouter IBGP. BGP ist in RFC 4271, 6286, 6608, 6793, 7606, 7607, 7705, 8212 und 8654 dokumentiert (Kurose & Keith, 2017).

ICMP

Internet Control Message Protocol (ICMP) ist in der Vermittlungsschicht eines Routers implementiert, um Steuerungsinformationen auszutauschen. Für ICMP werden IP-Header verwendet. Das Datagramm beinhaltet jedoch nicht die Payload der Transportschicht. Stattdessen ist die Steuerungsnachricht im IP-Header gekapselt. Das erste Oktett des Headers ist das Feld (field), das zweite Oktett der Code (code). Durch die Kombination des Feld- und Codewerts ergibt sich eine bestimmte Steuerungsnachricht oder Fehlerbenachrichtigung. Beispiel: Die Kombination „ﬁeld = 3, code = 1“ gibt an, dass der Host nicht erreichbar ist. Bei der Ausgabe von „ﬁeld = 3, code = 3“ ist der Port nicht erreichbar. ICMP ist in RFC 792 und 6918 standardisiert (Kurose & Keith, 2017).

###### Software Deﬁned Network

Ein softwaredefiniertes Netzwerk oder Software Deﬁned Network (SDN) ist eine Technologie zur Netzwerkverwaltung, bei der das Routing von einer zentralen Entität übernommen wird. Bei dieser zentralen Entität handelt es sich genau genommen um ein verteiltes System aus mehreren Servern. Da ein SDN in erster Linie auf IPv6 basiert, ist eine datenflussbasierte Weiterleitung möglich. Ein gängiges Protokoll für SDN ist OpenFlow (Benzekki et al., 2016; Kurose & Keith, 2017). Als Plattform für die SDN-Technologie ist beispielsweise Open Network Operating System (ONOS) verfügbar (Berde et al., 2014).

Übung

Laden Sie auf der Wireshark-Website Beispielpakete (.pcap- oder .cap-Dateien) für ICMP, OSPF und RIP herunter (Wireshark Foundation, ohne Jahresangabe). Untersuchen Sie die Headerfelder und Schichtadressen der Pakete. Überprüfen Sie außerdem, welche Arten von Nachrichten für ein bestimmtes Protokoll ausgetauscht werden, in welcher Reihenfolge der Austausch erfolgt und welche Nachrichtenformate verwendet werden.

Kommunikationsprotokolle

### Die Transportschicht: Zuverlässigkeit und Flusskontrolle

Die zur Implementierung der Transportschichtdienste verwendete Software wird als Transportschichtentität bezeichnet. Bei Transportschichtentitäten handelt es sich primär um Systemprogramme, die im Betriebssystemkernel implementiert sind. Darüber hinaus kann eine Transportschichtentität Teil eines Anwendungsprogramms sein. Über die Transportschicht wird eine logische Verbindung zwischen den sendenden und empfangenden Anwendungsprogrammen hergestellt. Außerdem ist diese Schicht für zuverlässige Datenübertragungsdienste zwischen Sender und Empfänger verantwortlich. Diese zuverlässigen Dienste umfassen Flusskontrolle, Überlastungskontrolle, Multiplexing, Übermittlung von Paketen in der richtigen Reihenfolge, Ermittlung von Paketverlust und erneute Übermittlung sowie das Herstellen und Beenden von Verbindungen.

Auf Senderseite empfängt die Transportschicht eine Nachricht von der Softwareanwendung und unterteilt die Nachricht in Segmente gleicher Größe. Zur Identifizierung der Sender- und Empfängeranwendungen werden anschließend zu jedem Segmentheader die Portnummern der Sender- und Empfängeranwendungen hinzugefügt. Der Header beinhaltet zudem die Dienstbits. Im nächsten Schritt wird das Segment an die Vermittlungsschicht weitergeleitet. Auf Empfängerseite empfängt die Transportschicht Segmente von der Vermittlungsschicht und liest die Portnummer aus dem Segmentheader, um das Zielanwendungsprogramm zu bestimmen. Die nachfolgenden Segmente werden dann zusammengeführt und an die Anwendungsschicht weitergegeben.

Bei den Diensten der Transportschicht wird zwischen einer verbindungsorientierten und verbindungslosen Kommunikation unterschieden. Das bekannteste verbindungsorientierte Protokoll ist TCP (Transmission Control Protocol). TCP bietet zuverlässige Datenübertragungsdienste. Das am häufigsten genutzte verbindungslose Protokoll ist UDP (User Datagram Protocol). UDP arbeitet nach dem Best-Effort-Prinzip, eine zuverlässige Datenübertragung ist also nicht sichergestellt. Weitere gängige Protokolle der Transportschicht sind Stream Control Transmission Protocol (SCTP), Resource Reservation Protocol (RSVP), Datagram Congestion Control Protocol (DCCP) und QUIC (vom englischen Wort „Quick“) (Papastergiou et al., 2016; Polese et al., 2019).

###### User Datagram Protocol (UDP)

UDP ist ein verbindungsloses Protokoll der Transportschicht, das keine zuverlässigen Datenübertragungsdienste gewährleistet. Dies bedeutet, dass keine Dienste für eine Paketübermittlung in der richtigen Reihenfolge, Flusskontrolle, Überlastungskontrolle, Fehlererkennung, Paketverlusterkennung oder erneute Übermittlung bereitgestellt werden. Für bestimmte Anwendungen, bei denen keine zuverlässige Datenübertragung zugesichert oder unbedingt erforderlich ist, ist UDP ein nützliches Protokoll. Dazu zählen Anwendungsfälle wie Echtzeitdienste, VoIP (Voice over Internet Protocol), Datenstreaming, Multicasting und Broadcasting. Transaktionsbasierte Protokolle oder Anfrage-Antwort-Protokolle wie DNS (Domain Name System) oder DHCP (Dynamic Host Configuration Protocol) nutzen ebenfalls UDP.

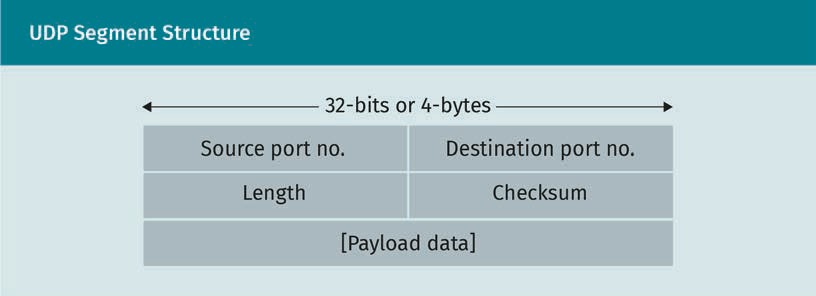
UDP-Segmentstruktur

UDP ist in RFC 768 standardisiert (Postel, 1980). UDP-Datagramme weisen eine einfache Struktur auf. Der Header ist 8 Bytes lang und umfasst die folgenden Felder:

Datenfluss

Als Datenfluss bezeichnet man einen Datenstrom aus Paketen, die zwischen der Quelle und dem Ziel übertragen werden und sich durch verschiedene Attribute bzw. eine Kombination aus Attributen wie IP-Adresse, MAC-Adresse und Portnummer von anderen Paketen unterscheiden.

* Der „Quellport“ umfasst eine 16-Bit-Portnummer der sendenden Anwendung. Dieses Feld ist optional.
* Der „Zielport“ umfasst die 16-Bit-Zielportadresse.
* „Länge“ ist die Größe des Datagramms in Bytes (einschließlich Header- und Payloadbits). Der Mindestwert des Felds „Länge“ könnte 8 Bytes sein. In diesem Fall enthält das Datagramm lediglich die Headerbits.
* „Prüfsumme“ ist eine 16-Bit-Zahl für die Fehlererkennung. Diese wird auch als Internetprüfsumme bezeichnet.



Internetprüfsumme

Zum Berechnen der Prüfsumme werden die Daten in 16-Bit-Blöcke unterteilt. Wenn die Anzahl der Bits kein Vielfaches von 16 ist, werden am Anfang der Datenbits zusätzliche Nullen hinzugefügt. Dies wird als „Zero Padding“, also das Auffüllen mit Nullen bezeichnet. Anschließend werden die 16-Bit-Blöcke addiert. Wenn dadurch ein Übertragsbit entsteht, wird dieses zur Summe addiert. Das Einerkomplement des Ergebnisses liefert schließlich die Internetprüfsumme. Sehen wir uns als Beispiel die folgenden 4-Byte-Daten an:

10001001 11000000 01101100 01111000

Diese 4 Byte großen Daten müssen zunächst in 16-Bit-Wörter unterteilt werden. Diese 16-Bit-Wörter werden addiert, um die Summe 11110110 00111000 zu erhalten. Bildet man das Einerkomplement der Summe, erhält man die Prüfsumme 00001001 11000111.

Beim Berechnen der Prüfsumme eines Segments werden sowohl die Header- als auch die Payloadbits berücksichtigt. Die Werte des Prüfsummenfelds werden jedoch auf Null gesetzt. Beim Empfang des Segments berechnet die Empfängerseite die Prüfsumme des Segments. Stimmt die berechnete Prüfsumme mit dem Wert des Prüfsummenfelds überein, liegt kein Bitfehler vor.

###### Transmission Control Protocol (TCP)

TCP ist ein verbindungsorientiertes Protokoll für eine zuverlässige Datenübertragung, das in RFC 793, 879, 6528 und 6691 standardisiert ist (Information Sciences Institute, 1981). Dieses Protokoll wird als „verbindungsorientiert“ bezeichnet, weil es vor dem Übertragen von Daten oder Segmenten eine Verbindung zwischen Sender und Empfänger herstellt. Nach der Übertragung wird die Verbindung von

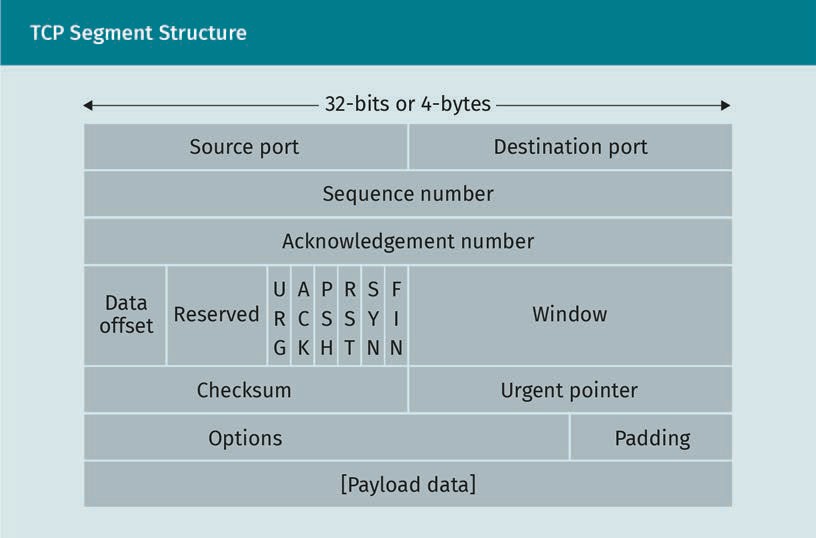
Kommunikationsprotokolle

beiden Parteien geschlossen. Die maximale Größe eines TCP-Segments (Maximum Segment Size, MSS) beträgt standardmäßig 536 Oktette. TCP wird als zuverlässig eingestuft, weil Dienste für eine fehlerfreie Übertragung von Daten in der richtigen Reihenfolge bereitgestellt werden. Diese Dienste umfassen Fehlererkennung, Verlusterkennung, erneute Übermittlung, Übertragung in der richtigen Reihenfolge, Flusskontrolle und Überlastungskontrolle.

TCP-Segmentstruktur

Die TCP-Segmentstruktur ist in RFC 793 definiert (Information Sciences Institute, 1981). Der Segmentheader ist üblicherweise 40 Bytes lang und umfasst die folgenden Felder:

* Das Feld „Quellport“ enthält eine 16-Bit-Portadresse der Senderanwendung.
* Das Feld „Zielport“ enthält eine 16-Bit-Portadresse der Empfängeranwendung.
* Das Feld „Sequenznummer“ enthält eine 32-Bit-Zahl. Dies ist die Sequenznummer des ersten Datenoktetts des Segments. Wenn das SYN-Bit 1 ist, entspricht dieser Feldwert dem ISN-Wert (Initial Sequence Number), und ISN+1 gibt das erste Datenoktett an.
* Das Feld „Bestätigungsnummer“ enthält eine 32-Bit-Zahl. Dieser Feldwert entspricht der Sequenznummer des Segments, das vom Sender als nächstes empfangen werden sollte.
* Das Feld „Datenoffset“ gibt an, wo die Payloaddaten innerhalb des Segments beginnen.
* Das Feld „Reserviert“ ist 6 Bits groß und für die zukünftige Verwendung reserviert.
* Das Feld mit den Steuerungsflags enthält sechs 1-Bit-Flags. Das URG-Flag zeigt dringende Daten an, ACK steht für eine Bestätigung, PSH bedeutet, dass das Segment umgehend an die nächste Schicht weitergegeben werden soll, RST gibt an, dass die Verbindung zurückgesetzt werden soll, SYN steht für eine Verbindungsanforderung und FIN bedeutet, dass die Datenübertragung beendet werden soll.
* Das 16-Bit-Feld „Fenster“ zeigt die Größe des Empfangspuffers an, also die Anzahl an Bytes, die vom Sender des Segments erwartet werden.
* Das Feld „Prüfsumme“ enthält eine 16-Bit-Internetprüfsumme.
* Wenn im Feld „Dringlichkeitsanzeiger“ der URG-Wert auf 1 gesetzt ist, gibt dieser 16-Bit-Wert die Sequenznummer des Oktetts an, das sich neben den dringenden Daten befindet.
* Das Feld „Optionen“ kann den MSS-Wert für die maximale Segmentgröße enthalten. Ist dieses Feld nicht vorhanden, können Segmente eine beliebige Größe aufweisen.
* Das Feld „Füllbits“ gibt die Anzahl an Zero Padding-Bits an, die hinzugefügt werden, damit die Segmentgröße ein Vielfaches von 32 ist.



Herstellen und Beenden einer TCP-Verbindung

In einigen Kulturen begrüßen sich zwei Parteien zunächst per Handschlag, bevor sie geschäftliche Dinge besprechen. Und nach Abschluss ihrer Besprechungen verabschieden sich die Parteien voneinander. Auf ähnliche Weise werden bei TCP vor der eigentlichen Datenübertragung zunächst Nachrichten zum Herstellen einer Verbindung zwischen Sender und Empfänger ausgetauscht. Nach der erfolgreichen Datenübertragung tauschen die beiden Parteien erneut Nachrichten aus, um die Kommunikation zu beenden.

Das Softwareprogramm, das die Verbindung initiiert, wird als Clientprozess bezeichnet. Die Gegenseite, die die Verbindungseinladung annimmt, ist der Hostprozess. Der Schritt zum Herstellen der Verbindung heißt Handshaking. Bei TCP werden drei Schritte ausgeführt, um eine Verbindung aufzubauen. Aus diesem Grund wird das Verfahren auch Drei-Wege-Handshake genannt.

* + 1. Der Client sendet einen TCP-Header mit )> 2 = 1, A6K = 0 und )e/Aence 2B . = 0 (oder eine beliebige Zahl).
    2. Der Host antwortet mit einem TCP-Header, bei dem )> 2 = 1, A6K = 1,

AcknBDle’ge7ent 2B . = 1 ist (die empfangene Sequenz ist also 2B . + 1) und

)e/Aence 2B . = 0.

* + 1. Der Client antwortet mit einem TCP-Header he8’e, D1th )> 2 = 0, A6K = 1,

)e/Aence 2B . = 1 und AcknBDle’ge7ent 2B . = 1.

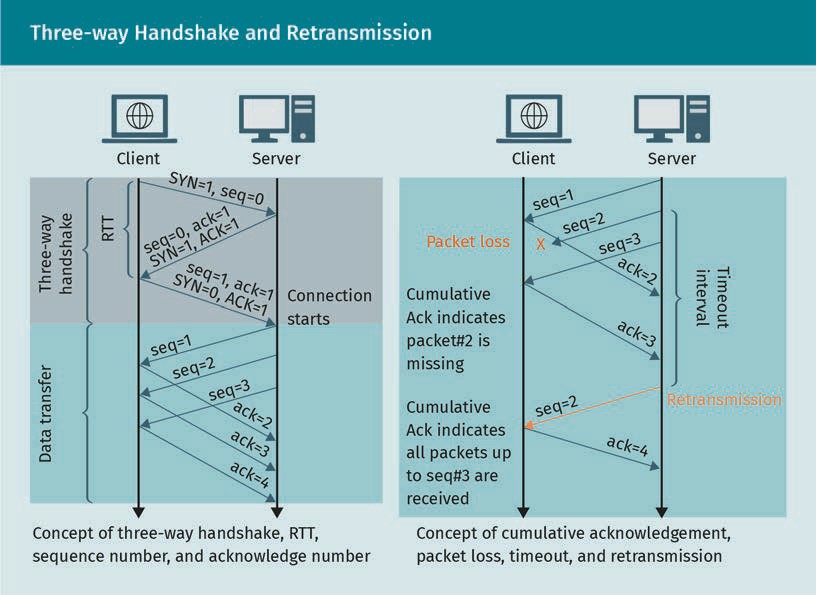
Zum Beenden der Verbindung werden die folgenden vier Schritte ausgeführt:

1. Der Client sendet GI2 = 1 und )e/Aence 2B . = X. Dies bedeutet, dass der Client den Vorgang zum Senden von Daten abgeschlossen hat, jedoch weiterhin Daten empfangen kann.
2. Der Host sendet A6K = 1 und AcknBDle’ge7ent 2B . = X + 1.

Kommunikationsprotokolle

1. Der Host sendet GI2 = 1 und )e/Aence 2B . = > . Der Host beendet den Vorgang zum Senden von Daten.
2. Der Client sendet A6K = 1 und AcknBDle’ge7ent 2B . = > + 1. Der Host schließt die Verbindung, nachdem er diese Bestätigung empfangen hat. Der Client schließt die Verbindung nicht umgehend, sondern wartet die zweifache Dauer des Timeouts, bevor er die Verbindung beendet.

Das Prinzip des Drei-Wege-Handshake ist in der nachfolgenden Abbildung „Drei-Wege-Handshake und erneute Übertragung“ gezeigt.



Verlusterkennung und Wiederherstellung

Nachdem der Client ein Paket mit )e/Aence 2B . = X gesendet hat, wartet er, bis er die Antwort des Hosts mit AcknBDle’ge7ent 2B . = X + 1 empfängt. Die maximale Wartezeit ist durch das Timeout-Intervall festgelegt. Wenn der Client bis zum Ablauf dieser Wartezeit keine Antwort empfangen hat, wird das Paket als verloren betrachtet und der Client übermittelt das Paket erneut. Das Timeout-Intervall für die erneute Übertragung wird mithilfe der folgenden Formel berechnet.

t17eBAt = est178te’RTT + 4 · ’e0RT T

est178te’RTT = 1 − K × est178te’RTT + K · s87pleRT T ’e0RT T = 1 − /3 × ’e0RTT + /3 est178te’RTT − s87pleRTT

Die Paketumlaufzeit bzw. Round Trip Time (RTT) ist das Zeitintervall zwischen dem Senden eines Pakets und dem Empfang der entsprechenden Bestätigung. Die Round Trip Time wird auch als Round Trip Delay, also Pakettransportverzögerung bezeichnet. Die RTT des zuletzt empfangenen Pakets heißt sampleRTT.

Flusskontrolle

Ein empfangenes Paket verbleibt in einem Empfangspuffer, bis das vorherige Paket verarbeitet wurde. Wenn Pakete nicht in der richtigen Reihenfolge empfangen werden, werden sie ebenfalls im Puffer platziert, bis das fehlende Paket empfangen wurde. Wenn die Senderseite Pakete so schnell überträgt, dass die Verarbeitungsgeschwindigkeit des Empfängers nicht ausreicht, um die Pakete rechtzeitig zu verarbeiten, kommt es zu einem Überlauf des Empfangspuffers. Weitere Pakete, die nach diesem Überlauf empfangen werden, gehen verloren oder werden verworfen. Durch die TCP-Flusskontrolle wird die Übertragungsrate des Senders so eingeschränkt, dass der Empfangspuffer nicht überlaufen kann. Dazu wird das Feld „Fenster“ des TCP-Headers auf die Größe des freien Puffers festgelegt. Dieses Fenster heißt auch Empfangsfenster. Wenn ein Client das Empfangsfenster auf 1.000 Bytes festlegt, sendet der Server eine maximale Datenmenge von 1.000 Bytes ohne Bestätigung. Angenommen, ein Server hat fünf Pakete mit je 1.000 Bytes gleichzeitig gesendet. Der Server muss die Übertragung nun unterbrechen und warten, bis er alle Bestätigungen für diese 5.000 Bytes erhalten hat. Nach dem Empfang der Bestätigungen kann der Server die nächsten 5.000 Bytes senden.

Überlastungskontrolle

Client und Server können über eine Vielzahl von Verbindungen und Routern miteinander verbunden sein. Da ein Router im Allgemeinen über mehrere Verbindungen verfügt, kann er Daten von mehreren Quellen gleichzeitig empfangen. Wenn die Datenempfangsrate eines Routers höher ist als seine Dienstrate (also die Verarbeitungskapazität), kann es zu einem Pufferüberlauf kommen, sodass Pakete verworfen werden oder verloren gehen. Dieses Phänomen der Übertragung zu großer Datenmengen über eine Netzwerkverbindung wird als Netzwerküberlastung bezeichnet. Bei TCP wird eine Netzwerküberlastung auf zwei Arten ermittelt: über das Retransmission Timeout und anhand von drei doppelten Bestätigungen. Sobald eine Überlastung ermittelt wird, wird das Überlastungsfenster verringert. Die beiden gängigsten Protokolle zur Überlastungskontrolle sind TCP Tahoe und TCP Reno. Es sind jedoch mehr als dreißig Protokollvarianten für die TCP-Überlastungskontrolle verfügbar. Allerdings werden in der Praxis nur sehr wenige verwendet. Bevor auf TCP Tahoe und TCP Reno eingegangen wird, sollen zunächst die folgenden Konzepte erläutert werden (Abed et al., 2011):

* + Das „Überlastungsfenster“ ist die Anzahl an Oktetten oder Bytes, die ein Sender nacheinander sendet, ohne eine Bestätigung zu empfangen.
  + Bei TCP werden sogenannte „kumulative Bestätigungen“ verwendet. Die Bestätigung von n zeigt an, dass der Empfänger alle Pakete bis n − 1 empfangen hat und auf den Empfang von Paket n wartet.
  + „Doppelte Bestätigung“ bedeutet, dass beim Empfang von Paket n + 1 vor Paket n (Übermittlung in der falschen Reihenfolge) der Empfänger die Bestätigung n erneut sendet. Der Sender empfängt also eine doppelte Bestätigung von n. Wenn das Paket n + 2 ebenfalls mit der falschen Reihenfolge vor n empfangen wird, sendet der Empfänger die Bestätigung n ein weiteres Mal. Wird auch das Paket n + 3 in der falschen Reihenfolge empfangen, erhält der Sender eine dritte doppelte Bestätigung für n.
  + „Fast Retransmit“, also eine schnelle Neuübertragung, bedeutet, dass der Sender beim Empfang einer dritten doppelten Bestätigung das Paket n umgehend erneut übermittelt, ohne die Timeout-Dauer abzuwarten. Dieser Mechanismus weist auf eine geringe Überlastung innerhalb des Netzwerks hin.
  + Ein weiteres Feature von IP- und TCP-Protokollen ist „Explicit Congestion Notification“ (ECN). Diese Erweiterung ermöglicht eine End-to-End-Benachrichtigung in Überlastungssituationen.

Kommunikationsprotokolle

TCP Tahoe und TCP Reno

TCP Tahoe und TCP Reno bieten zwei Methoden, um Überlastungssituationen zu verhindern: Congestion Avoidance (Vermeiden einer Überlastung) und Fast Recovery (schnelle Wiederherstellung der Sendeleistung). Die Congestion Avoidance-Phase ist bei beiden Algorithmen identisch, in der Fast Recovery-Phase unterscheiden sie sich jedoch.

Congestion Avoidance

Diese Phase wird auch als langsamer Start (Slow Start) bezeichnet. Anfänglich ist für die Größe des Überlastungsfensters 6W2( eine maximale Segmentgröße (Maximum Segment Size, MSS) von 1 festgelegt. Für jede erfolgreiche Übermittlung wird 6W2( zum Quadrat erhoben. Nach der ersten erfolgreichen Übertragung lautet der Wert also 6W2( = 2, nach einer zweiten erfolgreichen Übertragung 6W2( = 4 usw. Sobald 6W2( jedoch den vordefinierten Schwellenwert der Segmentgröße erreicht, wird 6W2( nicht mehr exponentiell, sondern linear erhöht. Das heißt, dass 6W2( nach jeder erfolgreichen Übermittlung um 1 MSS erhöht wird.

Fast Recovery

Sobald eine erneute Übermittlung ausgelöst wird, startet TCP die Fast Recovery-Phase. Außerdem beginnt erneut die Congestion Avoidance-Phase. Dabei werden ssth,eshBl’ = 6W2(/2 und 6W2( = 1 festgelegt. Wird die erneute Übermittlung aufgrund einer dritten doppelten Bestätigung ausgelöst wird, reagiert TCP Reno jedoch anders. In diesem Fall werden ssth,eshBl’ = 6W2(/2 und 6W2( = 6W2( − 3 festgelegt.

Übung

Laden Sie die Beispielpakete von der Wireshark-Website herunter (Wireshark Foundation, ohne Jahresangabe). Wenden Sie das Gelernte an, um die TCP- und UDP-Headerstruktur und -Informationen zu untersuchen.

Zusammenfassung

Explicit Congestion Notiﬁcation (ECN) Bei dieser IP-Erweiterung werden im IP-Header zwei ECN-Bits angegeben, um eine Überlastung erkennen zu können, ohne dass Pakete verworfen werden. Wenn es bei einem Router zu einer Überlastung kommt, wenn also aufgrund eines Pufferüberlaufs Pakete verworfen werden, setzt der Router das ECN-Bit des Headers, um die Endknoten über die Überlastung zu informieren.

In Computernetzwerken sind sieben Schichten implementiert: Anwendungs-, Darstellungs-, Sitzungs-, Transport-, Vermittlungs-, Sicherungs- und Bitübertragungsschicht. Die bereitgestellten Dienste und die Adressierung unterscheiden sich in den verschiedenen Schichten. Portnummer, IP-Adresse und MAC-Adresse sind die Adressen der Transport-, Vermittlungs- bzw. Sicherungsschicht. Über die Anwendungsschicht wird eine Anwendungsschnittstelle für Benutzer bereitgestellt. In der Darstellungsschicht werden Daten der Anwendungsschicht zu Komprimierungs- und Verschlüsselungszwecken codiert. Die Sitzungsschicht ist für die Sitzungsverbindung zwischen Sender und Empfänger verantwortlich und über die Transportschicht werden Dienste für eine zuverlässige Datenübertragung, Fehlererkennung, Flusskontrolle und Überlastungskontrolle bereitgestellt. Die beiden gängigsten Protokolle der Transportschicht sind TCP und UDP. Während TCP ein verbindungsorientiertes, zuverlässiges Protokoll ist, handelt es sich bei UDP um ein unzuverlässiges, verbindungsloses Protokoll. Das bedeutet, dass UDP keine Dienste für Flusskontrolle, Überlastungskontrolle, Erkennung von Paketverlust, erneute Übermittlung oder Übertragung in der richtigen Reihenfolge bietet. Die beiden wichtigsten Dienste dieser Schicht sind Paketweiterleitung und Routing. Zwei weit verbreitete Routingalgorithmen sind der Link-State- und der Distanzvektoralgorithmus. Das Link-State-Routing basiert auf dem Dijkstra-Algorithmus zur Bestimmung des kostengünstigsten Pfads, das Distanzvektorrouting auf der Bellman-Ford-Gleichung. Weitere gängige Protokolle der Netzwerkschicht sind OSPF, RIP und BGP. OSPF basiert auf dem Link-State-Routing, RIP auf Distanzvektor-Routingalgorithmen. Beide Protokolle sind

in erster Linie Intradomain-Routingprotokolle. BGP ist ein Interdomain-Protokoll. ICMP wird zum Austausch von Steuerungsnachrichten zwischen Routern eingesetzt.



# Lektion 3

## Die Internetprotokollfamilie

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… die Geschichte des Internets und des World Wide Web wiederzugeben.

… die Konzepte des TCP/IP-Referenzmodells (Transmission Control Protocol/Internet Protocol) sowie des Protokollstapels zu erläutern.

… Ihr Wissen zu Internetprotokollen und -diensten für die Analyse von Datenpaketen anzuwenden.

… die Sicherheitsaspekte in Computernetzwerken und im Internet zu analysieren.

DL-E-DLMCSNDS01-U03

1. Die Internetprotokollfamilie

### Einführung

Das Internet ist das am meisten genutzte Computernetzwerk unserer Zeit. Mit Blick auf die Gesamtzahl der verbundenen Geräte, die Anzahl an Benutzern, die räumliche Ausbreitung und den organisatorischen Geltungsbereich ist es zudem das größte Computernetzwerk weltweit. Der Begriff „Internet“ setzt sich aus den englischen Wörtern „inter“ und „network“ zusammen und weist darauf hin, dass es sich um eine Vielzahl von miteinander verbundenen Netzwerken handelt (Kurose & Keith, 2017). Das Internet umfasst praktisch jede Art von Netzwerk: von Netzwerken mit unterschiedlicher räumlicher Ausdehnung (WAN, MAN, LAN oder PAN) oder verschiedenen organisatorischen Geltungsbereichen (Intranet oder Extranet) bis hin zu den unterschiedlichsten Netzwerktopologien und Netzwerkgeräten. Die bereitgestellten Netzwerkdienste reichen vom World Wide Web (WWW) über E-Mail- und Dateifreigabedienste bis hin zum Streaming von Audio- und Videoinhalten. Anstelle des ISO/OSI-Modells der Internationalen Organisation für Normung (International Organization for Standardization) wird im Internet das TCP/IP-Modell (Transmission Control Protocol/Internet Protocol) als Grundlage für die Kommunikation zwischen verbundenen Geräten verwendet. In dieser Lektion wird die Geschichte des Internets und des World Wide Web beschrieben. Außerdem werden das Konzept, die Protokolle und die Dienste des TCP/IP-Protokollstapels erläutert und es wird auf die Sicherheitsrisiken bei der Kommunikation im Internet eingegangen.

### Geschichte des Internets und World Wide Web

Die Entstehung des Internets begann in den frühen 1960er Jahren mit der Erfindung eines Netzwerks für die Paketvermittlung (Cohen-Almagor, 2013). Dieses Netzwerk wurde kontinuierlich weiterentwickelt und in den 1980er Jahren wurde der Protokollstandard TCP/IP eingeführt. Die Ausbreitung des modernen Internets begann nach der Erfindung des World Wide Web (WWW) in den 1990er Jahren (Forouzan, 2013).

###### 1960-1970: Entwicklung der Paketvermittlung (ARPANET)

Leitungsvermittlung Bei der leitungsvermittelnden Technologie sind die kommunizierenden Knoten über eine dedizierte Verbindung miteinander verbunden (Comer, 2015).

Vor der Entstehung des Internets galten Telegrafen- und Telefonnetze als hochmoderne Technologien für die Kommunikation über große Entfernungen. Bei der Telegrafie wurden Textnachrichten ausgetauscht, das Telefon bot die Möglichkeit der Audiokommunikation. Grundlage dieser beiden Technologien waren leitungsvermittelnde Netzwerke, in denen Daten mit einer konstanten Rate ausgetauscht wurden. 1961 entwickelte das Massachusetts Institute of Technology (MIT) das Konzept der Paketvermittlung, bei dem Daten mit einer variablen Bitrate übertragen werden konnten (Cohen-Almagor, 2013). Diese Technologie wurde 1964 vom Rand Institute für eine sichere Audioübertragung in Militärnetzwerken implementiert.

1965 wurde das Konzept des „Pakets“ vom britischen National Physical Laboratory (NPL) eingeführt (Cohen-Almagor, 2013). Die Erfindung des Pakets und der Paketvermittlung schufen die Grundlage für die Entstehung des Internets. Das erste paketvermittelte Netzwerk wurde 1967 von der Advanced Research Projects Agency (ARPA) des US-amerikanischen Verteidigungsministeriums implementiert, um ein kleines Computernetzwerk zu erstellen, das sogenannte ARPANET (Advanced Research Projects Agency Network) (Kurose & Keith, 2017; Leiner et al., 2009). Hintergrund des ARPANET war der Wunsch, Computer unterschiedlicher Hersteller

Die Internetprotokollfamilie

mit einem spezialisierten Computer, dem Interface Message Processor (IMP), zu verbinden. Im Jahr 1969 konnten im ARPANET vier Knoten über Interface Message Processors (IMPs) verbunden werden, die sich an unterschiedlichen Universitätsstandorten befanden: in der University of California in Los Angeles, der University of California in Santa Barbara, im Stanford Research Institute und in der University of Utah (Forouzan, 2013; Leiner et al., 2009).

###### 1970-1980: Entwicklung von TCP (ALOHANet, PRNET und SATNET)

1972 konnten über das ARPANET fünfzehn Knoten verbunden werden. Für die Kommunikation zwischen den Endknoten wurde zunächst das Network Control Protocol (NCP) verwendet. Anfangs war das ARPANET ein einzelnes Netzwerk. Später, im Jahr 1972, wurde das ARPANET-Projekt zur Verknüpfung mehrerer Netzwerke gestartet, um Knoten aus unterschiedlichen Netzwerken mit verschiedenen Eigenschaften zu verbinden. Diese Netzwerke unterschieden sich beispielsweise durch variable Übertragungsraten oder Paketgrößen (Kurose & Keith, 2017; Cohen-Almagor, 2013). Bei diesem Projekt entstand die Idee eines Gateways, also eines Geräts, mit dem zwei unterschiedliche Netzwerke miteinander verbunden werden. Dieselbe Forschungsgruppe stellte 1973 das Konzept des Transmission Control Protocol (TCP) vor (Forouzan, 2013). Neben dem ARPANET wurden Mitte der 1970er Jahre weitere paketvermittelnde Netzwerke eingeführt. Dazu zählten z. B. das mikrowellenbasierte ALOHANet, das von der University of Hawaii entwickelt wurde, das Packet Radio Network (PRNET) der Defense Advanced Research Projects Agency (DARPA) und das Atlantic Packet Satellite Network (SATNET) von BBN Technologies und der Advanced Research Projects Agency (Forouzan, 2013; Leiner et al., 2009).

###### 1980-1990: Entwicklung von TCP/IP (BITNET, CSNET, MILNET, NSFNET und ANSNET)

Nachdem zusätzlich zu ARPANET weitere Technologien entwickelt wurden, wurde TCP in zwei Protokolle unterteilt: Transmission Control Protocol (TCP) und Internet Protocol (IP) (Forouzan, 2013). Die Kombination aus TCP und IP wurde als TCP/IP-Protokollstapel bezeichnet. IP wird für das Paketrouting eingesetzt, TCP für andere Dienste höherer Schichten. Dazu zählen beispielsweise Paketstrukturierung, Datensegmentierung und erneute Zusammensetzung von Datensegmenten sowie die Fehlererkennung. 1981 kombinierte die University of California (UC) in Berkeley TCP/IP mit einem UNIX-Betriebssystem. Im selben Jahr entwickelte die National Science Foundation (NSF) das Computer Science Network (CSNET), um Universitäten zu verbinden, die nicht mit dem US-amerikanischen Verteidigungsministerium zusammenarbeiteten und somit keinen Zugriff auf das ARPANET hatten (Forouzan, 2013; Kurose & Keith, 2017; Cohen, 2013). Ebenfalls im Jahr 1981 entwickelte Ira Fuchs das BITNET („Because It’s There Network“ oder „Because It’s Time Network“), um E-Mail- und Dateifreigabedienste für Universitäten bereitzustellen (Kurose & Keith, 2017; Leiner et al., 2009).

1983 wurde das Military Network (MILNET) für militärische Zwecke entwickelt, sodass das ARPANET für die nicht militärische Nutzung verwendet werden konnte. Im gleichen Jahr löste TCP/IP das zuvor verwendete NCP als neues offizielles Protokoll im ARPANET ab. Im Jahr 1986 entwickelte die NSF das NSFNET, um die Supercomputer-Center mit einer Übertragungsrate von 1,5 MBit/s zu verbinden (Kurose & Keith, 2017). 1990 wurde das ARPANET durch das NSFNET abgelöst. Aufgrund der enormen Datenverkehrsanforderungen wurde das NSFNET durch das Advanced Network Services Network (ANSNet) ersetzt, einem Produkt der gemeinnützigen Organisation Advanced Network Services (ANS), die von drei Unternehmen gegründet wurde: IBM, Merit und Verizon (Forouzan, 2013; Kurose & Keith, 2017).

Paketvermittlung

In einem paketvermittelnden Netzwerk gibt es keinen dedizierten Pfad zwischen Sender und Empfänger. Pakete aus einer Quelle können über verschiedene Pfade ans Ziel transportiert werden.

###### Die 1990er Jahre: Entwicklung des World Wide Web

In den 1990er Jahren wurden zahlreiche Internetanwendungen und -dienste entwickelt. Eine entscheidende Rolle bei dieser Entwicklung spielte die Erfindung des World Wide Web, das zwischen 1989 und 1991 in der Europäischen Organisation für Kernforschung (European Organization for Nuclear Research, CERN) entstand (Cohen-Almagor, 2013). Am CERN wurden vier grundlegende Elemente des WWW entwickelt: Hypertext Markup Language (HTML), Hypertext Transmission Protocol (HTTP), der Webbrowser und der Webserver. Bis zum Jahr 1993 waren weltweit etwa zweihundert Webserver im Einsatz (Forouzan, 2013; Cohen-Almagor, 2013).

###### Wer regiert das Internet? Standards und Administratoren

Das Internet wird nicht von einer einzelnen Behörde oder Organisation kontrolliert oder gesteuert. Es gibt jedoch verschiedene Gruppen und Organisationen, die die Infrastruktur, Standards, Protokolle und weitere relevante Aspekte im Internet steuern und regulieren. Nachfolgend sind einige dieser Organisationen aufgeführt:

* Die Internet Society (ISOC) ist der führende Verband für die Internetstandardisierung durch IAB, IETF, IRTF und IANA (Forouzan, 2013).
* Das Internet Architecture Board (IAB) unterstützt die ISOC beratend bei technischen Fragen. Das Komitee überwacht die Entwicklung der TCP/IP-Protokollfamilie und RFCs durch IETF und IRTF (Forouzan, 2013).
* Die Internet Engineering Task Force (IETF) ist für die Entwicklung und Vorbereitung von Entwürfen für Protokollstandards verantwortlich, den sogenannten RFCs (Request for Comments). Die IETF wird von der Internet Engineering Steering Group (IESG) geleitet (Forouzan, 2013, Kurose & Keith, 2017).
* Die Internet Research Task Force (IRTF) beschäftigt sich mit langfristigen Forschungsthemen in Bezug auf das Internet. Diese Arbeitsgruppe arbeitet unter Leitung der Internet Research Steering Group (IRSG) (Forouzan, 2013).
* Die Internet Assigned Numbers Authority (IANA) ist für die Zuordnung von IP-Adressen, DNS (Domain Name System) und Medientypen verantwortlich. Sie untersteht der Organisation Public Technical Identiﬁers (PTI), die mit der Internet Corporation for Assigned Names and Numbers (ICANN) verbunden ist (Forouzan, 2013, Kurose & Keith, 2017).

### TCP/IP-Referenzmodell und -Protokollstapel

Im Kontext von TCP/IP (Transmission Control Protocol/Internet Protocol) wird häufig auch vom TCP/IP-Protokollstapel oder der TCP/IP-Protokollfamilie gesprochen. Diese Protokollfamilie ist im Wesentlichen eine Sammlung standardisierter Protokolle, die in hierarchischen Schichten angeordnet sind. Dabei werden die Protokolle der oberen Schichten durch die Protokolle der unteren Schichten unterstützt. Darüber hinaus ist jedes Protokoll mit dem entsprechenden Protokoll der nächsthöheren und nächsttieferen Schicht kompatibel. TCP/IP ist die Standardprotokollfamilie für das Internet. Traditionell ist diese Protokollfamilie in vier Schichten untergliedert: Anwendung, Transport, Internet und Netzzugang. Moderne Internetprotokolle werden jedoch in fünf Schichten unterteilt: Anwendung, Transport, Internet, Sicherung und Bitübertragung (Forouzan, 2013). In den verschiedenen Schichten werden die übertragenen Daten

Die Internetprotokollfamilie

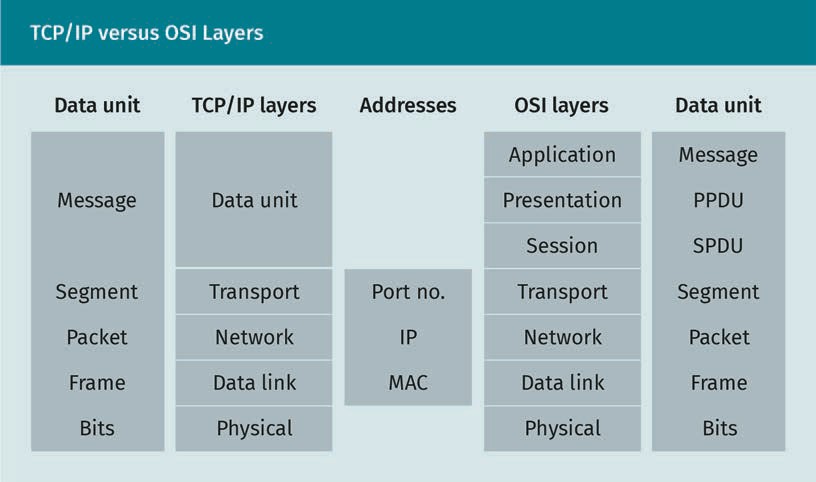
durch unterschiedliche Einheiten gekennzeichnet. Bei den ersten vier Schichten werden z. B. die Einheiten „Nachricht“, „Segment“, „Paket“ und „Frame“ verwendet.

###### TCP/IP-Schichten

Im Gegensatz zum OSI-Modell umfasst das TCP/IP-Modell keine Darstellungs- und Sitzungsschicht. Die Dienste dieser beiden Schichten sind in der Anwendungs- und Transportschicht vereint. Im Folgenden sind die einzelnen Schichten der TCP/IP-Protokollfamilie beschrieben.

Schicht 5: Anwendungsschicht

Die Anwendungsschicht ist die fünfte Schicht der TCP/IP-Protokollfamilie und stellt damit die oberste Schicht des Protokollstapels dar. Diese Schicht ist für die Kommunikation zwischen den Prozessen verantwortlich. Sie stellt logische Verbindungen zwischen den Anwendungsprogrammen oder Prozessen her, die auf Sender- oder Empfängerknoten ausgeführt werden (z. B. Browser, E-Mail-Manager oder Messenger). Die Anwendungsschicht ist nur auf Endknoten verfügbar. Auf Verbindungsknoten wie Routern, Switches, Bridges und Hubs ist sie nicht vorhanden. Die gesendeten oder empfangenen Daten dieser Schicht werden als Nachricht bezeichnet. Für verschiedene Internetdienste sind eine Vielzahl von Protokollen der Anwendungsschicht vorhanden. Zu den häufig verwendeten Protokollen der Anwendungsschicht zählen Hypertext Transfer Protocol (HTTP) für Webanwendungen, Simple Mail Transfer Protocol (SMTP) für die E-Mail-Weiterleitung, Post Ofﬁce Protocol (POP) und Internet Message Access Protocol (IMAP) für den E-Mail-Zugriff, Domain Name System (DNS) zur Ermittlung von Adressen der Internetschicht, File Transfer Protocol (FTP) für Dateifreigaben, Simple Network Management Protocol (SNMP) für die Netzwerkverwaltung, TELNET für den Remotezugriff sowie Secure Shell (SSH) für einen abgesicherten Remotezugriff. (Forouzan, 2013).



Schicht 4: Transportschicht

Die nächste Schicht ist die Transportschicht, die für logische Verbindungen zwischen End-to-End-Systemprozessen verantwortlich ist. Auch diese Schicht ist auf Verbindungsknoten nicht vorhanden. Auf Senderseite empfängt die Transportschicht die Nachricht der Anwendungsschicht und unterteilt diese Nachricht in kleinere Abschnitte identischer Größe, die mit Headerbits der Transportschicht gekapselt werden. Diese gekapselten Abschnitte heißen Segmente. Die Einheit der übertragenen Daten in der Transportschicht ist also das Segment. Die Headerbits enthalten die Quell- und Zieladressen der Transportschicht, die als Quell- und Zielportnummern bezeichnet werden. Anhand dieser Portnummern lässt sich das gewünschte Anwendungsprogramm auf einem Gerät identifizieren. Die Hauptaufgabe eines Transportschichtprotokolls ist eine zuverlässige Paketübermittlung. Zu diesem Zweck werden verschiedene Dienste eingesetzt, wie z. B. Fehlererkennung, Paketübermittlung in der richtigen Reihenfolge, Erkennung von Paketverlust und erneute Übermittlung sowie Überlastungserkennung und -kontrolle. Zu den wichtigsten Protokollen der Transportschicht zählen Transmission Control Protocol (TCP) und User Datagram Protocol (UDP). TCP wird für Dienste eingesetzt, die eine zuverlässige Datenübertragung erfordern (z. B. E-Mail- oder Dateiübertragungen), UDP hingegen wird für Dienste genutzt, bei denen keine zuverlässige Datenübertragung zugesichert wird (z. B. das Streamen von Videoinhalten oder Echtzeitdienste). Ein weiteres Protokoll der Transportschicht ist Stream Control Transmission Protocol (SCTP). Dieses Protokoll wird für Multimedia-Dienste verwendet (Forouzan, 2013; Kurose & Keith, 2017).

Schicht 3: Internetschicht

Die dritte Schicht des TCP/IP-Protokollstapels ist die Internetschicht. Diese Schicht ist für die Kommunikation von Systemprogrammen zwischen Sender und Empfänger verantwortlich. Segmente werden mit Headerbits der Internetschicht in Datagramme gekapselt, die als Pakete bezeichnet werden. Die Einheit der übertragenen Daten dieser Schicht ist also das Paket. Adressen der Internetschicht heißen IP-Adressen. Anhand dieser IP-Adressen lassen sich Geräte oder Betriebssysteme bzw. das Netzwerksystemprogramm identifizieren. Router werden als Geräte der Internetschicht betrachtet, da die höchste Schicht dieser Geräte die Internetschicht ist. Die beiden wichtigsten Dienste, die mit Protokollen dieser Schicht ausgeführt werden, sind Paketweiterleitung und Routing. Das wichtigste Protokoll ist IP. Es wird für die Paketformatierung und für das Routing eingesetzt. Weitere wichtige Protokolle der Internetschicht sind Internet Control Message Protocol (ICMP), Dynamic Host Conﬁguration Protocol (DHCP), Address Resolution Protocol (ARP) und Internet Group Management Protocol (IGMP) (Forouzan, 2013; Kurose & Keith, 2017).

Frame Frames sind die Einheit übertragener Daten in der

Sicherungsschicht.

Schicht 2: Sicherungsschicht

//Diese Schicht wird auch als Data Link Layer bezeichnet. Auf Senderseite werden in dieser Schicht Pakete der Internetschicht empfangen und mit Headerbits der Sicherungsschicht in einem Frame gekapselt. Diese Schicht ist für logische Verbindungen zwischen den Netzwerkkarten der Sender- und Empfängerknoten verantwortlich. Eine Netzwerkkarte (Network Interface Card, NIC) ist ein Hardwaregerät, mit dem ein Knoten mit dem Netzwerk oder Internet verbunden wird. Knoten können über verschiedene Methoden mit dem Internet verbunden werden (z. B. über WLAN, Twisted-Pair-Kabel-basiertes Ethernet, Glasfaserkabel, Satellitenfunk oder Mobilfunk). Wenn ein Knoten verschiedene Methoden für die Verbindung mit dem Netzwerk nutzt, ist für jede dieser Verbindungen eine separate NIC erforderlich. Die von Protokollen der Sicherungsschicht bereitgestellten Dienste umfassen Frameerstellung, Fehlererkennung, Medienzugriffssteuerung (Media Acces Control, MAC) und Flusskontrolle. Zu den wichtigsten Protokollen dieser Schicht zählen Ethernet, IEEE 802 und Point-to-Point Protocol (PPP) (Forouzan, 2013; Kurose & Keith, 2017).

Die Internetprotokollfamilie

Schicht 1: Bitübertragungsschicht

Die Bitübertragungsschicht entspricht der physikalischen Verbindung zwischen Sender und Empfänger. Dabei kann es sich um ein Kupferkabel, ein Koaxialkabel, ein Glasfaserkabel oder um elektromagnetische Wellen handeln. In dieser Schicht werden die Bits eines Frames als elektrische, elektromagnetische oder optische Signale übertragen (Forouzan, 2013, Kurose & Keith, 2017).

Übung

Laden Sie Beispielpakete (.pcap- oder .cap-Dateien) für HTTP und TCP von der Wireshark-Website herunter und öffnen Sie die Pakete (Wireshark Foundation, ohne Jahresangabe). Untersuchen Sie die Einheiten der übertragenen Daten der einzelnen Protokollschichten. Ermitteln Sie die Paketsequenznummer, die Anzahl der Schichten, die Headerbitlänge sowie die Adressen. Finden Sie außerdem heraus, welche Informationen Sie für die verschiedenen Schichten extrahieren können.

### Beispiele für Internetprotokolle und -dienste

Der TCP/IP-Protokollstapel ist eine Sammlung von Protokollen für das Internet. Diese Protokolle werden in Schichten untergliedert. Dabei werden in jeder Schicht eine Reihe von Protokollen für die verschiedenen Dienste eingesetzt. In diesem Abschnitt werden die gängigsten Protokolle der TCP/IP-Protokollfamilie erläutert.

###### Protokolle der Anwendungsschicht

Hypertext Transfer Protocol (HTTP)

HTTP definiert, wie ein Client-Server-Programm codiert wird, um webbasierte Objekte von einem Server abzurufen. Die Übertragung eines Objekts vom Server an den Client erfolgt über Anforderungs- und Antwortnachrichten, die von HTTP definiert werden. Für HTTP wird Port 80 verwendet (Solomon & Kim, 2021). Der Client sendet eine Anforderungsnachricht an den Server, welche die Portnummer 80 und die URL (Uniform Resource Locator) oder Webadresse des gewünschten Objekts enthält. Bei diesem Objekt kann es sich beispielsweise um eine HTML-Datei, eine Text- oder Dokumentdatei, ein Bild oder eine Audio- bzw. Videodatei handeln (Comer, 2015, 2018; Kurose & Keith, 2017). Der Server antwortet mit der Antwortnachricht, die das angeforderte Objekt enthält. Eine HTTP-Verbindung kann entweder persistent oder nicht persistent sein (Kurose & Keith, 2017).

File Transfer Protocol (FTP)

FTP wird zum Übermitteln von Dateien vom Server zum Client verwendet und eignet sich besser für die Übertragung großer Dateien als HTTP. Außerdem werden bei FTP Aspekte wie die Dateibenennungskonvention, das Verzeichnissystem oder die verwendete Methode zur Datenformatierung berücksichtigt. Ein wichtiger Vorteil, da diese Aspekte nicht auf jedem System gleich sind (Comer, 2015; Comer, 2018). Für FTP wird TCP-Port 21 verwendet (Solomon & Kim, 2021).

Simple Mail Transfer Protocol (SMTP)

SMTP wird zum Weiterleiten von E-Mails von einem Client an einen E-Mail-Server bzw. zwischen E-Mail-Servern verwendet. Für SMTP wird TCP-Port 25 verwendet (Kurose & Keith, 2017).

Post Ofﬁce Protocol-Version 3 (POP3)

POP3 wird eingesetzt, um E-Mails vom E-Mail-Server auf den Computer oder das System des Empfängers abzurufen. Dazu stellt der Client zunächst an Port 110 eine TCP-Verbindung her. Anschließend sendet er den Benutzernamen und das Kennwort für den Authentifizierungsprozess (Kurose & Keith, 2017; Tanen baum & Wetherall, 2014). Nachdem der Authentifizierungsprozess durchlaufen wurde, sollte dem Client die Liste der empfangenen E-Mails angezeigt werden. Bei POP3 kann der Client E-Mails nach dem Abrufen beibehalten oder löschen.

Internet Message Access Protocol-Version 4 (IMAP4)

Wie POP3 wird auch IMAP4 verwendet, um E-Mails vom Server abzurufen. Im Gegensatz zu POP3 bietet IMAP4 Benutzern jedoch die Möglichkeit, E-Mails in Verzeichnissen oder Ordnern zu organisieren. Auch eine Suchoption ist verfügbar. Für IMAP4 wird Port 143 bzw. 993 für eine sichere TCP-Verbindung verwendet (Tanenbaum & Wetherall, 2014).

Domain Name System (DNS)

DNS wird verwendet, um die IP-Adresse aus einem Hostnamen abzurufen. Dazu wird eine verteilte Datenbank verwaltet, die in einer Hierarchie aus DNS-Servern implementiert ist. Für DNS wird UDP-Port 53 verwendet (Kurose & Keith, 2017). Im Allgemeinen wird DNS von Protokollen der Anwendungsschicht (z. B. HTTP und SMTP) verwendet, um einen Hostnamen in eine IP-Adresse zu übersetzen.

Simple Network Management Protocol (SNMP)

SNMP wird zum Verwalten und Überwachen von Netzwerken und Geräten verwendet. Das Protokoll wird auf mehreren Servern (den sogenannten Verwaltungsstationen) eingesetzt, um andere Geräte zu steuern und zu überwachen. Diese Geräte werden als Agents bezeichnet. Dabei nutzt SNMP zwei Unterprotokolle: Structure of Management Information (SMI) und Management Information Base (MIB) (Kurose & Keith, 2017).

###### Protokolle der Transportschicht

Transmission Control Protocol (TCP)

TCP wird für eine zuverlässige Datenübertragung verwendet. Zu diesem Zweck bietet das Protokoll die folgenden Dienste: Fehlererkennung, Übermittlung in der richtigen Reihenfolge, Flusskontrolle, Überlastungserkennung und -kontrolle, Erkennung von Paketverlust sowie erneute Übermittlung. Die Fehlererkennung erfolgt anhand einer Internetprüfsumme. Um eine zuverlässige Übermittlung in der richtigen Reihenfolge sicherzustellen, nutzt TCP auf Empfängerseite einen Puffer. Zur Flusskontrolle wird dafür gesorgt, dass das Übertragungsfenster kleiner ist als die freie Pufferkapazität im Empfängersystem. Bei TCP sendet der Empfänger eine Bestätigung, sobald er ein Paket empfangen hat. Auf Senderseite werden mithilfe eines Timers verzögerte Bestätigungen ermittelt. Wird die für die Bestätigung festgelegte Timeout-Dauer überschritten, wird das Paket als verlorenes Paket betrachtet. Zur Wiederherstellung wird das verlorene Paket erneut übermittelt. Um einer Netzwerküberlastung entgegenzuwirken, stehen verschiedene TCP-Varianten zur Verfügung. Die beiden gängigsten Varianten sind TCP Tahoe und TCP Reno (Kurose & Keith, 2017).

User Datagram Protocol (UDP)

Da bei TCP eine Übermittlung in der richtigen Reihenfolge sowie eine erneute Übermittlung verloren gegangener Pakete sichergestellt wird, kann es bei Echtzeitdiensten zu unerwünschten Verzögerungen kommen. UDP ist für Echtzeitdienste und Dienste konzipiert, bei denen eine bestimmte Bitrate zugesichert wird. Da UDP ein

Die Internetprotokollfamilie

verbindungsloses Protokoll ist, wird der Paketempfang nicht vom Empfänger bestätigt. Folglich bietet UDP keine Dienste für Flusskontrolle, Erkennung von Paketverlust, erneute Übermittlung oder Überlastungskontrolle. Es wird jedoch ein Prüfsummenmechanismus verwendet, um eine Fehlererkennung zu ermöglichen (Kurose & Keith, 2017).

Für Echzeit-Multimedia-Dienste werden Real-Time Transport Protocol (RTP) und Stream Control Transmission Protocol (SCTP) verwendet. Wenngleich es sich um Protokolle der Transportschicht handelt, werden die RTP- und SCTP-Datagramme in UDP-Headerbits gekapselt (Tanenbaum & Wetherall, 2014). Anders als bei Anwendungen wird für Payload-RTP keine bestimmte Portnummer verwendet. Die Portnummer für RTP wird willkürlich gewählt, es handelt sich jedoch immer um eine gerade Zahl. Die nächste ungerade Zahl wird als Portnummer für das Begleitprotokoll SCTP verwendet. RTP unterstützt lediglich einen Datentyp. Bei mehreren Datentypen kommt SCTP als Begleitprotokoll zum Einsatz.

###### Protokolle der Internetschicht

Internet Protocol (IP)

IP ist innerhalb der TCP/IP-Protokollfamilie der Internetschicht zugeordnet. Es ist in zwei Versionen verfügbar: IPv4 und IPv6. IPv4 verwendet eine 32-Bit-Adresse, IPv6 eine 128-Bit-Adresse (Kurose & Keith, 2017). Da die Kapazität von IPv4 schon bald nicht mehr für die sich schnell ändernde Menge an Internetgeräten ausreichen wird, wird IPv4 schrittweise durch IPv6 ersetzt. Dieser Vorgang könnte jedoch einige Jahrzehnte dauern. Aktuell wird für die Mehrzahl der Internetgeräte weiterhin IPv4 verwendet. Neben IP bietet die TCP/IP-Internetschicht einige ergänzende Protokolle. Dazu zählen Dynamic Host Conﬁguration Protocol (DHCP), Internet Control Message Protocol (ICMP), Internet Group Management Protocol (IGMP) und Address Resolution Protocol (ARP) (Tanenbaum & Wetherall, 2014).

Wenn ein neues Gerät zu einem Netzwerk hinzugefügt wird, verfügt das Gerät zunächst nicht über eine IP-Adresse. Die IP-Adresse kann manuell oder automatisch eingerichtet werden. Zur automatischen Einrichtung einer IP-Adresse für ein neues Gerät wird DHCP verwendet. Dabei übermittelt das neue Gerät bzw. der neue Client zunächst eine Discover-Nachricht per Broadcast. In dieser Nachricht sind die Quell- und Ziel-IP-Adressen auf 0.0.0.0 bzw. 255.255.255.255 festgelegt. Die Quell- und Zielportnummern sind 68 bzw. 67 (Kurose & Keith, 2017). Der DHCP-Server antwortet mit einer Angebotsnachricht, die eine IP-Adresse für den Client enthält.

ARP arbeitet unterhalb der IP-Schicht. Mithilfe dieses Protokolls werden Adressen der Internetschicht in Adressen der Sicherungsschicht übersetzt. Basierend auf der IP-Adresse eines Geräts wird also die entsprechende MAC-Adresse abgerufen. Dazu übermittelt der Sender die Ziel-IP-Adresse per Broadcast an alle Knoten. Der Zielknoten erkennt seine IP-Adresse und gibt die MAC-Adresse an den Sender zurück. Alle anderen Knoten verwerfen die Nachricht (Forouzan, 2013). ARP ist in RFC 826, 5227 und 5494 definiert (Arkko & Pignataro, 2009).

ICMP ist ein wichtiger Bestandteil von IP und bietet Fehlerberichte zur Datagrammverarbeitung. In diesen Berichten kann z. B. gemeldet werden, wenn ein Datagramm nicht zugestellt wird, ein Gateway nicht über ausreichend Pufferkapazität zum Verarbeiten des Datagramms verfügt oder das Gateway eine bessere Route zum Umleiten der Daten bieten kann. (Forouzan, 2013). ICMP ist in RFC 792, 777 und 760 definiert (Postel, 1981).

IGMP bietet ein Gruppenmitgliedschafts-Reporting für Multicasting-Dienste. Der IP-Host nutzt dieses Protokoll, um den //unmittelbaren Multicast-Nachbarrouter über Multicast-Gruppenmitgliedschaften zu informieren (Forouzan, 2013). Wie ICMP ist dieses Protokoll ebenfalls Teil von IP. Inzwischen wird eine aktualisierte Version von IGMP (IGMPv2) verwendet, die in RFC 2236 und 3376 definiert ist (Cain et al., 2002).

Interior Gateway Protocols (IGPs)

Jeder Internetdienstanbieter (Internet Service Provider, ISP) kann als Domäne oder autonomes System (AS) betrachtet werden. Das innerhalb einer Domäne verwendete Routingprotokoll wird als Intradomain-Routingprotokoll bzw. Interior Routing Protocol bezeichnet. Beispiele für diese Art von Routingprotokoll sind Routing Information Protocol (RIP), Open Shortest Path Frst (OSPF) oder Border Gateway Protocol (BGP). Verschiedene Domänen können über unterschiedliche IGPs verfügen. RIP basiert auf einem Distanzvektor-Routingalgorithmus, OSPF auf einem Link-State-Routingalgorithmus und BGP auf einem Pfadvektor-Routingalgorithmus (Forouzan, 2013). IGP ist in RFC 3906 definiert (Shen & Smit, 2004).

Exterior Gateway Protocols (EGPs)

EGPs werden für das Interdomain-Routing verwendet, um einen Pfad zwischen Knoten zu bestimmen, die sich in unterschiedlichen Domänen oder bei unterschiedlichen ISPs befinden (Forouza, 2013). EGP ist in RFC 827 und 904 definiert (Mills, 1984).

Protocol Independent Multicast (PIM)

PIM ist ein „One-to-Many“- und „Many-to-Many“-Multicast-Protokoll. Hauptmerkmal dieses Protokolls ist, dass die Routingtabelle basierend auf den Informationen bzw. der Routingtabelle erstellt wird, die von den Unicast-Protokollen (beispielsweise Distanzvektor- oder Link-State-Protokolle) bereitgestellt werden (Forouzan, 2013). PIM ist in RFC 7761 und 8736 definiert (Venaas & Retana, 2020).

###### Protokolle der Sicherungsschicht

Reverse Address Resolution Protocol (RARP)

Im Gegensatz zu ARP, wo basierend auf einer bekannten IP-Adresse nach einer unbekannten MAC-Adresse gesucht wird, wird bei RARP basierend auf einer bekannten MAC-Adresse nach einer unbekannten IP-Adresse gesucht (Forouzan, 2013). RARP ist in RFC 903 definiert (Finlayson et al., 1984). Eine aktualisierte Version von RARP, das sogenannte Dynamic RARP (DRARP), ist in RFC 1931 definiert (Brownell, 1996).

Neighbor Discovery Protocol (NDP)

Bei diesem Protokoll der Sicherungsschicht aus der TCP/IP-Protokollfamilie wird IPv6 verwendet. NDP wird eingesetzt, um Netzwerkinformationen zu erfassen. So werden beispielsweise Knoten, Router oder Erreichbarkeitsinformationen für die Routen zu benachbarten Knoten ermittelt. NDP ist in RFC 4861, 5942, 6980, 7048, 7527, 7559, 8028, 8319 und 8425 definiert (Department of Defence High Performance Computing Modernization (DoD HPC), ohne Jahresangabe).

Übung

Laden Sie Beispielpakete (.pcap- oder .cap-Dateien) für HTTP, SMTP, DNS, TCP, UDP, SCTP, DHCP und ARP von der Wireshark-Website herunter und öffnen Sie die Pakete (Wireshark Foundation, ohne Jahresangabe). Untersuchen Sie die Pakete der einzelnen Protokolle sowie die Nachrichtentypen, die zwischen den Sender- und Empfängerknoten ausgetauscht wurden, und sehen Sie sich die Adressen und Inhalte der einzelnen Nachrichten an.

Die Internetprotokollfamilie

### Sicherheitsaspekte bei der Kommunikation im Internet

In unserer heutigen Welt hängen fast alle Aspekte unseres Alltags – vom Gesundheitswesen über das Finanzwesen bis hin zum Bildungswesen – vom Internet ab. Dies hat zur Folge, dass die Sicherheitsrisiken des Internets sich auch auf viele Bereiche unseres Lebens auswirken. Und gerade deshalb ist es so entscheidend, für eine sichere Kommunikation im Internet zu sorgen. Sichere Kommunikation bedeutet, dass die Vertraulichkeit, Integrität und Verfügbarkeit von Daten jederzeit sichergestellt ist. Die Kombination dieser drei Faktoren wird basierend auf den englischen Begriffen Confidentiality, Integrity und Availability als CIA-Triade bezeichnet (Stallings, 2017).

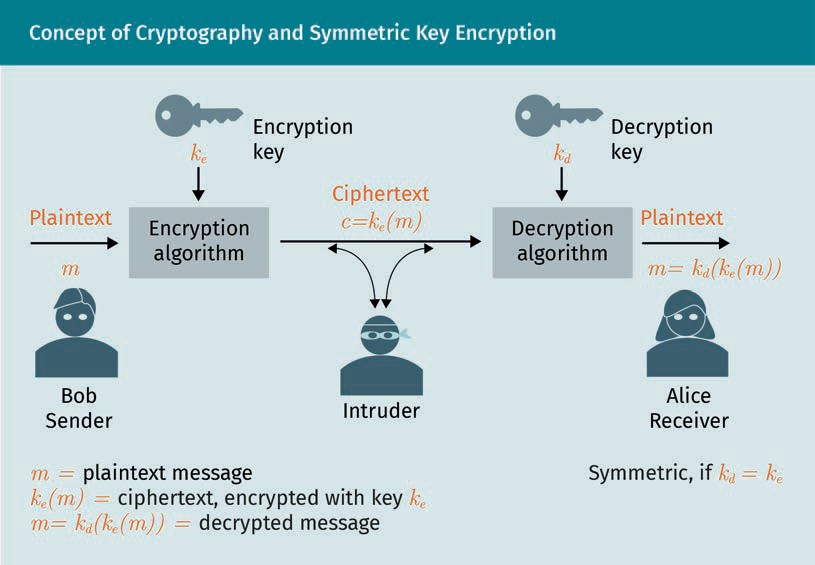
* Vertraulichkeit bedeutet, dass Daten keinesfalls für nicht autorisierte Benutzer offengelegt werden dürfen.
* Mit Integrität ist gemeint, dass verhindert werden muss, dass Daten durch nicht autorisierte Benutzer bearbeitet oder verändert werden.
* Verfügbarkeit bezieht sich darauf, dass autorisierte Benutzer jederzeit in der Lage sein sollen, auf die Ressourcen zuzugreifen. Nicht autorisierte Benutzer hingegen sollten keinen Zugang zu den Ressourcen erhalten.

Eine entscheidende Säule für die Sicherheit im Internet ist die Kryptografie. Es gibt im Wesentlichen zwei Arten von kryptografischen Verfahren: die symmetrische und die asymmetrische Kryptografie, die auch Kryptografie mit öffentlichem Schlüssel genannt wird. Diesen beiden Kategorien wiederum sind eine Vielzahl unterschiedlicher kryptografischer Verfahren zugeordnet. In diesem Abschnitt werden die aktuellsten und am häufigsten verwendeten Verfahren erläutert.

Jedes dieser Verfahren umfasst einen Verschlüsselungs- und einen Entschlüsselungsalgorithmus, bei denen jeweils zwei Eingaben verarbeitet werden, um eine Ausgabe zu erzeugen. Die beiden relevanten Eingaben für

Verschlüsselungsalgorithmen sind eine Klartext- oder Nachrichteneingabe 7 und ein Schlüssel ke . Die Ausgabe ist ein Geheim- bzw. Chiffretext c = ke 7 . Die relevanten Eingaben für einen Entschlüsselungsalgorithmus sind ein Geheimtext c und ein Schlüssel k’ , die Ausgabe ist der entschlüsselte Text ’ = k’ c = k’ ke 7. Wenn der Chiffretext ordnungsgemäß entschlüsselt wurde, entspricht er der Klartexteingabe (Stallings, 2017).

In der folgenden Abbildung ist das Konzept eines kryptografischen Verfahrens dargestellt. Der Sender Bob sendet eine Nachricht 7 an die Empfängerin Alice über ein sicheres Verfahren, um die Nachricht gegen Angreifer im Netz zu schützen.



###### Kryptografie mit symmetrischem Schlüssel

Bei der Kryptografie mit symmetrischem Schlüssel verwenden der Verschlüsselungs- und der Entschlüsselungsalgorithmus denselben Schlüssel: ke = k’ = k. Das einfachste Beispiel für dieses Verfahren ist die Caesar-Verschlüsselung (Caesar Cipher Cryptography). Dabei variiert der Schlüssel für den CCC-Verschlüsselungsalgorithmus zwischen 0 und 25: k ∈ 0, 1, 2, …, 25 (Stallings, 2017). Unter Verwendung des Schlüssels k wird der Chiffretext berechnet, indem ein Buchstabe K der Klartexteingabe durch K + k th

ersetzt wird. Beispiel: Wenn der Schlüssel k = 1 und die Klartexteingabe 7 = c8t ist,

ist der Chiffretext c = k 7 = K + k 7B’ 26 = ’bA. Wenn k = 2 ist, ist der Chiffretext c = k 7 = K + k 7B’ 26 = ec0. Der entschlüsselte Text wird mit der folgenden Formel berechnet: ’ = K − k 7B’ 26*.* Die Caesar-Verschlüsselung ist jedoch ein recht simples Beispiel für ein kryptografisches Verfahren mit symmetrischem Schlüssel und der verschlüsselte Code lässt sich leicht dechiffrieren.

Es gibt eine Vielzahl weiterer Beispiele für dieses Verschlüsselungsverfahren. In den meisten Fällen lassen sich die Geheimtexte mithilfe moderner Computer jedoch problemlos entschlüsseln. Der am häufigsten verwendete Algorithmus mit symmetrischem Schlüssel ist AES (Advanced Encryption Standard). Ein weiterer gängiger Algorithmus ist DES (Data Encryption Standard). Da sich DES-Code jedoch ebenfalls recht leicht entschlüsseln lässt, wird stattdessen die verbesserte Version 3DES verwendet. AES, DES und 3DES sind Blockverschlüsselungsverfahren. Bei jedem dieser Verfahren wird der Klartext in Blöcke fester Größe unterteilt, bevor die einzelnen Blöcke verschlüsselt werden. Für Verschlüsselung, Entschlüsselung und Schlüsselgenerierung werden jeweils mehrere Schritte ausgeführt, die als Runden oder Iterationen bezeichnet werden. In der nachfolgenden Tabelle sind die Unterschiede der AES-, DES- und 3DES-Algorithmen aufgeführt (Stallings, 2017). Ein entscheidender Nachteil der Kryptografie mit symmetrischem Schlüssel ist, dass Angreifer, die über den Schlüssel verfügen,

Die Internetprotokollfamilie

den gesamten Geheimtext entschlüsseln können. Aus diesem Grund muss der Schlüssel über eine sichere Methode zwischen Sender und Empfänger ausgetauscht werden. Für den Austausch von Schlüsseln zwischen Sender und Empfänger wird ein Verschlüsselungsverfahren mit asymmetrischem bzw. öffentlichem Schlüssel verwendet.

|  |  |  |  |
| --- | --- | --- | --- |
| Unterschiede zwischen AES, DES und 3DES | | | |
|  | AES | DES | 3DES |
| Schlüsselgröße (Bits) | 128/192/256 | 56 | 168 |
| Klartext-Blockgröße (Bits) | 128 | 64 | 64 |
| Anzahl an Runden | 10/12/14 | 16 | 48 |

###### Kryptografie mit asymmetrischem bzw. öffentlichem Schlüssel

Bei einem Verschlüsselungsverfahren mit asymmetrischem Schlüssel wird für den Verschlüsselungsalgorithmus ein anderer Schlüssel verwendet als für den Entschlüsselungsalgorithmus: ke ≠ k’. Für den Verschlüsselungsalgorithmus wird ein Schlüsselpaar benötigt, das aus dem öffentlichen Schlüssel (k4+A) und dem privaten Schlüssel (ks+ ,) des Senders besteht. Für den Entschlüsselungsalgorithmus wird ebenfalls ein Schlüsselpaar benötigt, das sich aus dem öffentlichen Schlüssel (k,+ A) und dem privaten Schlüssel (k,+ ,) des Empfängers zusammensetzt. Die öffentlichen Schlüssel sind für beliebige Benutzer zugänglich, ein privater Schlüssel jedoch nur für den Besitzer des Schlüssels. Der Geheimtext wird erstellt, indem der Klartext (7) mit dem öffentlichen Schlüssel des Empfängers verschlüsselt wird (k,+ A). Es gilt also c = k,+ A 7 *.* Zur Entschlüsselung wird der private Schlüssel des Empfängers verwendet: ’ = k,+ , c = k,+, k,+A 7 = 7. Das am häufigsten verwendete kryptografische Verfahren mit asymmetrischem Schlüssel ist der RSA-Algorithmus, der nach den Mathematikern Ramir, Shivest und Adleman benannt ist, die den

Algorithmus 1978 erstmalig öffentlich beschrieben. Weitere gängige Verschlüsselungsverfahren mit asymmetrischem Schlüssel sind der ECC-Algorithmus (Elliptic Curve Cryptography) und der Difﬁe-Hellman-Algorithmus (Stallings, 2017; Kurose & Keith, 2017).

Der RSA-Algorithmus

Der RSA-Algorithmus umfasst die folgenden drei Phasen (Stallings, 2017):

1. Schlüsselgenerierung: Zum Generieren des Schlüsselpaars aus öffentlichem und privatem Schlüssel wählt der Algorithmus zwei Primzahlen p und / . Dabei gilt p ≠ /. Anschließend erfolgt die Berechnung von n = p/ und cp n = p − 1 / − 1 sowie die Auswahl der Ganzzahl e . Dabei gilt, dass cp n und e teilerfremd sind und 1 < e < cp n . Dann wird ’ = e−1 7B’ cp n berechnet. Der öffentliche Schlüssel ist nun kpA = e, n *, der* private Schlüssel kp, = ’, n
2. Verschlüsselung: Der Geheimtext wird mit c = 7e 7B’ n berechnet.
3. Entschlüsselung: Der entschlüsselte Text wird mit ’ = c’ 7B’ n berechnet.

Gehen wir von folgendem Beispiel aus: 7 = A, k,+A = 7, 187 und k,+, = 23, 187 . Um zunächst den Geheimtext zu berechnen, muss der Buchstabe A in einen Dezimalwert umgewandelt werden. Laut ASCII-Tabelle (American Standard Code for Information Exchange) ist dem Buchstaben A der Dezimalwert 65 zugewiesen.

(Carnegie Mellon University (CMU), ohne Jahresangabe). Der Geheimtext ist also:

c = 657 7B’ 187 = 142. Der entschlüsselte Text lautet:

’ = 14223 7B’ 187 = 65 = A.

Authentifizierung mithilfe eines Verschlüsselungsverfahrens mit öffentlichem Schlüssel

//Neben der Verschlüsselung mit symmetrischem Schlüssel ist die Benutzerauthentifizierung eine weitere Verwendungsmöglichkeit für die Verschlüsselung mit öffentlichem Schlüssel. Sobald ein Empfänger eine Nachricht empfängt, muss sichergestellt werden, dass die Nachricht tatsächlich vom angegebenen Benutzer gesendet wurde. Zu diesem Zweck kann die Nachricht m mit dem privaten Schlüssel des Senders (ks+ ,) verschlüsselt werden, der an den Geheimtext angefügt wird. Der Empfänger entschlüsselt den Text mithilfe des öffentlichen Schlüssels des Senders (ks+ A). Da nur der Sender über den privaten Schlüssel verfügen kann, kann die Nachricht nur ordnungsgemäß entschlüsselt werden, wenn sie tatsächlich vom angegebenen Benutzer gesendet wurde. Das Verschlüsseln der gesamten Nachricht ist jedoch sehr zeitaufwendig. Eine schnellere Lösung ist das Anwenden einer Hash-Funktion (Stallings, 2017; Kurose & Keith, 2017).

Hash-Funktion

Eine Hash-Funktion wird in erster Linie verwendet, um unabhängig von der Länge der Eingabe eine Ausgabe mit fester Länge zu erzeugen. Das Ergebnis einer 128-Bit-Hash-Funktion ist z. B. unabhängig von der Eingabelänge immer eine 128-Bit-Ausgabe. Die Ausgabe kann abhängig von unterschiedlichen Eingaben variieren, die Ausgabelänge ist jedoch immer 128 Bits. Eine gute Hash-Funktion weist die folgenden Eigenschaften auf:

* Für eine Nachricht 7 ist das Ergebnis einer n-Bit-Hash-Funktion die Ausgabe h 7 mit einer Länge von n Bit.
* Es sollte rechnerisch unmöglich sein, die Umkehrung der Hash-Funktion

h−1 h 7 = 7 zu finden. Dies wird als „Preimage Resistance“ (Resistenz gegen das Finden von Urbildern) bezeichnet.

* Es sollte rechnerisch unmöglich sein, eine Nachricht 7’ zu erzeugen, für deren Hash-Wert h 7’ = h 7 gilt. Dies wird als „Second Preimage Resistance“ (Resistenz gegen das Finden zweiter Urbilder) bezeichnet.

Häufig verwendete Hash-Algorithmen sind Message Digest 5 (MD5), Secure Hash Algorithm-1 (SHA-1) und SHA-2.

Digitale Signatur

Der Empfänger muss sicherstellen, dass die empfangene Nachricht vom eigentlichen Benutzer stammt und nicht bearbeitet oder geändert wurde. Zu diesem Zweck wird eine digitale Signatur verwendet, die der Authentifizierung dient und die Integrität sicherstellt. In einer digitalen Signatur wird zunächst der Hash-Wert der Nachricht h 7 berechnet. Anschließend wird h 7 mit dem privaten Schlüssel des Senders verschlüsselt, um die digitale Signatur

s = ks+ , h 7 zu erzeugen. Aus Vertraulichkeitsgründen wird die Nachricht m mit dem öffentlichen Schlüssel des Empfängers verschlüsselt, um den Geheimtext c = k,+ A 7 zu finden. Dann wird die Signatur an den Geheimtext s c = ks+ , 7 k,+A 7 angefügt bzw. mit diesem Text zusammengefügt und an den Empfänger gesendet. Der Empfänger nutzt den öffentlichen Schlüssel des Senders, um den empfangenen Hash-Wert ks+A ks+, h 7 zu entschlüsseln. Zum Entschlüsseln der empfangenen Nachricht ’ = k,+ , k,+A 7 nutzt der Empfänger seinen privaten Schlüssel. Anschließend berechnet der Empfänger den Hash-Wert der entschlüsselten Nachricht h ’ = h k,+, k,+A 7 . Wenn h ’ = k,+A ks+, h 7 oder h k,+, k,+A 7 = ks+A ks+, h 7 ist, dann ist der Sender authentisch und die Integrität ist sichergestellt (Stallings, 2017; Kurose & Keith, 2017).

Die Internetprotokollfamilie

###### Sicherheit in TCP/IP-Schichten

Der TCP/IP-Protokollstapel umfasst keine Sicherheitsschicht. In den einzelnen Schichten werden jedoch verschiedene Sicherheitsprotokolle verwendet, um für eine hohe Sicherheit zu sorgen. Im Folgenden sind die am häufigsten verwendeten Protokolle der verschiedenen Schichten beschrieben.

Sicherheit in der Anwendungsschicht

In der Anwendungsschicht sind eine Vielzahl von Sicherheitsprotokollen verfügbar, mit denen sich verschiedene Anwendungen schützen lassen. Nachfolgend sind einige Sicherheitsprotokolle der Anwendungsschicht aufgeführt:

* Hypertext Transfer Protocol Secure (HTTPS). HTTPS ist eine sichere Version von HTTP, da HTTP selbst keine Sicherheitsfunktion bietet. Um Daten gegen Lauschangriffe und Man-in-the-Middle-Angriffe zu schützen, unterstützt HTTPS einen Verschlüsselungsmechanismus. HTTPS entspricht im Wesentlichen HTTP, bietet jedoch zusätzlich eine Verschlüsselung mittels SSL/TLS (Secure Sockets Layer/Transport Layer Security). HTTPS kann anstelle von Port 80 den Port 443 verwenden (Tanenbaum & Wetherall, 2014).
* File Transfer Protocol Secure (FTPS). FTPS ist eine Erweiterung von File Transfer Protocol (FTP) und nutzt wie HTTPS ebenfalls SSL/TLS. Dieses Protokoll ist in RFC 4217 definiert (Ford-Hutchinson, 2005).
* Secure Shell (SSH). SSH ermöglicht einen sicheren Remotezugriff und nutzt Port 22 (Tanenbaum & Wetherall, 2014).
* Pretty Good Privacy (PGP). Dieses Protokoll wird für eine sichere Authentifizierung, Vertraulichkeit und digitale Signaturen verwendet (Stallings, 2017).

Sicherheit in der Transportschicht

Zum Schutz von Transport- und Anwendungsschicht wird eine Sicherheitsebene zwischen diesen beiden Schichten implementiert, die als Secure Sockets Layer (SSL) bezeichnet wird. Die aktualisierte Version von SSL heißt Transport Layer Security (TLS). TLS nutzt in erster Linie AES für Vertraulichkeit, den Difﬁe-Hellman-Algorithmus (mit Unterstützung des Modus für elliptische Kurven) oder RSA für den Schlüsselaustausch und Secure Hash Algorithm (SHA) für das Hashing. TLS/SSL unterstützt eine Reihe von Verschlüsselungsalgorithmen, die unter dem Begriff Cipher Suite zusammengefasst sind (Stallings, 2017). Das Client/Server-Paar entscheidet beim TLS-Handshake, also beim Herstellen der Verbindung, welche Cipher Suite verwendet wird.

Sicherheit in der Internetschicht

In dieser Schicht wird IPsec (Internet Protocol Security) eingesetzt, um die Sicherheit zu erhöhen. IPsec kann sowohl mit TCP- als auch mit UDP-Transportprotokollen verwendet werden. Mit IPsec wird das gesamte Paket geschützt und gekapselt, das an die IP-Schicht übergeben wird. Dies umfasst auch sämtliche Headerbits der danebenliegenden Schicht (Forouzan, 2014).

OWASP

OWASP steht für Open Web Application Security Project (OWASP). Auf dieser Online-Plattform für Open-Source-Content werden Artikel, Dokumentation, Softwaretools und Technologien im Zusammenhang mit der Sicherheit von Webanwendungen bereitgestellt.

Übung

* 1. Laden Sie Beispielpakete (.pcap- oder .cap-Dateien) für HTTPS und HTTP von der Wireshark-Website herunter und öffnen Sie die Pakete (Wireshark Foundation, ohne Jahresangabe). Vergleichen Sie die Pakete und notieren Sie sich die Unterschiede.
  2. Laden Sie auf derselben Website Beispielpakete (.pcap- oder .cap-Dateien) für TLS herunter. Ermitteln Sie, welche Arten von Verschlüsselungsalgorithmen unterstützt werden, sowie den jeweiligen Verschlüsselungsschlüssel.

Zusammenfassung

Das Internet ist das komplexeste und am meisten genutzte Computernetzwerk mit der größten Reichweite weltweit. Der Protokollstapel für das Internet wird TCP/IP-Protokollfamilie genannt. Die Kommunikationsprotokolle für das Internet werden in fünf Schichten untergliedert: Anwendungs-, Transport-, Internet-, Sicherungs- und Bitübertragungsschicht. Im Gegensatz zum OSI-Modell umfasst das TCP/IP-Modell keine Darstellungs- und Sitzungsschicht. Die Dienste dieser beiden Schichten sind in der Anwendungs- und Transportschicht vereint. Zu den gängigen Protokollen der Anwendungsschicht zählen HTTP für Webdienste, SMTP für die Weiterleitung von E-Mails, POP3 und IMAP für das Abrufen von E-Mails von Servern und DNS für die Übersetzung von Hostnamen in IP-Adressen. Die meistgenutzten Protokolle der Transportschicht sind TCP für eine zuverlässige Datenübertragung und UDP für eine nicht zuverlässige Übermittlung (z. B. Echtzeit-Übertragungen). Das Protokoll der Internetschicht ist IP bzw. IPv4. Für IPv4 sind eine Vielzahl von ergänzenden Protokollen verfügbar. Dazu zählen DHCP, ICMP, IGMP und APR. Zwei häufig verwendete Protokolle der Sicherungsschicht sind RARP und NDP. TLS wird eingesetzt, um die Sicherheit der Transportschicht zu erhöhen. Für eine sichere Kommunikation in der Internetschicht wird IPsec genutzt. Zu den Sicherheitsprotokollen der Anwendungsschicht zählen HTTPS, SSH, PGP und FTPS.



# Lektion 4

## Architekturen verteilter Systeme

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… die Konzepte verteilter Systemarchitekturen zu beschreiben. Dazu zählen u. a. die Client-Server-Architektur und die serviceorientierte Architektur (Service-Oriented Architecture).

… die Konzepte von Edge Computing, Cloud Computing und Peer-to-Peer-Computing darzulegen.

… die Konzepte von Webdiensten und Microservices zu erläutern.

… verschiedene Arten von verteilten Systemen zu analysieren.

… verteilte Systeme zu entwerfen und zu implementieren.

DL-E-DLMCSNDS01-U04

1. Architekturen verteilter Systeme

### Einführung

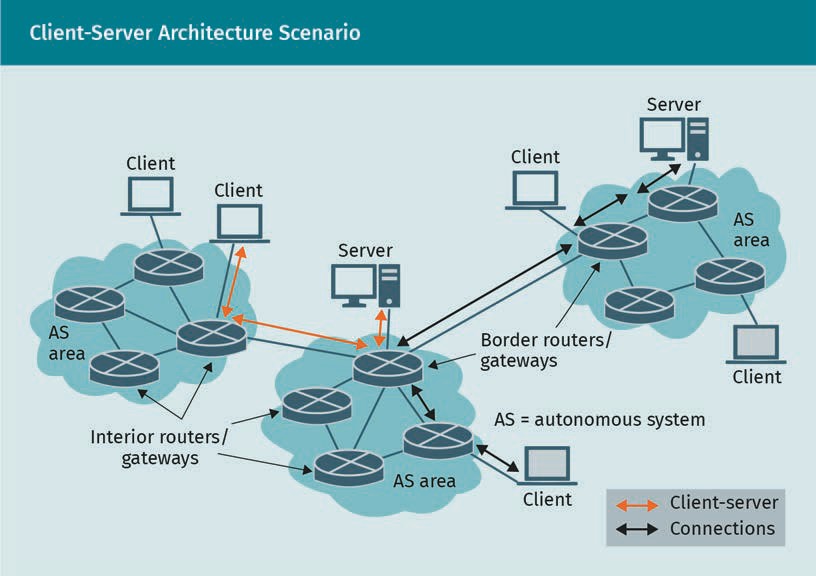
Das Konzept eines verteilten Systems ist in verschiedenen Quellen unterschiedlich definiert. Vereinfacht ausgedrückt führt ein verteiltes System mehrere Programme oder Prozesse auf mehreren Knoten aus, um ein gemeinsames Ziel zu erreichen bzw. einen gemeinsamen Dienst bereitzustellen. Ein gutes Beispiel für ein verteiltes System ist ein Computernetzwerk, insbesondere das Internet. Gehen wir z. B. von einem Webdienst aus, bei dem ein Computer eine Clientanwendung wie einen Browser ausführt, um Anforderungen zu senden und Webdienste von einem anderen Computer zu empfangen, der einen Serverprozess wie den Apache HTTP-Webserver ausführt. Im Internet werden vorwiegend Client-Server-basierte verteilte Architekturen genutzt. Eine weitere gängige verteilte Architektur ist ein Peer-to-Peer-Netzwerk (P2P), bei dem eine Gruppe von Computern eine gemeinsame Anwendung oder einen gemeinsamen Prozess nutzt, um Dateien untereinander auszutauschen. Aufgrund der Weiterentwicklung und zunehmenden Komplexität von Internetdiensten wurden eine Reihe erweiterter Varianten von Client-Server-Architekturen eingeführt. Beispiele für diese Architekturen sind serviceorientierte Architekturen, Edge Computing, Cloud Computing, Fog Computing, Ubiquitous Computing und Pervasive Computing. In dieser Lektion werden die verschiedenen Typen von verteilten Systemen genauer untersucht. Dabei wird insbesondere auf Client-Server-Architekturen, serviceorientierte Architekturen, Edge Computing und Cloud Computing sowie Peer-to-Peer-Netzwerke eingegangen.

### Client-Server-Architekturen

Client-Server-Architekturen sind die für Computernetzwerke am häufigsten verwendete Art einer verteilten Architektur. Die Begriffe „Client“ und „Server“ beziehen sich dabei auf Softwareprogramme. Im Allgemeinen wird der Begriff „Server“ jedoch für Computer verwendet, die eine hohe Verarbeitungsleistung sowie eine große Arbeitsspeicher- und Speicherkapazität aufweisen und das Serverprogramm ausführen. Der Client ist der Computer, auf dem das Clientprogramm ausgeführt wird. Er entspricht dem Prozess, der Dienste anfordert und empfängt, der Server ist der Prozess, der die Dienste für Clients bereitstellt (Comer, 2015). Server und Client werden anhand einer Kombination aus Bezeichnern oder Adressen (Computer- und Anwendungsadresse) identifiziert. Diese Kombination aus IP-Adresse und Portnummer heißt Socket. Eine IP-Adresse kennzeichnet einen bestimmten Computer oder ein bestimmtes Betriebssystem. Eine Portnummer dient der Identifizierung eines bestimmten Prozesses auf diesem Computer. In einer Client-Server-Architektur ist der Server immer aktiv. Der Client hingegen ist nicht unbedingt immer aktiv. Der Server lauscht dauerhaft an einem Port, um Anforderungen des Clients zu empfangen. Dieser startet die Verbindung, indem er eine Verbindungsanforderung über das Socket an den Server sendet (Kurose & Keith, 2017). Empfängt der Server die Verbindungsanforderung, sendet er eine Bestätigung, mit der er die Anforderung akzeptiert. Sobald die Verbindung hergestellt wurde, sendet der Client die Anforderung für den gewünschten Dienst oder das gewünschte Objekt (z. B. eine Datei oder Daten). Der Server überträgt das angeforderte Objekt an den Client und beendet anschließend die Verbindung. Auf Clientseite wird die Verbindung ebenfalls beendet, nachdem das Objekt empfangen wurde (Kurose & Keith, 2017; Tanenbaum & Wetherall, 2014). Gängige Client-Server-Architekturen sind Webdienste, E-Mail-Dienste und DNS (Domain Name System) (Forouzan, 2013; Tanenbaum & Wetherall, 2014).

Die folgende Abbildung zeigt ein Szenario mit Client-Server-Architektur. Die orangefarbenen und schwarzen Verbindungen zeigen zwei separate Client/Server-Paare.

Architekturen verteilter Systeme



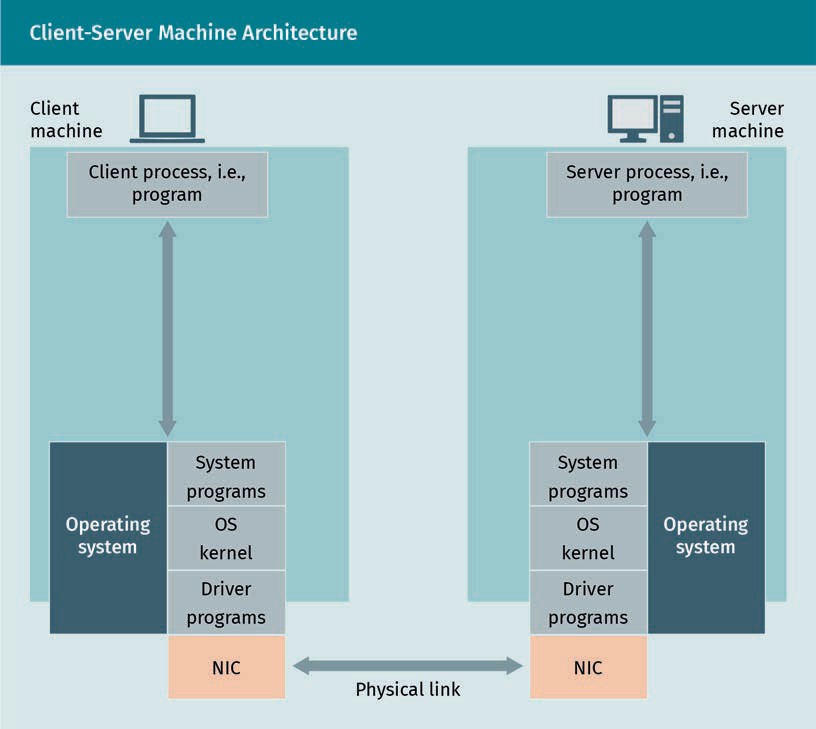
###### TCP/IP-Schichten in Client-Server-Architekturen

Da mit einer Client-Server-Architektur Internetdienste bereitgestellt werden, wird der TCP/IP-Protokollstapel verwendet. Einem Webdienst beispielsweise, der auf einer Client-Server-Architektur basiert, liegt das TCP/IP-Modell zugrunde. Die Anwendungsschichtentität für den Client ist als Clientanwendung (z. B. der Browser) implementiert, die Anwendungsschichtentität für den Server als Webserveranwendung (z. B. der Apache-Server). Da bei Client-Server-Architekturen sowohl verbindungslose als auch verbindungsorientierte Dienste bereitgestellt werden können, kann als Protokoll der Transportschicht sowohl TCP als auch UDP genutzt werden (Kurose & Keith, 2017). Als Protokoll der Internetschicht kann IPv4 oder IPv6 verwendet werden. Dabei sind die Protokolle von Transport- und Internetschicht als Systemprogramme des Betriebssystems implementiert. Die Sicherungsschicht ist softwareseitig als Systemprogramme und Treiber, hardwareseitig als Netzwerkkarte (Network Interface Card, NIC) implementiert.

Die folgende Abbildung zeigt die Architektur aus Client- und Servercomputern. Der Client- oder Serverprozess (also das Anwendungsprogramm) wird oberhalb der Betriebssystemebene installiert. Dies ist die Anwendungsschicht von Client oder Server. Einige Systemprogramme sind im Betriebssystem implementiert, wo die Aktivitäten der Transport- und Internetschicht von Client oder Server ausgeführt werden. Zusätzlich sind im Betriebssystem einige Treiberprogramme implementiert, die die softwareseitigen Aktivitäten der Sicherungsschicht von Client oder Server ausführen. Und über die Netzwerkkarte (NIC) werden die hardwareseitigen Aktivitäten der Sicherungsschicht von Client oder Server implementiert.

Apache-Server Ein Open-

Source-HTTP-Server für moderne Betriebssysteme (einschließlich UNIX und Windows).



###### Client-Server-Interaktion

Wie in der obigen Abbildung gezeigt, werden für die Interaktion zwischen Client und Server eine Reihe von Vorgängen ausgeführt. Diese Vorgänge sind als Funktionen, also als Systemaufrufe, implementiert. Nachfolgend sind diese Vorgänge näher beschrieben (Comer, 2015; Kurose & Keith, 2017; Tanenbaum & Wetherall, 2014):

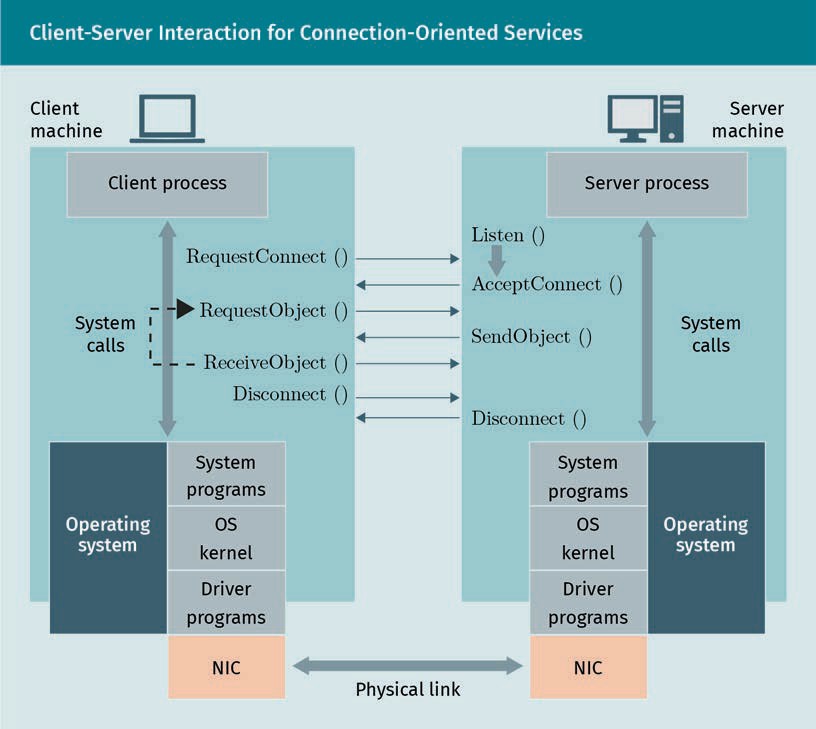
* Warten und lauschen. Dieser Vorgang findet nur auf Serverseite statt. Der Servercomputer und -prozess sind immer aktiv. Der Server wartet und lauscht auf Clientanforderungen. Zu diesem Zweck wird eine Funktion oder ein Systemaufruf wie Listen() implementiert.
* Anfordern der Verbindung. Der Client sendet eine Verbindungsanforderung an den Server. Dies erfolgt über einen Systemaufruf wie z. B. RequestConnect(). Für die Funktion RequestConnect()werden die Sockets von Client und Server als Argumente verwendet. Diese Sockets enthalten im Wesentlichen die IP-Adressen der Client- und Servercomputer sowie die Portnummern der Client- und Serverprozesse. Die Server-IP-Adresse kennzeichnet den Servercomputer, die Serverportnummer den Serverprozess. Für Webdienste wird Port 80 verwendet. Zur Identifizierung nutzt der Client bei Webdiensten zudem anfänglich den Hostnamen. Mithilfe des DNS-Protokolls wird basierend auf dem Hostnamen die IP-Adresse des Servers abgerufen.

Architekturen verteilter Systeme

* Akzeptieren der Verbindung. Sobald der Server die Verbindungsanforderung erhält, identifiziert er die angeforderte Anwendung anhand der Portnummer. Wenn der Dienst verfügbar und der Clientsocket gültig ist, führt der Server den Systemaufruf (z. B. AcceptConnect()) aus, um die Verbindungsanforderung zu akzeptieren und den Client zu benachrichtigen.
* Anfordern des Objekts. Nachdem die Verbindung zwischen Server und Client hergestellt wurde, wird eine weitere Funktion wie RequestObject()aufgerufen, um den gewünschten Dienst (also die Objektdatei oder die Daten) anzufordern. Der Objektbezeichner wird als Argument an den Systemaufruf übergeben. Bei Webdiensten ist der Objektbezeichner die URL der Zieldatei.
* Senden des Objekts. Nachdem die Objektanforderung empfangen wurde, überprüft der Server die Verfügbarkeit des Objekts. Ist das Objekt nicht verfügbar, sendet er eine Fehlermeldung. Anderenfalls sendet der Server das angeforderte Objekt. Dieser Vorgang ist ebenfalls durch einen Systemaufruf (z. B. SendObject()) implementiert. Mit dieser Funktion wird außerdem die Datengröße zurückgegeben.
* Empfangen des Objekts. Für den Empfang des Objekts auf Clientseite wird ebenfalls ein Systemaufruf verwendet (z. B. ReceiveObject()). Mit dieser Methode wird die belegte oder freie Puffergröße zurückgegeben.
* Trennen der Verbindung. Nachdem der Client die gesamte Datei empfangen hat, kann er erneut RequestObject() ausführen, um das nächste Objekt anzufordern. Wenn keine weiteren Objekte angefordert werden sollen, verwendet der Client den Systemaufruf Disconnect(), um den Server zu benachrichtigen, dass die Verbindung getrennt werden kann. Die Verbindung wird jedoch nicht sofort getrennt. Der Client wartet zunächst für eine bestimmte Dauer. Sobald der Server die disconnect()-Benachrichtigung empfängt, beendet er die Verbindung auf Serverseite, indem er ebenfalls Disconnect() aufruft.

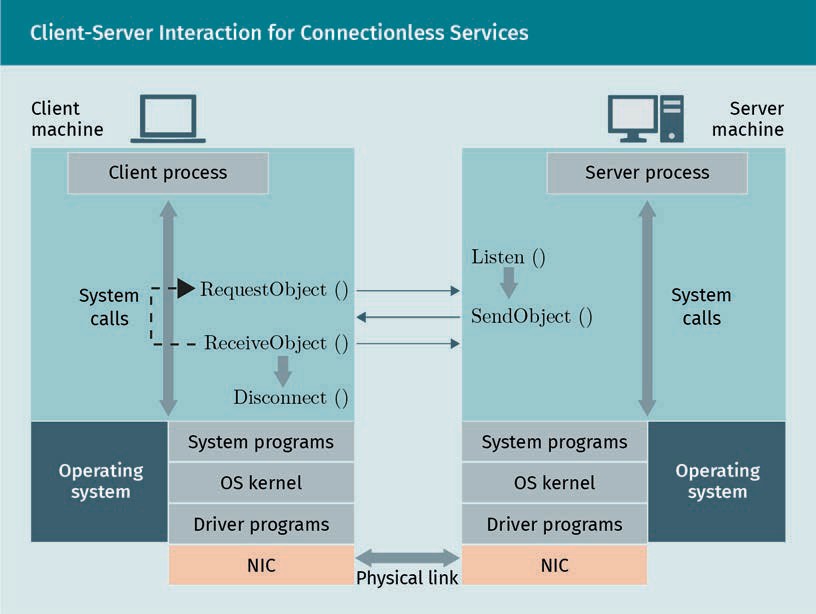
Bei einem verbindungslosen Dienst werden die Vorgänge zum Anfordern und Akzeptieren der Verbindung nicht ausgeführt. Auch sendet der Client keine Benachrichtigung zum Trennen der Verbindung. Stattdessen beendet der Client die Verbindung, ohne den Server zu benachrichtigen. Auf Serverseite wird die Verbindung beendet, sobald alle Daten übermittelt wurden. In der folgenden Abbildung ist die Client-Server-Interaktion bei verbindungslosen Diensten gezeigt (Kurose & Keith, 2017).

Nachfolgend sind die Schritte der TCP-basierten verbindungsorientierten Interaktion zwischen Client- und Servercomputern gezeigt.



In der folgenden Abbildung ist die Client-Server-Interaktion bei UDP-basierten verbindungslosen Diensten gezeigt.

Architekturen verteilter Systeme



Übung

Öffnen Sie die Wireshark-Software und wechseln Sie zum Erfassungspaket (Wireshark Foundation, ohne Jahresangabe). Öffnen Sie anschließend im Browser eine Website und wechseln Sie auf dieser Website für eine kurze Zeitdauer (ca. eine halbe Minute) zwischen verschiedenen Seiten. Filtern Sie die Pakete dann so, dass nur Pakete angezeigt werden, die zwischen Ihrem Computer und der Website ausgetauscht wurden, die Sie soeben besucht haben. Finden Sie heraus, welche Pakete für das Herstellen der Server-Client-Verbindung verantwortlich sind. Sie sollten die Pakete „Client Hello“ und „Server Hello“ finden. Suchen Sie anschließend nach der HTTP-Anforderungsnachricht, also nach der Get-Methode. Ermitteln Sie die HTTP-Antwortnachricht und untersuchen Sie alle oben erwähnten Nachrichten.

### Serviceorientierte Architekturen, Webdienste und Microservices

In einer serviceorientierten Architektur (Service-Oriented Architecture, SOA) kann der Server mit verschiedensten Anwendungen kompatibel sein, die in unterschiedlichen Programmiersprachen entwickelt wurden. Das Hauptziel einer SOA ist die Bereitstellung einer Struktur für lose miteinander verbundene Softwarekomponenten, um die vorhandene Software mit neu entwickelten Funktionen anzupassen, während gleichzeitig die Kosten minimiert werden (Gabhart & Bhattacharya, 2008).

###### Vorteile von serviceorientierten Architekturen

Bei IT-Technologien und in der Softwarebranche sind häufige und schnelle Änderungen an der Tagesordnung. Aus diesem Grund sollte eine Softwareinfrastruktur die erforderliche Flexibilität bieten, um problemlos neue Funktionen zu integrieren. Gleichzeitig sollten die Kosten für diese Anpassungen jedoch gering sein. Um diese beiden wichtigen Anforderungen zu erfüllen, wurde die serviceorientierte Architektur entwickelt. In einer serviceorientierten Architektur sind die Dienste in Schichten organisiert. Durch diese Abstraktion in Schichten können Organisationen vorhandene Ressourcen als Dienst bereitstellen und so neue Funktionen und Technologien integrieren. Da der Dienst nicht von Grund auf neu erstellt werden muss, lassen sich die neuen Funktionen schneller einführen. Außerdem wird die Verwaltung neuer Funktionen vereinfacht und die Implementierungskosten werden gesenkt. Nachfolgend sind einige Vorteile einer SOA aufgeführt (Endrei et al., 2004):

* + Nutzung vorhandener Ressourcen
  + einfachere Integration von neuen Funktionen als Erweiterung vorhandener Ressourcen
  + vereinfachte Verwaltung
  + schnellere Anpassung mit neuen Funktionen, Diensten oder Technologien
  + Möglichkeit der Wiederverwendung vorhandener Ressourcen
  + geringere Implementierungskosten

###### Entwicklung von SOA-Standards

Zur Verbesserung der Kommunikation zwischen Softwareprogrammen bei der verteilten Verarbeitung wurden in den 1970er Jahren eine Reihe von Standards entwickelt. Dazu zählen u. a. DCE/RPC (Distributed Computing Environment/Remote Procedure Calls) und COBRA (Common Object Request Broker Architecture) (Gabhart & Bhattacharya, 2008). DCE/RPC ist eine funktionsorientierte, CORBA eine objektorientierte Lösung. Mit diesen Standards können Anwendungen unabhängig davon miteinander kommunizieren, in welcher Programmiersprache sie erstellt oder unter welchem Betriebssystem sie ausgeführt werden. Dies wird als Interoperabilität bezeichnet. Interoperabilität ist eines der Hauptziele einer verteilten Architektur.

Die genannten Standards wurden von mehreren Anbietern entwickelt und gepflegt (Endrei et al., 2004; Gabhart & Bhattacharya, 2008). Da sie jedoch recht komplex waren und die Kommunikation zwischen den Anbietern nicht reibungslos lief, erwies sich die Weiterentwicklung dieser Standards als schwierig. Dies wiederum führte dazu, dass die Standards auch von anderen Anbietern nicht problemlos implementiert werden konnten. Später wurde das SOAP-Protokoll (Simple Object Access Protocol) von Microsoft entwickelt, das deutlich weniger komplex ist als DCE/RPC und CORBA. Zudem wird SOAP nur von einem Anbieter gepflegt. Für ein erweitertes verteiltes Modell sind jedoch eine Vielzahl zusätzlicher Funktionen erforderlich. Dazu zählen z. B. Verschlüsselung, Authentifizierung, Sicherheit sowie Dienst- und Transaktionsverwaltung. Ein entscheidender Vorteil von SOAP ist, dass die vorhandenen Funktionen problemlos um neue Funktionen erweitert werden können. In einer SOA sind Standards in Schichten implementiert, wie in der folgenden Abbildung gezeigt (Gabhart & Bhattacharya, 2008).

|  |  |
| --- | --- |
| SOA-Dienstschichten und -Standards | |
| Dienstschichten | Beispielstandards |

Architekturen verteilter Systeme

|  |  |
| --- | --- |
| SOA-Dienstschichten und -Standards | |
| Dienstverwaltung | WS Distributed Management |
| Orchestrierung von Geschäftsprozessen | WSBPEL, WS-BPEL Extension |
| Sicherheit Authentifizierung Autorisierung Verschlüsselung | Web Services Security WS Federation Language  WS-Security Kerberos Binding |
| Transaktionsverwaltung | Web Services Coordination |
| Garantierte Nachrichtenübermittlung | Web Services Reliable Messaging |
| Erweitertes Messaging Asynchrone Benachrichtigungen Anfügen von Dateien an Nachrichten | Web Services Notiﬁcation Web Services Addressing |
| Grundlegendes Messaging SOAP | SOAP 1.2, HTTP usw. |

###### Webdienste

Webdienstarchitekturen sind heute so beliebt, weil sie von Haus aus verteilt sind und heterogene Anwendungen über das Internet unterstützen können. Der Begriff „heterogen“ bedeutet in diesem Kontext, dass keine Abhängigkeit von einer Programmiersprache, einem Betriebssystem oder einer Hardwarespezifikation besteht. Außerdem sind Consumer und Provider lose miteinander verbunden. Auch diese Architekturen basieren auf offenen Standards wie eXtensible Markup Language (XML), SOAP, Universal Description, Discovery and Integration (UDDI) sowie Web Services Description Language (WSDL) (Tanenbaum & Wetherall, 2014). Die Verwendung offener Standards ermöglicht ein hohes Maß an Interoperabilität, sodass die Lösungen verschiedener Anbieter miteinander eingesetzt werden können. Darüber hinaus ist es so möglich, Webdienste zu entwickeln, ohne die Consumer-Lösungen im Detail zu kennen. Gleichermaßen sind für Consumer-Lösungen keine Detailinformationen zu den Webdiensten erforderlich. Nachfolgend sind einige der wichtigsten Merkmale von Webdiensten aufgeführt (Endrei et al., 2004):

* eigenständig. Eine einfache in XML oder HTTP entwickelte Clientanwendung sollte ausreichen. Es ist keine weitere Anwendung erforderlich. Serverseitig sind eine einfache Webserveranwendung und eine Servlet-Engine ausreichend.
* selbstbeschreibend. Client- und Serveranwendungen müssen lediglich das Format und den Inhalt der Anforderungs- und Antwortnachricht kennen.
* modular. Webdienste werden unter Verwendung von J2EE, CORBA, DCOM und anderen Technologien über das Internet bereitgestellt, um Heterogenität sicherzustellen.
  + sprachunabhängig und interoperabel. Client-Server-Anwendungen sind unabhängig von der jeweiligen Programmiersprache kompatibel.
  + basieren auf offenen Standards. Beispiele für diese Standards sind XML, SOAP, UDDI und WSDL.
  + zusammensetzbar. Ein einfacher Webdienst kann als Grundlage dienen, um komplexe Dienste zusammenzusetzen.
  + dynamisch. Webdienste sollten dynamisch sein.

###### Microservices

Microservice WireMock ist ein Simulationsserver zum Testen von Microservices.

Microservices sind eine neue Art von verteilter Softwarearchitektur. In einer Microservice-Architektur werde Softwareanwendungen als Sammlung kleiner unabhängiger Dienste entwickelt, wobei jeder Dienst einen eigenen unabhängigen Prozess ausführt. Der Begriff „klein“ ist dabei nicht streng definiert. Die Prozesse sind heterogen, können also in unterschiedlichen Sprachen entwickelt werden und individuelle Datenstrukturen oder eine separate zentralisierte Verwaltung aufweisen (Baresi & Garigga, 2020).

Eine Microservice-Architektur ist das Gegenteil einer monolithischen Architektur, in der Anwendungen mit umfangreichem Programmiercode geschrieben werden (Namiot & Sneps-Sneppe, 2014). Monolithische Architekturen gehen mit einer Reihe von Nachteilen einher. Beispiele sind eine geringere Produktivität, eine zeitaufwendige Anpassung mit neuen Diensten oder Probleme bei der Integration neuer Funktionen. Außerdem ist es nicht möglich, dass Entwickler und Anbieter unabhängig voneinander arbeiten. Auch die fehlende Skalierbarkeit und Interoperabilität müssen in diesem Zusammenhang genannt werden. Microservice-Architekturen wurden entwickelt, um die oben genannten Einschränkungen zu überwinden (Dragoni et al., 2017).

Service-Mesh-Architekturen

Das Service-Mesh ist eine Infrastrukturschicht, über die bei Microservices die Kommunikation zwischen den Diensten ermöglicht wird. Zu diesem Zweck wird ein Proxy eingesetzt (Khatri & Khatri, 2020). Moderne Softwareanwendungen für ein verteiltes Computing werden in mehrere kleine Komponenten (Dienste) unterteilt, die gemeinsam ein Netzwerk aus Microservices bilden. Dabei führt jeder Dienst eine dedizierte Geschäftsfunktion aus. Für die Ausführung eines Diensts werden gegebenenfalls Daten mit anderen Diensten geteilt oder ausgetauscht. Bei einer zu großen Anzahl von Anforderungen anderer Dienste kann es jedoch zu einer Überlastung kommen. Mit einem Service-Mesh wird eine Infrastruktur zum Übergeben der Anforderungen zwischen den Diensten bereitgestellt. Das Service-Mesh ist als Array aus Netzwerkproxys Teil der Anwendungssoftware. Wenn ein Benutzer eine Webseite anfordert, wird die Anforderung an einen Proxyserver geleitet. Ist das angeforderte Objekt für den Proxy nicht verfügbar, wird die Anforderung an den Server weitergeleitet. Der Server sendet das Objekt an den Proxyserver, der es wiederum an den Benutzer weiterleitet. Wenn ein weiterer Benutzer das Objekt anfordert, wird es direkt vom Webproxy gesendet. In einem Service-Mesh werden die Anforderungen zwischen Microservices über Proxys geleitet. Da die Microservices parallel ausgeführt werden, werden sie als Sidecars (Beiwagen) bezeichnet (Sharma & Singh, 2020). Ohne das Sidecar-Konzept muss jeder Microservice separat für die Kommunikation zwischen den Diensten programmiert werden.

Architekturen verteilter Systeme

Übung

Docker ist ein beliebtes Open-Source-Tool für die Bereitstellung von Diensten innerhalb von virtualisierten Containern. Dies ermöglicht eine problemlose Entwicklung, Bereitstellung und Ausführung von Softwareanwendungen. Mithilfe von Virtualisierungsverfahren bietet Docker die Möglichkeit, Anwendungsprogramme von der Infrastruktur zu trennen. Außerdem können Benutzer die Infrastruktur auf die gleiche Weise verwalten wie Anwendungen. Sehen Sie sich zunächst die offizielle Docker-Website an, um sich mit diesem Tool vertraut zu machen. Befolgen Sie anschließend die Anweisungen auf der Website, um das Docker-System herunterzuladen und zu installieren. Befolgen Sie dann die Anweisungen im Tutorial auf der Docker-Website, um eine Testanwendung auszuführen (Docker, ohne Jahresangabe).

### Edge Computing und Cloud Computing

Cloud Computing und Edge Computing sind erweiterte verteilte Computing-Architekturen. Sie basieren auf Client-Server-Architekturen und werden als erweiterte serviceorientierte Architekturen betrachtet. Das primäre Ziel beim Cloud Computing ist die On-Demand-Bereitstellung von IT-Ressourcen über das Internet. Die „Cloud“ ist dabei ein Remotespeicher oder eine Computing-Plattform, der bzw. die von einem Drittanbieter bereitgestellt wird. Der Zugriff erfolgt remote über das Internet. Im Wesentlichen handelt es sich also um Remote-Datenbankcenter zur Bereitstellung verschiedener Dienste über das Internet. Wenngleich die Cloud logisch betrachtet eine zentralisierte Infrastruktur ist, ist sie in der Praxis auf eine Gruppe oder einen Cluster aus Servern verteilt, die als Datacenter bezeichnet werden (Ward & Barker, 2013). Über eine Cloud können unterschiedliche Arten von Diensten für mehrere Clients bereitgestellt werden. Beispiele für diese Dienste sind eine große Speicherkapazität, eine schnelle und komplexe Verarbeitung oder Verarbeitungsplattformen, für die auf dem vorhandenen Computer keine Hardware- oder Softwareressourcen bereitgestellt werden müssen. Cloudserver befinden sich üblicherweise im Zentrum des Netzwerks (Core), die Clients am Netzwerkrand (Edge). Aufgrund der großen Entfernung bei der Kommunikation zwischen Cloud und Clients (den Edge-Geräten) eignet sich eine Cloud daher gegebenenfalls nicht für eine Datenverarbeitung und Analyse in Echtzeit. Eine Möglichkeit, um diese Einschränkung zu umgehen, besteht darin, einige Datacenter in der Nähe der Edge-Geräte aufzubauen. So kann die zeitkritische Datenverarbeitung in die Edge-Datacenter verlagert werden, während die weniger zeitkritische Datenverarbeitung in Remote-Clouds oder Datacenters im Netzwerk-Core erfolgt. Bei dieser erweiterten Art von Architektur spricht man von Edge Computing (Khan et al., 2019).

###### Eigenschaften der Cloud

Für die Konzepte von Cloud- und Cloud Computing-Systemen finden sich in verschiedenen Quellen unterschiedliche Beschreibungen. Nachfolgend sind die Hauptmerkmale von Cloud Computing-Systemen aufgeführt (Ward & Barker, 2013):

* On-Demand-Bereitstellung. Bei einem Cloud Computing-System sollten die Dienste nach Bedarf (On-Demand) für autorisierte Benutzer bereitgestellt werden können.
* umfangreiche Zugriffsmöglichkeiten. Der Zugriff auf die Cloud sollte von mehreren Standorten aus und über verschiedene Plattformen (z. B. Desktop-PCs, Laptops, Smartphones, Windows-, MAC- oder Android-Betriebssysteme) möglich sein.
* Ressourcenpools. Der Cloudanbieter sollte in der Lage sein, Ressourcen in Pools zusammenzufassen, die von mehreren Kunden oder Consumern mit verschiedenen Hardware- oder Softwarespezifikationen gemeinsam verwendet werden können.
* Elastizität. Der Anbieter sollte die bereitgestellten Dienste basierend auf den Anforderungen der Benutzer problemlos skalieren können.
* Erfassung der Dienstnutzung. Der Cloudanbieter sollte überwachen und messen können, welche Dienste und Ressourcen für die einzelnen Consumer bereitgestellt wurden.

###### Clouddienste

Über Cloud Computing-Systeme können die verschiedensten Dienste bereitgestellt werden. Diese lassen sich üblicherweise in die nachfolgenden Kategorien einordnen (Ward & Barker, 2013; Rimal et al., 2009):

* + Software-as-a-Service (SaaS) bezieht sich auf Endbenutzer-Anwendungen, die vom Dienstanbieter bereitgestellt und verwaltet werden. Die Endbenutzer müssen sich keine Gedanken über die Infrastruktur, die Plattform oder die zugrunde liegenden Wartungs- und Verwaltungsaufgaben der Anwendungen oder Softwaredienste machen. Sie nutzen ganz einfach das fertige Produkt, also die Software, für die sie sich interessieren. Ein Beispiel für einen solchen Dienst ist ein webbasierter E-Mail-Dienst eines bestimmten Unternehmens. Consumer können über verschiedene Browser, Betriebssysteme und Geräte auf die E-Mails zugreifen, ohne die zugrunde liegende Infrastruktur des E-Mail-Webdiensts zu kennen.
  + Platform-as-a-Service (PaaS) ermöglicht einen Remotezugriff auf die gewünschten Plattformen (z. B. Betriebssysteme oder Hardware). Benutzer können Anwendungen auf der bevorzugten Plattform installieren. Ein Beispiel für ein beliebtes Betriebssystem ist Kali Linux. Dieses System umfasst eine Vielzahl von Softwaretools und Tools für die digitale Forensik, mit denen sich die Netzwerksicherheit testen lässt. Aufgrund der IT-Sicherheitsrichtlinien einer Organisation oder zu hoher Bereitstellungskosten ist dieses Betriebssystem jedoch nicht immer eine geeignete Lösung. Eine Reihe von Cloudanbietern bieten

cloudbasierte virtuelle Computer und Betriebssysteme mit unterschiedlichen Spezifikationen an.

* + Infrastructure-as-a-Service (IaaS) bietet Clouddienste mit dem höchsten Maß an Flexibilität. Diese Dienste umfassen grundlegende Netzwerkeinrichtungen, Hardwarevirtualisierung und Speicherdienste.

###### Cloudtypen

Bei Clouds wird üblicherweise zwischen drei Typen unterschieden: private Cloud, öffentliche Cloud und Hybrid Cloud (Rimal et al., 2009). Bei einer privaten Cloud werden IT-Ressourcen über das Internet bereitgestellt. Diese Art von Cloud ist auf eine einzelne Organisation beschränkt. Sie kann innerhalb der Organisation oder von einem Drittanbieter bereitgestellt und verwaltet werden. Bei einer öffentlichen Cloud werden die Clouddienste für mehrere Organisationen bereitgestellt, sodass die Ressourcen von mehreren Consumern gemeinsam genutzt werden können. Folglich müssen bei öffentlichen Clouds umfangreichere Sicherheitsaspekte und -risiken berücksichtigt werden als bei privaten Clouds. Eine Hybrid Cloud-Infrastruktur ist eine Kombination aus einer öffentlichen und einer privaten Cloudinfrastruktur. Einige Ressourcen und Dienste werden also dediziert für eine einzelne Organisation bereitgestellt, während andere Ressourcen und Dienste von mehreren Organisationen gemeinsam verwendet werden.

###### Edge Computing

Das Edge Computing ist eine verteilte Computing-Architektur, die als Grundlage für beliebte moderne Technologien wie das Internet der Dinge (Internet of Things, IoT), 5G und die Fahrzeug-zu-Fahrzeug-Kommunikation dient. Das Hauptziel beim Edge Computing sind geringe Latenzzeiten, während gleichzeitig Mobilität und Location Awareness (Standortbestimmung) für die Echtzeitanwendung sichergestellt sind. In herkömmlichen Architekturen verfügen die Endknoten (insbesondere mobile Knoten) über geringere Verarbeitungsgeschwindigkeiten sowie eine geringere Arbeitsspeicher- und Speicherkapazität (Khan et al., 2019). Core-Geräte wie Clouds und Datacenter hingegen bieten höhere Verarbeitungsgeschwindigkeiten sowie eine größere Arbeitsspeicher- und

Architekturen verteilter Systeme

Speicherkapazität. Aus diesem Grund werden auf den Endgeräten weniger rechenintensive Prozesse ausgeführt und in der Cloud die rechenintensiveren Daten verarbeitet. Für Echtzeitanwendungen oder verzögerungsempfindliche Anwendungen eignen sich diese Architekturen jedoch nicht (Khan et al., 2019). Denn würden verzögerungsempfindliche Daten auf mobilen Endgeräten verarbeitet, käme es aufgrund der eingeschränkten Verarbeitungs- und Speicherkapazität der Geräte zu hohen Latenzzeiten.

Eine Verarbeitung von verzögerungsempfindlichen Daten ist zudem nicht ratsam, da die Cloud sich im Netzwerk-Core befindet und die mobilen Geräte am Netzwerk-Edge. Die Entfernung zwischen Cloud und Endgerät kann also zu groß sein, sodass es aufgrund von Ausbreitungs-, Warteschlangen- und Verarbeitungsverzögerung an mehreren Hops zu hohen Latenzzeiten kommt. Dieses Problem lässt sich mit einer Edge Computing-Architektur lösen, in der die Cloud und das Datacenter an den Rand des Netzwerks verlagert werden. Die Entfernung zwischen Netzwerk-Edge und Endgerät ist so deutlich geringer. Und da dadurch auch die Latenzzeiten verringert werden, ist das Edge Computing eine geeignete Lösung für verzögerungsempfindliche Anwendungen und mobile Knoten (Khan et al., 2019; Liu et al., 2019).

Merkmale des Edge Computing

Nachfolgend sind die wichtigsten Merkmale beim Edge Computing aufgeführt (Khan et al., 2019):

* Nähe zum Benutzer. Das Edge Computing zeichnet sich insbesondere dadurch aus, dass die Clouddienste in möglichst geringer Entfernung zum Endbenutzer bereitgestellt werden. Zu diesem Zweck werden die Cloud Computing-Dienste über kleinere Datacenters oder Clouds am Netzwerk-Edge bereitgestellt, die mit hoher Dichte aufgebaut sind.
* Unterstützung von Mobilität und Location Awareness. Das Edge Computing ermöglicht eine Echtzeitverarbeitung für mobile Knoten. Für die Unterstützung von Mobilität wird LISP (Locator/ID Separation Protocol) eingesetzt. Bei diesem Protokoll werden zwei Nummern verwendet, eine zur Identifizierung des Netzwerklocators, eine zur Identifizierung des Knotens. Dabei kann es sich um IP-Adressen, MAC-Adressen oder GPS-Koordinaten handeln.
* geringe Latenzzeiten. Indem Clouds in der Nähe der Endbenutzer bereitgestellt werden, wird die physische Entfernung zwischen Cloudservern und Endgeräten verringert. Da auch die Anzahl an Hops sinkt, werden gleichzeitig die Ausbreitungsverzögerung, die routerbasierte Verarbeitungsverzögerung und die Warteschlangenverzögerung reduziert. Insgesamt lassen sich also geringere Latenzzeiten erreichen.
* Kontextsensitivität. Um kontextbezogene Dienste für Endknoten bereitzustellen, werden Netzwerklast, Benutzerstandort und weitere Echtzeit-Netzwerkinformationen berücksichtigt. Mithilfe dieser Kontextinformationen kann die Benutzerzufriedenheit und damit die Dienstqualität verbessert werden.
* Heterogenität. Beim Edge Computing sollten verschiedene Softwareanwendungen, Plattformen, Betriebssysteme, Hardwarespezifikationen und Infrastrukturen unterstützt werden können, die von unterschiedlichen Anbietern entwickelt wurden.

Varianten des Edge Computing

Die Edge Computing-Technologie wurde in den vergangenen Jahren umfassend untersucht. Basierend auf der Infrastruktur und dem Diensttyp wurden verschiedene Edge Computing-Varianten entwickelt. Nachfolgend sind einige der wichtigsten Varianten aufgeführt (Khan et al., 2019):

* + Das Mobile Edge Computing wurde entwickelt, um Dienste für mobile Endpunkte bereitzustellen. Diese Variante ist auch unter dem Namen Multi-Access Edge Computing (MEC) bekannt. Dabei werden Clouds an den Basisstationen von Mobilfunknetzen bereitgestellt.
  + Das Fog Computing ist eine Edge Computing-Cloud, die mit einer Vielzahl von Sensoren (Fog) verbunden werden kann. Durch diese Sensoren werden gegebenenfalls immer wieder Daten generiert und weitergeleitet, was zu einer enormen Menge an Daten und damit zu einer Überlastung der Edge-Cloud führen kann. Aus diesem Grund werden die Daten zunächst von Computinggeräten verarbeitet, die sich in unmittelbarer Nähe zu den Geräten befinden, die die Daten generieren. Indem anstelle von Rohdaten bereits verarbeitete Daten an die Cloud weitergeleitet werden, wird die Verarbeitungsleistung der Edge-Cloud optimiert.
  + Ein Cloudlet ist ein kleines Cloud-Datacenter, das zwischen mobilen Endgeräten und der Cloud im Netzwerk-Core bereitgestellt wird. Hauptziel eines Cloudlets ist die Minimierung von Latenzzeiten für mobile Anwendungen.

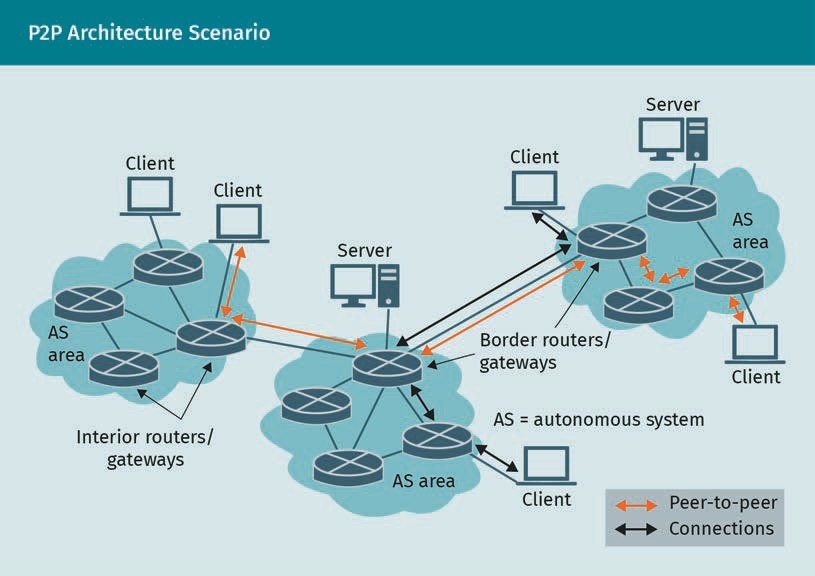
In den vergangenen Jahren haben eine Reihe von Unternehmen Cloud Computing- und Edge Computing-Plattformen, -Tools und -Produkte entwickelt und auf den Markt gebracht. Einige beliebte Edge Computing-Produkte sind Cloudpath, pCloud, ParaDrop, SpanEdge, AWS IoT Greengrass, Azure IoT Edge und Cloud IoT Edge. Eine detaillierte Untersuchung von Edge Computing-Tools finden Sie bei Liu et al. (2019).

### Peer-to-Peer-Computing

In einer Client-Server-Architektur ist der Server immer aktiv, während der Client sowohl aktiv als auch inaktiv sein kann. Außerdem fordert der Client Objekte wie Dateien oder Daten nur an, der Server stellt das angeforderte Objekt bereit. In einer Peer-to-Peer-Architektur (P2P) gibt es keine dedizierten Server und kein Knoten oder Peer ist immer aktiv. Knoten bzw. Peers können sich über mehrere Verbindungspunkte miteinander verbinden, ohne dass zu diesem Zweck ein Server benötigt wird (Kurose & Keith, 2017). Auf jedem Peer wird eine gemeinsame P2P-Anwendung ausgeführt und jeder dieser Peers gibt als Teil des Netzwerks Objekte oder Dateien frei. Für die Suche nach Objekten können Peers die API (Application User Interface, Anwendungsschnittstelle) verwenden. Die API gibt eine Liste der verfügbaren Objekte aus verschiedenen Quellen (Peers) zurück. Der Peer hat also die Möglichkeit, das Objekt aus einer beliebigen verfügbaren Quelle herunterzuladen. Zwei gängige P2P-Anwendungen für die Dateifreigabe sind BitTorrent und µ. Auch Skype und andere VoIP-Dienste (Voice over IP) nutzen eine P2P-Architektur (Tanenbaum & Wetherall, 2014).

In der folgenden Abbildung ist ein Szenario mit P2P-Architektur gezeigt. Die orangefarbenen und schwarzen Linien stehen für zwei separate Peer-to-Peer-Verbindungen.

Architekturen verteilter Systeme



###### BitTorrent

BitTorrent ist ein P2P-Protokoll für die Verteilung von Dateien. Dabei teilen alle Peers innerhalb eines Netzwerks Inhalte mit den anderen Peers. Die Peers können Dateien von anderen Peers herunterladen bzw. auf andere Peers hochladen. Jeder Peer verfügt über ein Verzeichnis, den sogenannten „Torrent“, in dem alle verfügbaren Peers und ihre Inhalte sowie der eigene Standort aufgeführt sind. Anfänglich ist dieses Verzeichnis leer. Da jeder Peer jedoch die Metadateninformationen seiner geteilten Dateien in den Torrent hochlädt, enthält das Verzeichnis schließlich eine Liste sämtlicher geteilter Dateien aller Peers. Es sind jedoch nicht die vollständigen Inhalte der Dateien, sondern lediglich die Metadateninformationen enthalten. Die Peers des P2P-Netzwerks nutzen einen Tracker, um alle aktiven Peers nachzuverfolgen. Als aktive Peers werden alle Knoten betrachtet, die Daten hoch- oder herunterladen. Diese Knotengruppe wird als Schwarm bezeichnet (Tanenbaum & Wetherall, 2014). Wenn eine Datei von mehreren Peers geteilt wird, heißen diese Seeds (Tanenbaum & Wetherall, 2014). Der Tracker liefert die Informationen zu den Standorten der Seeds. Für eine bessere Funktionsweise und optimale Leistung sollte jeder Knoten Dateien teilen und Daten hochladen. Auf einigen Knoten werden Daten jedoch lediglich heruntergeladen. Diese Knoten laden keine Daten hoch oder beenden den Uploadvorgang, sobald die gewünschten Dateien heruntergeladen wurden. Diese Knoten werden als Leecher bezeichnet. Wenn ein Peer keine Daten hochlädt, vermeiden die anderen Peers, Inhalte mit diesem Leecher zu teilen (Tanenbaum & Wetherall, 2014).

Wenn nur ein einziger Tracker vorhanden ist, handelt es sich um ein zentralisiertes System, das anfällig ist für eine Überlastung oder Einzelfehlerstellen. Eine verteilte Hashtabelle (Distributed Hash Table, DHT) ist ein P2P-Verfahren, mit dem sich mehrere Tracker implementieren lassen, um für eine dezentralisierte Bereitstellung der Netzwerkdienste zu sorgen (Tanenbaum & Wetherall, 2014).

###### Leistungsvergleich von P2P- und Client-Server-Architekturen

Angenommen, ein Netzwerk verfügt über N Knoten. Die Uploadkapazität der Knoten ist A1, A2, A3, …, An − 1, An, die Downloadkapazität ’1, ’2, ’3, …, ’n − 1 , ’n und die Mindestdownloadkapazität ’71n. Angenommen, alle Knoten müssen eine Datei mit der Größe F herunterladen. Die Datei kann entweder über eine Client-Server-Architektur oder über ein P2P-Netzwerk verteilt werden (Kurose & Keith, 2017).

Bei einer Client-Server-Architektur wird die Datei vom Server verteilt. Bei einer Uploadrate des Servers von As ist die erforderliche Dauer für das Hochladen der Datei durch den Server G /As. Um die Datei an N Clients zu verteilen, muss sie N Mal hochgeladen werden. Die Uploaddauer ist also 2G /As. Beim 1. Knoten beträgt die Downloadzeit G /’1. Die maximale Downloadzeit ist G /’71n. Folglich ist die Dauer für die Verteilung der Datei an alle Knoten 78= 2G /As, G /’71n (Kurose & Keith, 2017).

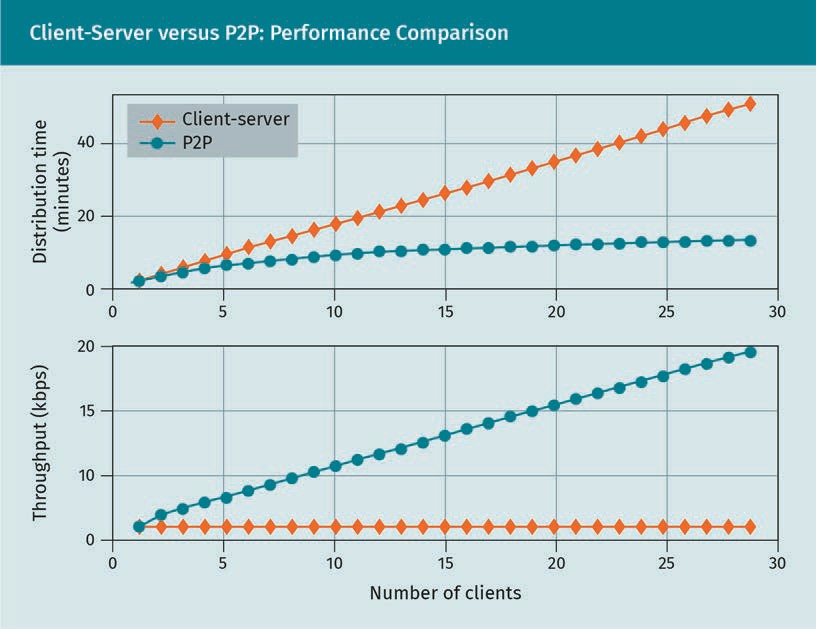
Angenommen, in einer P2P-Architektur verfügt anfänglich keiner der Knoten über die Datei. Zunächst lädt also einer der Knoten die Datei vom Server herunter. Die Uploaddauer des Servers ist G /As. Nun haben die übrigen Knoten zwei Möglichkeiten: sie können die Datei vom Server oder vom ersten Knoten herunterladen. Sobald der zweite Knoten die Datei heruntergeladen hat, verfügen die übrigen Knoten über drei Möglichkeiten zum Herunterladen der Datei: sie können sie vom Server, vom ersten Knoten oder vom zweiten Knoten herunterladen. Die Dauer für die Verteilung der Datei an alle Knoten ist bei der P2P-Architektur also

78= 2G /As, G /’71n, G / As + ∑n A1 (Kurose & Keith, 2017). In der folgenden Abbildung werden die Dauer für die Dateiverteilung und der Durchsatz bei einer Client-Server-Architektur und einer P2P-Architektur verglichen.

1 = 0

Grundlage dieser Darstellung sind die folgenden Parameter: eine Dateigröße von 5 MB, eine Uploadrate der Clients von 5 KBit/s, eine Uploadrate des Servers von 50 KBit/s und eine Downloadrate der Clients von 50 KBit/s. Die Abbildung zeigt, dass bei einer steigenden Anzahl an Knoten (oder Peers) beim P2P-Modell eine höhere Leistung erzielt wird als beim Client-Server-Modell.

Architekturen verteilter Systeme



###### Hybrid-Architektur

Client-Server-Architekturen und P2P-Netzwerke sind die gängigsten verteilten Architekturen, die aktuell verwendet werden. Es gibt jedoch auch eine Vielzahl von Hybrid-Architekturen, in denen das Client-Server- und das P2P-Modell miteinander kombiniert werden. Ein gutes Beispiel sind Messenger-Anwendungen (Kurose & Keith, 2017). Bei diesen Anwendungen verwaltet der Messenger-Server die Liste und Standorte der Benutzer. Sobald die Benutzer sich verbinden, tauschen sie Informationen basierend auf der P2P-Architektur aus.

Zusammenfassung

Verteilte Computing-Architekturen sind in verschiedenen Quellen unterschiedlich definiert. Im Wesentlichen verfolgen jedoch alle diese Architekturen das Ziel, mehrere Prozesse auf verschiedenen Plattformen und Geräten auszuführen, um ein gemeinsames Ziel zu erreichen. Diese Prozesse, Plattformen und Geräte können homogen oder heterogen sein. Ein gutes Beispiel für das verteilte Computing ist das Internet. Die beiden gängigsten verteilten Architekturen für das Internet sind Client-Server-Architekturen und P2P-Netzwerke. In einer Client-Server-Architektur fordert ein Computer (der Client) Verbindungen und Dienste (Objekte wie Dateien oder Daten) vom Servercomputer an. Bei einer P2P-Architektur hingegen wird kein Servercomputer benötigt. Stattdessen kommunizieren alle Knoten direkt über Verbindungspunkte miteinander. Die Knoten eines P2P-Netzwerks heißen Peers. Diese Peers sind

über ein gemeinsames Anwendungsprogramm miteinander verbunden, das zum Teilen, Hochladen und Herunterladen von Objekten verwendet wird. In einer einfachen Client-Server-Architektur sind teilbare Ressourcen als Objekte gekapselt. Eine serviceorientierte Architektur (SOA) ist eine erweiterte Client-Server-Architektur, in der teilbare Ressourcen als Dienste gekapselt sind, um die Interoperabilität und Heterogenität von Anwendungen, Plattformen und Infrastrukturen zu verbessern. Eine erweiterte SOA-Version ist eine Microservice-Architektur, in der Anwendungen als eine Sammlung aus lose miteinander verbundenen Diensten implementiert sind. Das Cloud Computing ist eine weitere Variante einer Client-Server-Architektur, bei der über eine Cloud (ein Cluster aus Datacenters), die sich im Zentrum des Netzwerks befindet, Speicher-, Plattform-, Anwendungs- und Datenverarbeitungsdienste für Benutzer bereitgestellt werden. Auch beim Cloud Computing gibt es erweiterte Varianten, z. B. das Edge Computing, bei dem eine kleine Cloud am Netzwerk-Edge implementiert wird. Durch die geringere Entfernung zu Benutzern lässt sich mit dieser Variante eine hohe Dienstqualität für verzögerungsempfindliche Dienste sowie Benutzermobilität sicherstellen.



# Lektion 5

## Distributed Algorithms and Applications

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… understand the concept of communication and synchronization of distributed systems.

… analyze the concept of transactions and data management.

… apply the aspects for distributed services and applications.

… design and implement distributed algorithms.

DL-E-DLMCSNDS01-U05

1. Distributed Algorithms and Applications

### Einführung

System call A computer pro- gramming term, which means requesting a func- tion or system pro- gram of an operating

system.

Unlike a centralized system, the programs of a distributed system are installed on dif- ferent geographical locations, machines, and operating systems. Hence, synchronizing the programs across different places and systems is crucial. In a centralized system, all programs are located on a single machine or operating system, meaning that synchro- nization is simply done by following the clock of the system. However, because of dif- ferences in locations and machines, the clock reading may differ from machine to machine in a distributed system. Also, inter-process communication is a big challenge in a distributed system because the callee and caller functions of a system call are located on different machines. A widely-used beneﬁt of a distributed system is that data or ﬁles can be replicated on multiple storages. However, it introduces a new chal- lenge of consistency, which means once a piece of data has been updated, all other replicas have to be updated accordingly. Hence, users or processes on various machines would read the same, i.e., consistent, information. Multiple processes of a distributed system may share a common data or memory. Therefore, distributed algo- rithms must be designed in a way that interrelated processes do not access the resource simultaneously in order to maintain the preferred order of execution, which means the algorithms should be designed to ensure that distributed processes are mutually exclusive. This unit aims to provide an insight to the topics of a distributed system, such as inter-process communication and synchronization, distributed algo- rithms for mutual exclusion, and data consistency of a distributed transaction. Addi- tionally, this unit sheds light on the security issues of a distributed system.

### Communication and Synchronization in Distributed Systems

In a centralized system, multiple programs (i.e., processes) are installed on a common system. The processes can communicate with each other through shared data and memory. Unlike a centralized system, the programs or processes of a distributed system are installed in different systems, meaning communication through shared memory is not possible. Communication generally takes place through exchanging messages between the processes. Modern distributed systems can be immensely large and com- plex, making it difﬁcult to implement inter-process communication through a simple message-passing technique. In this section, we will discuss the widely used communi- cation models for large-scale distributed systems. Furthermore, in a distributed system, processes can share resources. When a resource is shared among multiple processes, they should be synchronized in order to protect the resource from unwanted or acci- dental access. This section will also discuss various process-synchronization techni- ques for distributed systems.

Distributed Algorithms and Applications

###### Inter-Process Communication in Distributed Systems

Two widely used communication models for distributed systems are remote procedure call (RPC) and message-oriented middleware (MOM) (van Steen & Tanenbaum, 2017).

Remote procedure call (RPC)

In RPC, when a process on machine A calls a process on machine B, the caller process on machine A is suspended and the called process executes on machine B. Messages can be passed from the caller to the called process and vice-versa as function argu- ments and returns (Birrell & Nelson, 1984). Though the idea of RPC is simple and straightforward, the implementation is interactive, considering the different address spaces of the machines, speciﬁcation differences, data representation differences, availability of the machine and connection, and parameter passing challenges.

The RPC is known as a request-response protocol, like the HTTP client-server protocol. The client machine initiates the RPC call by sending a request message to the server machine. The request message carries the parameters that are to be passed to the called procedure. It is notable that the word “procedure” means subroutine, i.e., a piece of program. Some key terminologies of RPC are discussed below (van Steen & Tanen- baum, 2017):

* A “stub” is a piece of program that converts passed parameters during an RPC call. In a distributed system, the client and server procedures or functions are located in different machines. Hence, during an RPC call the passed parameters need to be converted. This conversion can be for various reasons. For example, the pointer value of one machine is different from another machine. Also, if the data formatting is different on a client-server machine, then the passed parameter has to be conver- ted.
* A “client stub” is the client-side stub. The client-stub encapsulates a message before transferring to the server.
* A “server stub” is the server-side stub. When the server receives a message, it passes through the server stub.
* “Marshalling” is the process of encapsulating a parameter. The purpose of marshal- ling is to represent data in a neutralized format. For example, some computers (e.g., Intel Pentium) write data in each byte from right to left, called little endian, while some other computers (e.g., older advanced RISC machines (ARM) processors) write left to right, called big endian. Hence, during RPC the data must be represented in a neutral format that is translateable to all types of machines. The goal of marshalling is to represent the parameter values in a neutral format.
* “Unmarshalling” is intended to convert data from a neutralized format to the desired format.

The six steps of RPC are as follows (van Steen & Tanenbaum, 2017):

Little endian

This is an order in which the least sig- niﬁcant value is stored in the lowest address space.

Big endian

An order in which the most signiﬁcant value is stored in the lowest address space.

* + 1. The client procedure calls the client-stub.
    2. The client-stub performs marshalling i.e., encapsulates the parameter to form the message or packet and calls the client's operating system (OS) program for message transmission.
    3. The client's OS (via link and physical layer) transmits the packet from the client’s machine to the server’s machine.
    4. The server’s OS passes the received packet to the server-stub.
    5. The server’s stub performs unmarshalling i.e., decapsulates the received packet.
    6. The server procedure calls the server-stub and repeats the above-mentioned steps in reverse order to respond to the client.

The RPC protocol is standardized in the Request for Comments (RFC) 5531 (Thurlow, 2009).

Message-oriented middleware (MOM)

The way of communication in RPC is synchronous in nature. Hence, the client process remains blocked until a response is received from the server side. Some modiﬁcations are made to make RPC less synchronous in nature. The concept of message-oriented middleware (MOM) is developed to provide asynchronous communication between the server and client in a distributed system.

Middleware is a software which lies between the OS and application software on both the client side and server side (Etzkorn, 2017). Middleware aids to deploy applications over a distributed system by encapsulating the complexities of distributed applica- tions, encapsulating heterogeneity of hardware and operating systems, providing uni- form interfaces, and providing common services. These services allow us to build port- able and interoperable applications.

Generally, MOM provides persistent synchronous communication where the implemen- tation of RPC is intricated. However, MOM supports both synchronous and asynchro- nous modes of communication. It also supports both persistent and nonpersistent modes of communication.

In a synchronous communication, the sender is blocked until the message is received by the receiver or even until a response is received by the sender. This means that the sender can send its second message only after the response from the ﬁrst message is received. In an asynchronous communication, the sender can send the second message immediately after sending the ﬁrst. The second message can be sent before receiving the response from the ﬁrst message, and the second message can even be sent before the ﬁrst message has been delivered to the receiver.

###### Synchronization in Distributed Systems

In a centralized system, synchronization among processes is done by asking the time to the OS. Each process resides in the same OS, thus implementing the synchronization is simple and straightforward. In the case of distributed systems, processes reside in dif-

Distributed Algorithms and Applications

ferent OSs. Thus, there is no idea of a global time. Some systems are faster or slower compared to others. This makes synchronization challenging to implement in a distrib- uted system.

The processes on a distributed system can be synchronized in two ways: using physical clocks or logical clocks (van Steen & Tanenbaum, 2017).

Physical clock synchronization

Each computer has a physical clock or timer. It is basically a crystal that oscillates at a certain frequency based on the amount of voltage applied to it. Each oscillation is counted as a clock tick. However, not all timers provide the exact same time. This time difference is known as clock skew. To overcome the clock skews in a distributed system, an external global clock is needed. The external global clock provides what is known as universal coordinated time (UTC) (van Steen & Tanenbaum, 2017). At the beginning of each second, 40 shortwave radio stations around the world broadcast a short pulse to regulate the time. Also, there are multiple satellites that provide a UTC time-keeping service. The machines in a distributed system are synchronized with UTC. Several algo- rithms have been developed to resolve this synchronization problem, such as network time protocol (NTP), Berkeley algorithm, and the reference broadcast synchronization (RBS).

Network Time Protocol (NTP)

Network time protocol is a client-server model-based protocol for clock synchroniza- tion (van Steen & Tanenbaum, 2017). It operates on connectionless transport protocol, such as user datagram protocol (UDP) (Mills, 1991). In NTP, at time T0, a client requests a timestamp from the server. The server’s clock is synchronized with UTC. The server records the message receive time (T1), then responds to the client with a timestamp which contains the message receiver time (T1) and the message transmission time at server (T2). The client receives the server’s timestamp and records the receive time (T3). Based on this information, the client calculates the round trip-time and estimates the delay. From this estimation, the client minimizes the clock synchronization error (van Steen & Tanenbaum, 2017). The NTP is standardized in RFC 5905, 7822, and 8573 (Mills et al., 2010).

Berkeley algorithm

The server plays a passive role in NTP protocol. This means that in NTP, the server only receives requests from the client, and it responds to them accordingly. The Berkeley algorithm is an active protocol where the time server plays an active role. In this case, the time server is basically a system program that collects time from the clients repeat- edly after a certain period. Based on the received average time, the server suggests for the client to slow down or speed up to adjust the time (van Steen & Tanenbaum, 2017). This protocol is useful when the time server is not synchronized with UTC.

Reference broadcast synchronization (RBS)

RBS protocol is suitable for synchronizing time in wireless sensor networks (Elson et al., 2002). Like the Berkeley algorithm, a sender broadcasts a message considering that all receivers are at a ﬁrst hop distance. The receiver accepts the message and compares

the message construction time and message delivery time, and gives an estimate of the delay (van Steen & Tanenbaum, 2017). The delay contains the propagation time and the message processing time at the network interface card.

Logical clock synchronization

In a distributed system, not all nodes must necessarily agree on a time based on a physical clock. What is important is that the order of events that occur at various nodes be synchronized. This concept of synchronization excludes physical clocks and it is known as logical clock synchronization. Two widely-known logical clock synchronization protocols are Lamport’s logical clocks and vector clocks (van Steen & Tanenbaum, 2017).

Lamport’s logical clock

Lamport’s logical clock utilizes the term “happens before,” indicated by the arrow sym- bol “ ”. A 5 means event A occurs before event B (van Steen & Tanenbaum, 2017). Therefore, the logical time of A, 6 A will be smaller that logical time of 5, 6 5 *,* hence 6 A < 6 5 . Suppose event 6 occurs on another node, and 6 occurs after 5, then the time 6 6 > 6 5 > 6 A . Logical time always increases and never decreases (van Steen & Tanenbaum, 2017). If A 5 is true, then 6 5 = 6 A + n, with n indicating the number of events occurred between event A and B.

Vector clocks

Lamport’s logical clock does not account for causality. Suppose message 71 and 72 are received at node n1. Also, 72 arrived after 71, hence, 6 71 < 6 72 . This relation does not mean the event of receiving message 71 and 72 are related to each other. The message 71 and 72 can be sent by two different nodes that are independent from each other. The vector clocks algorithm considers causality, which means the clock time information also provides information about the relation between messages (van Steen & Tanenbaum, 2017). This is done simply by including some additional information with the clock time value. This information may include the process ID number or node ID number.

Übung

Write a program to simulate the classic “producer-consumer” concurrency problem. The code should have two threads named “producer” and “consumer.” Assume that the pro- ducer and consumer share a common buffer. Consider that the buffer size is 2. The producer generates data and writes them into the buffer, while the consumer removes data from the buffer. Declare a variable called “counter.” The producer increases coun- ter value by one once it writes data in the buffer, and the consumer decreases the counter value by one each time it removes a piece of data from the buffer. While the counter value is 2, the producer is not allowed to write in the buffer. The consumer is not able to remove a data from the buffer while counter value is zero. Santiago (2020) provides a Python-based tutorial related to concurrency problems.

Distributed Algorithms and Applications

### Distributed Algorithms

The key to a distributed system is concurrency and collaboration among multiple pro- cesses running on various machines. This means that when multiple processes are not causally related, they could have simultaneous access to shared resources. However, if the processes are casually related, they should not have simultaneous access to any shared resource, but instead need to maintain the preferred order of execution. This is known as mutual exclusion (van Steen & Tanenbaum, 2017). Distributed algorithms should be designed in a way that the processes maintain mutual exclusion, i.e., do not have access to a shared resource simultaneously if causally related. The algorithms should also be designed in a way that avoids both starvation and deadlock.

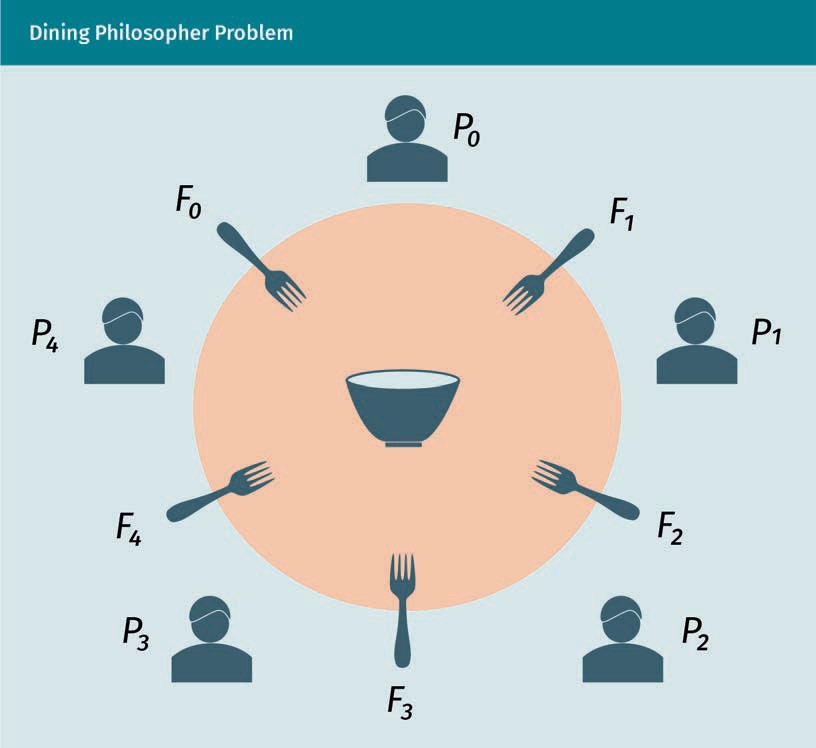
The dining philosophers problem is a classic illustration of deadlock and concurrency issues (Silberschatz et al., 2013). In this situation, a number of philosophers (represent- ing distributed processes) p0, p1, p2, …, pn − 1 are sitting around a dining table in a circular fashion in order to share food from a bowl in the center of the table. There are n forks (representing resources), which are distributed so that between any two philosophers, there is one fork. In the following ﬁgure, there are ﬁve philosophers and ﬁve forks. Now, if all philosophers need to pick up two forks in order to eat, then it would not be possible for them all to do this simultaneously. Only two philosophers could have two forks at any given time. It is possible that all philosophers can each have one fork simultaneously. However, in that case, all of them would be waiting for the next philosopher to release a fork since two are required to eat. Since they are all waiting for another fork, none of them is going to release their fork. This means all phi- losophers will be starving together, which will lead to a deadlock situation. To avoid starvation and deadlock, the solution should consider allowing only four philosophers to sit. A philosopher is able to pick forks only if both forks are available. An odd num- bered philosopher should pick the right fork ﬁrst, then left and vice versa (Silberschatz et al., 2013).

Starvation

This is a situation in which a process unfairly waits for a shared resource while other pro- cesses keep pro- gressing.

Deadlock

This is a situation in which causally rela- ted processes are forced to wait for each other, meaning that no process can progress.



Various types of algorithms for distributed systems have been developed over the years. Some commonly used distributed algorithms are discussed below.

###### Centralized Algorithm

In a centralized algorithm, a process acts as a scheduling coordinator. If a process wants to access to a shared resource, it ﬁrst sends a request to the coordinator proc- ess. If the resource is not occupied by other processes, then the coordinator sends a notiﬁcation granting permission (van Steen & Tanenbaum, 2017). A centralized algo- rithm is easier to implement and is more fair. The coordinator monitors to make sure no process is starving. However, the coordinator suffers from single point failure; if the coordinator is out of order, then the whole system starves. Additionally, if there are too many processes, then the coordinator suffers from request overloading.

Distributed Algorithms and Applications

###### Distributed Algorithms

Distributed algorithms were ﬁrst introduced by Ricart and Agrawala (1981). In a distrib- uted algorithm, if a process wants to access a resource, it sends a request message to all other processes. The message contains the process identiﬁcation (ID), the resource name, and a logical timestamp. If all other processes are not using the resource or the timestamps of all other processes are greater than the timestamp of the requesting process, then they respond with an okay (OK) message. If a process is already using the resource, then it keeps the request message in queue, and sends the okay message after it is done with the resource. If a process receives a request message and its time- stamp is earlier than the request message’s timestamp, then the process does not respond with an OK message. Instead, it utilizes the shared resource ﬁrst, and only then responds with the OK message. Distributed algorithms remove the single point of fail- ure shortcoming of the centralized algorithm. However, it still suffers from N point fail- ure. This means if there are N processes and any of them is out of order, then all other processes will not receive the OK response from the failed process, causing all other processes to starve. If there are too many processes, then there will also be too many broadcast messages, which will reduce the system throughput (Ricart & Agrawala, 1981).

###### Token Ring Algorithm

In this algorithm, the distributed processes create a logical ring in which each process knows the neighboring process (van Steen et al, 2017). Once the process p0 is done with

using a shared resource, it passes a token to the neighbor p1. If p1 is done using the resource, then it passes the token to p2. Whichever process possesses the token has the

right to access the resource. This algorithm is fair, and processes do not starve. How- ever, an issue could be that if a process is out of order, then there must be a mecha- nism to bypass the failed process to form a new ring. Also, the inclusion of a new proc- ess requires reforming the ring.

###### Decentralized Algorithm

Lin et al. (2004) proposed a fully distributed algorithm in which 2 replicas are created of a resource. A coordinator is assigned for each resource replica. Once a process requests to use a resource, the coordinators vote for the access permission. If the num- ber of votes for permission is 7 > 2 /2, then the process is allowed to access the resource.

### Transactions and Data Management (Consistency and Replication)

In a distributed system, the term transaction refers to operating on a set of data or ﬁle that is dispersed over multiple database or network nodes. A modern-day example of distributed transaction could be multiple users or processes working on a common ﬁle on cloud from different machines or network nodes. Data replication is another beneﬁt of distributed transaction. Nowadays, it is common practice to replicate data or ﬁles on cloud storage for backup purposes or to improve quality of service (QoS). Maintaining consistency is crucial for distributed transactions, which means once the ﬁle or data are edited or updated in one machine, they must be updated on all other machines too.

###### Distributed Transaction

A distributed transaction maintains the following properties: location transparency, replication transparency, concurrency transparency, and failure transparency (Traiger et al., 1982). The user can move and access the data from various locations and nodes. Multiple users or processes can also access the data from various locations and nodes simultaneously. This property is known as location transparency. The data can be repli- cated on various nodes, and the user only needs to update only one piece of data in order to update all of them. This property is called replication transparency. Multiple transactions may occur on a piece of distributed data by one or multiple process con- currently, a property which is known as concurrency transparency. According to the property of failure transparency, either all the transactions occur on a piece of data on various nodes, or none of them occur (Traiger et al., 1982).

###### Distributed Replication

Distributed replication is primarily necessary for two reasons: maintaining backups of data and improving quality of service by increasing throughput (van Steen et al, 2017). A server of a large data center receives too many requests from users, causing the server to overload, meaning it cannot satisfy the user demands. One widely practiced solution is replicating the contents to a server that is located closer to the users, thereby dis- tributing the server load to make it not overloaded anymore. The delay is also decreased due to the reduction in geographical distance and hop count. Distributed replication needs to deal with the following two challenges: placement of the replica- server and replica-content (van Steen & Tanenbaum, 2017).

Placement of replica-server

Placement of replica-server means ﬁnding the most suitable geographical location. One of the solutions, proposed by Qiu et al. (2001), has proposed that for each server node, the average distance from user nodes is to be calculated. The server node with smallest average should be chosen as the replica-server. However, they also suggested to consider the distance in terms of bandwidth rather than geographical distance.

Distributed Algorithms and Applications

Radoslavov et al. (2002) proposed a solution which ignores the distances between the clients and servers, rather it focusses on choosing the service area of the server and assumes that all users are uniformly distributed in that geographical region. Szymaniak et al. (2006) proposed an algorithm that ﬁnds the region where content is demanded by the greatest number of nodes. Once this has been identiﬁed, it ﬁnds the node which has the least inter-node latency.

Placement of replica-content

There can be three kinds of content replicas: permanent replicas, server-initiated repli- cas, and client-initiated replicas (van Steen & Tanenbaum, 2017). Permanent replicas are the initial replicas (i.e., initiated locally) and usually fewer in number. Permanent replicas are mainly the data storage for contents. Server-initiated replicas are created by a server process based on a request received by another server process. The pur- pose of the server-initiated replicas is to enhance system performance. Client-initiated replicas are created by a server process based on a request from a client process.

###### Consistency Management

Once a replica is edited, all other replicas must be updated in order to be consistent. There are two types of protocols to propagate updates to the other nodes: push and pull (van Steen & Tanenbaum, 2017). Push protocols are implemented in a server. Once there is a change in data, the push protocol forwards the update to all other nodes that contain replicas. Push protocols are also known as server-based protocols. In contrast, a pull protocol forwards update information only when an update is requested by a cli- ent. Pull protocols are also known as client-based protocols.

There are several models of consistency in distributed systems. Maintaining continuous consistency is crucial and challenging. There are three criteria used to characterize inconsistency: deviation in numerical values between replicas, deviation in staleness between replicas, and deviation in the ordering of update information (van Steen & Tanenbaum, 2017). Some important consistency models are discussed below.

Data-centric consistency

The data centric consistency model is a contract between a server process and data store. As long as the process operates following the rules, the memory performs cor- rectly (van Steen & Tanenbaum, 2017). Some data-centric consistency models are described here:

* Sequential consistency was introduced for multi-process systems with shared mem- ory (Lamport, 1979). A shared memory or data store is sequentially consistent if the read and write operations of multiple processes are performed sequentially by fol- lowing the preferred order speciﬁed in the program’s algorithm.
* Causal consistency is a modiﬁcation of sequential consistency. Two events are said to be causally related if the occurrence of one event depends upon another (van Steen et al, 2017). Suppose process p1 performs the event e1, which writes a piece of data x. Process p2 performs the event e2, which reads the piece of data *x*. If e2 is per- formed before e1 then the reading of x will be erroneous. Two events are called con-

Inter-node latency This is the time that elapses while data are transferred between two nodes.

current events if they are not causally related (van Steen & Tanenbaum, 2017). Causal consistency writes the causally related events in a speciﬁed order in a single machine or in multiple distributed machines. The order of concurrent events may differ on different machines.

* + Eventual consistency is relevant in a large-scale distributed system if data stores do not experience updates for a long time. If this occurs, then all stores eventually become consistent, i.e., they all contain the same data. This type of consistency is one of the key concepts in the basically availabile, soft-state, eventually consistent (BASE) model (Fox et al., 1997). Eventual consistency is applicable to a large-scale distributed system where databases are comparatively more tolerant of inconsis- tency. The domain name system (DNS) is an example that uses eventual consistency. DNS maintains a hierarchy of databases in which lower-level databases keep repli- cas of the top-level database. Any modiﬁcation in the top-level database does not propagate immediately to the lower-level databases. The top-level database sends an update to a low-level database once the low-level database requests an update. Over time, all databases become consistent eventually (van Steen & Tanenbaum, 2017).

Client-centric consistency

Unlike data-centric consistency, which focuses on maintaining global consistency, i.e., it makes replicas consistent for various users, client-centric consistency focuses on main- taining local or individual consistency. It means the replicas will be consistent on dis- tributed nodes for the individual client. Some popular client-centric consistency proto- cols are listed below (van Steen & Tanenbaum, 2017).

* + Monotonic read consistency ensures that once a process has seen the updated value of a data, it will never see an older value of that data. One example of this is an email database. Individual email mailboxes can be installed on multiple machines. Once a new email is included in the mailbox, all the machines do not necessarily update the mailbox immediately. The mailbox will only update when the user logs in.
  + Monotonic write consistency means if the client performs the write operation, then before updating the data ﬁrst, it will check if there has been any other write opera- tion prior to this performed on a different machine.
  + Read your writes consistency means the write operations will be completed before read operations, irrespective to which node has performed the writing operation.
  + Writes follow reads consistency implies any consecutive write operation by a client process on data will be performed on a replica that is updated with the value most recently read by the client process.

### Security Aspects for Distributed Services and Applications

In general, ensuring the security of a computer system means ensuring the conﬁden- tiality, integrity, and authentication triad (CIA-triad) (Stallings, 2017). However, ensuring the security of a distributed system is more challenging, compared to a centralized sys-

Distributed Algorithms and Applications

tem, due to the complex and distributed design, which makes the system more vulner- able. The three major security challenges of a distributed system are secure channel, access control, and security management, all of which depent on the CIA-triad (van Steen & Tanenbaum, 2017).

###### Secure Channel

Securing the channel means authenticating the communicating parties, i.e., sender and receiver, so that an attacker will not be able to pretend to be a valid sender or receiver. Having a secure channel also means the transferred message will not be accessed by an unauthorized user. So, only the authorized senders and receivers will have access to the message to read it and acquire the information. Another property of a secure chan- nel is that the message will not be edited or altered by an unauthorized user or system. A secure channel must guarantee conﬁdentiality, authentication, and integrity.

Conﬁdentiality is achieved through encrypting the message using symmetric-key encryption algorithm. Among symmetric key algorithms advanced encryption standard (AES) is most widely used (Stallings, 2017). However, an asymmetric key encryption algo- rithm is used to secure the exchange of symmetric key. The sender encrypts the sym- metric key using receiver’s public key (Stallings, 2017). Only the receiver has the private key so, no intruders are able to disclose the symmetric key. The most commonly used asymmetric key encryption algorithm is RSA (Stallings, 2017). The authentication of the sender and receiver is achieved through encrypting the message using asymmetric key encryption. However, in this case, the message is encrypted using the sender’s private key. The receiver can collect the public key from a trusted authority to decrypt the encrypted message. However, asymmetric key encryption-decryption is a time consum- ing process. For this reason, instead of encrypting the actual message ﬁrst, the hash code of the message is encrypted. This process is known as a digital signature (Stal- lings, 2017). Data integrity is achieved through a digital signature, which protects against data forgery. It ensures that content is not edited and ownership is authentic.

###### Access Control

Access control is also known as authorization (van Steen & Tanenbaum, 2017). Authori- zation ensures that only the authorized users, i.e., processes, are able to access to the software and hardware resources. Authorization comes after the authentication of the user or process. A common example of violation of authorization is a denial of service (DoS) attack or distributed denial of service (DDoS) attack. Access control can be imple- mented in the following ways:

* The access control list (ACL) is a database or list that included the name or ID of the authorized users or processes (van Steen & Tanenbaum, 2017).
* Access certiﬁcates are carried by each user and process. They list the rights of using various resources. One shortcoming of this process is that it requires a third-party authority to issue certiﬁcates. Thus, revoking a certiﬁcate may add additional delay (van Steen & Tanenbaum, 2017).
* Firewalls regulate or ﬁlter the access of a user based on the various properties of the user such as internet protocol (IP) address, media access control (MAC) address, and device name. A ﬁrewall can be implemented in between the machine and gate- way router. This type of ﬁrewall is known as packet-ﬁltering gateway. Firewalls can also be implemented at the application level. A proxy-gateway is a special type of application-level ﬁrewall which is implemented as the frontend of an application (van Steen & Tanenbaum, 2017).

###### Security Management

Security management involves the generation and distribution of public-private key pair of asymmetric key encryption and can be implemented in various ways (Stallings, 2017). The most prevalent solution is a third-party trusted authority that distributes public key and certiﬁcates for authentication and digital signature purposes.

###### Secure Mobile Code

Securing mobile code is an important issue in modern distributed systems. Nowadays, a distributed system transfers both data and source codes. Securing programming code is challenging because code can be malicious, such as malware, spyware, or ransom- ware. Malicious code can be downloaded with a piece of data or a ﬁle. Implementing sandbox is a popular solution for this issue. For example, every smartphone app runs in a sandbox. Sandbox executes the code in a restricted environment once installed and monitors activities. If any malicious activity is detected, sandbox halts the execu- tion of the code.

Zusammenfassung

The inter-process communication of a distributed system is done through remote procedure call (RPC). RPC is a request-response protocol and synchronous in nature, thus the concept of message-oriented middleware (MOM) is developed to provide asynchronous communication. Middleware is a program that resides between system programs and application programs on both client and server sides. The synchronization in distributed system can be achieved by using either global physical clock or logical clock. The global physical clock is known as univer- sal coordinated time (UTC). Two widely known logical clock synchronization proto- cols are Lamport’s logical clocks and vector clocks.

Distributed Algorithms and Applications

A key advantage of a distributed system is the ability to replicate data on various stores. However, when there is a change in a store, the other stores should be noti- ﬁed to update the replicas in order to provide same information to all users, which is known as a consistency issue. Consistency can be managed from the server side (data-centric consistency), as well as the client side (client-centric consistency). A distributed system can be secured by ensuring conﬁdentiality, integrity, and authentication. Access control or authorization can be maintained by access control list (ACL), access certiﬁcate, and ﬁrewall.



# Lektion 6

## From Distributed Systems to Ubiquitous Computing

#### LERNZIELE

Nach Abschluss dieser Lektion werden Sie in der Lage sein, ...

… understand the concepts of decentralized and mobile computing.

… analyze the concepts of mobile computing protocols.

… apply the aspects for distributed ledger technology in Internet of Things (IoT).

… design and implement a decentralized application.

DL-E-DLMCSNDS01-U06

1. From Distributed Systems to Ubiquitous Computing

### Einführung

Services and decisions of a centralized computer network are controlled in a single node. In a distributed system, the computation is done parallelly in multiple nodes. A distributed system can be either centralized or decentralized. In a centralized distrib- uted system, computation is done by multiple nodes, but the decision or service is still controlled centrally by a single node. In a decentralized distributed system, the deci- sion or service is not controlled centrally. Distributed ledger technology is a decentral- ized distributed system where a ledger i.e., database is distributed over multiple nodes. Different nodes can access and update the ledger concurrently. This unit will shed light on distributed ledger system. This unit will also discuss ubiquitous computing and Internet of Things (IoT). Ubiquitous computing also known as pervasive computing aims to enable computing on every kind of devices at every location. These devices may include computers, laptops, smartphones, smart watches, wearables, home appliances, and cars. Casually speaking, ubiquitous computing aims to perform computing on vari- ous things. The IoT aims to connect these various things to the internet.

### Distributed Ledger Technology

A distributed ledger technology (DLT) is based on a distributed decentralized network where a database is deployed over multiple distributed machines and there is no cen- tral authority to maintain the database. DLT provides a platform for developing decen- tralized and distributed applications for registering, sharing, and synchronizing transac- tions on digital assets (Antal et al., 2021). Since DLT is a decentralized and distributed technology, it is primarily built on peer-to-peer (P2P) network architecture instead of client-server architecture. Properties of distributed ledger technology are

* decentralized. DLT is implemented based on decentralized architecture, meaning there is no central authority. This architecture protects from any single point failure, or single point security attack, and is a protection against tempering by single party (Zhu et al., 2019).
* distributed. The resources of DLT are distributed on multiple machines. These resources include both software and hardware resources (Zhu et al., 2019).
* immutable. The actions and transactions on a device can be traced and audited by a ledger (Chowdhury et al., 2019).
* irreversible. The ledger transaction activities are irreversible after a certain period of time.
* data consistency. DLT ensures consistency of ledger data stored in distributed stor- age (Chowdhury et al., 2019).
* data provenance. Each transaction is digitally signed by public key cryptography which ensures the authenticity of the data source (Chowdhury et al., 2019).

From Distributed Systems to Ubiquitous Computing

* distributed consensus. DLT follows consensus algorithms in order to come to an agreement about a piece of data (Zhu et al., 2019).
* accountable and transparent. Since DLT preserves authenticity, immutability, irrever- sibility, and provenance, it promotes the accountability and transparency of transac- tions (Zhu et al., 2019).

###### Types of Distributed Ledgers

There are two main types of distributed ledgers: public and private. (Chowdhury et al., 2019). A public ledger is transparent and open to the public, meaning anyone can update the ledger state through transactions. A public ledger can raise privacy con- cerns for certain data. Unlike the public ledger, in a private ledger, only authorized enti- ties are allowed to access and update the ledger state. This allows it to preserve data privacy (Chowdhury et al., 2019).

###### DLT Data Structure

There are two main data structures for a distributed ledger: blockchain and direct acy- clic graph (DAG). Blockchain structure forms a chain of blocks or link list with hashes. Each block contains all transactions took place recently, usually within last couple of seconds or minutes (Antal et al, 2021). Hash of all transactions in a block is generated and forms a Merkel Tree data structure. The root of the tree is pointed in the block. Blockchain is an append-only data structure. All new transactions are appended to the tree. Thus, blockchain structure conﬁrms an immutable log of entire history of transac- tion in that network.

Direct acyclic graph (DAG) is comparatively less popular data structure for distributed transactions. The graph starts with a beginning transaction that is approved directly or indirectly by the other transactions of the graph. Once a new transaction is submitted, it must validate two previous transactions of the graph that were not approved yet (Antal et al., 2021).

###### DLT Protocol Stack

Anwendungsschicht

The application layer provides the software user interface, which can be developed by a third party on top of an underlying DLT architecture (Zhu et al., 2019).

Contract layer

Smart contracts are computer protocols that deﬁne a set of rules about state transi- tions and corresponding actions. Smart contracts are executed automatically on each node. Each contract has its own set of assets and states with a unique address. A smart contract can execute polynomial computational tasks. In other words, smart contracts provide a platform to build business applications such as cryptocurrencies (Zhu et al., 2019).

Consensus

This type of algo- rithm is used to ach- ieve agreement on a single data value in a distributed system.

Consensus layer

The consensus layer deals with distributed consensus in order to ensure the trustwor- thiness of a block. It also ensures that all nodes have consistent ledger copies. Due to network faults, delay, or malicious nodes, there might be disagreement between net- work nodes, and consensus layer aims to resolve these disputes (Farahani et al., 2021).

Internetschicht

The goal of the network layer is to provide distributed network architecture. It is done through forming a P2P network based on communication protocols like the transmis- sion control protocol/internet protocol (TCP/IP), for example.

Data layer

The data layer collects data from the lower layer through transactions and encapsu- lates it with data layer headers (Farahani et al., 2021). It also digitally signs the data for authentication and integrity purposes.

Device layer

This layer represents the hardware or physical devices such as sensors, actuators, and networking hardware, which can be connected both with wires and wirelessly (Farahani et al., 2021). The concept of the device layer is similar to that of the physical layer of the open systems interconnection (OSI) model or TCP/IP model.

###### DLT Platforms

Some of the most popular DLT platforms with their key features includeBitcoin, Ether- eum, Multichain, and the electro-optical system (EOS). Bitcoin is a DLT system that introduced the ﬁrst digital currency known as cryptocurrency. It is based on a P2P dis- tributed network. Moreover, it is a public DLT which means any user across the world is allowed to use the P2P network. It uses proof of work (PoW), also known as the Naka- moto consensus algorithm, to ensure the validity of each block. It improves the preci- sion accuracy of data stored in a distributed ledger (Chowdhury et al., 2019).

Ethereum, like Bitcoin, is also based on public DLT and its primary application is crypto- currency. It supports smart contract, which is deployed by transacting Ethereum crypto- currency called Ether. Ethereum also uses the proof of work (PoW) consensus algorithm (Chowdhury et al, 2019). Multichain is a public platform that allows the deployment of private distributed ledger applications. It was developed based on Bitcoin blockchain and allows the users to update block size and protocols, among other aspects. (Chowd- hury et al., 2019). The electro-optical system (EOS) was initially developed on the Ether- eum platform. The DLT platform uses a delegated proof of stake (DPoS) consensus algorithm. Details of the DPoS algorithm can be found in Luo et al. (2018). DPoS uses 21 validators known as block producers (Chowdhury et al., 2019). Some other well-known DLT platforms are Cardano, Fabric, Sawtooth, IOTA Multichain, Cobra, and Waltonchain (Chowdhury et al., 2019).

From Distributed Systems to Ubiquitous Computing

### Aspects of Mobile Computing

Mobile computing refers to a computing system that is based on a mobile network. The term mobile network means that the nodes of the network can move. Examples of mobile nodes are mobile phone devices, wearables like ﬁtness tracker or smart watches, laptops, and personal digital assistants (PDAs). A mobile computing system is developed on a wireless network and is mainly built on wireless communication proto- cols.

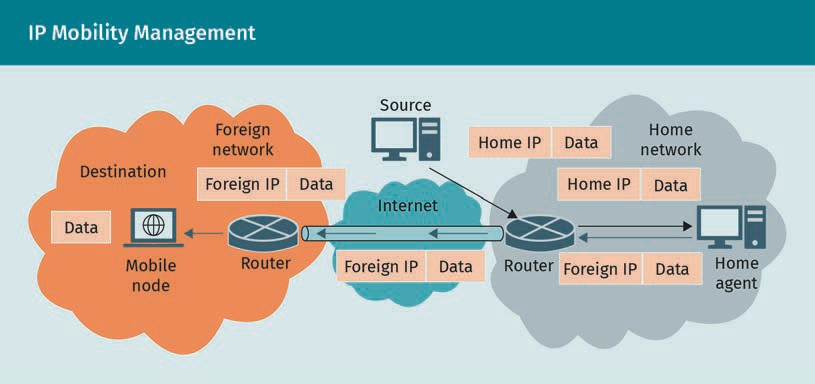
###### Mobile Computing and Mobile Networks

The key characteristic of a mobile computing system is that it is deployed on a mobile network. A network can be considered mobile if it shows any of the following criteria (Ahmed et al., 2010):

* user mobility. The user should be able to access to the same service from various locations.
* device mobility. The user should be access to a service from various end devices.
* network mobility. The entire network can be mobile for example mobile ad hoc or body sensor network. It could also mean that a user should be able to access to a service through different networks. For example, a user can connect to a real-time service through Wi-Fi while roaming through home, coffee shop, or a campus net- work.
* host mobility. The mobile device could be either a client or a server. If it is a server, then it should be able to serve a client while roaming through various geographical areas and networks.

###### Mobile IP Management

Mobile devices are continually roaming from network to network, depending on the mobility of its owner or carrier. As a result, the management of the IP address of a mobile node (known as IP mobility) is challenging. The solution to IP mobility is ach- ieved through implementing a home agent. In a network, the IP of a mobile node is always identiﬁed by its home IP address, even it moves to a new geographical location. However, while it is moving away from the home location, it is associated with a “care of” address, which identiﬁes the current location of the mobile node. Once a datagram is directed to a mobile node, the datagram is ﬁrst received by the home agent, which simply forwards the datagram to the mobile node. The protocol for managing IP mobi- lity is deﬁned in the internet protocol standards RFC 3344, 4721, 5944, and 6275 (Perkins, 2010).



###### Mobile Network Routing Protocols

The topology of a mobile network is not ﬁxed and continually changes over time. Thus, the ID or IP of a neighbor node, route between two nodes, or link cost between two nodes may change over time. Traditional routing algorithms or protocols perform poorly for mobile networks because they do not address this issue. Some of the com- mon routing protocols for mobile networks are provided below (Othman, 2007).

Ad hoc on-demand distance vector (AODV)

AODV is developed to resolve routing issues of mobile ad hoc network and is used by the mobile nodes of the network. It offers a small memory and processing overhead, low power consumption, and low network utilization. It also avoids the problems of classical distance vector protocols. AODV is deﬁned in RFC 3561 (Perkins et al., 2003).

Dynamic source routing protocol (DSR)

DSR is a simple routing protocol for mobile ad hoc network with multiple hops. Two major goals of this algorithm are route discovery and route maintenance to an arbitrary destination. This on-demand protocol allows multiple routes to a destination and allows the sender to choose a route. DSR guarantees loop-free routing and offers rapid recovery from a change in topology. DSR was primarily designed for a network with up to 200 mobile nodes. DSR protocol is deﬁned in RFC 4728 (Yih-Chun et al., 2007).

Optimized link state routing protocol (OLSR)

OLSR protocol is also developed for mobile ad hoc wireless networks. It is an optimized form of link state routing algorithm that selects some nodes as multipoint relays (MPRs). The purpose of these selected nodes is to broadcast messages to declare link state information. The use of MPRs reduces the message overhead compared to the classical link state algorithm. In the classical link state algorithm, each node retrans- mits each message once it has been received for the ﬁrst time. However, in OLSR, only

From Distributed Systems to Ubiquitous Computing

the selected nodes (i.e., MPRs) generate link state information, thereby reducing net- work ﬂooding. OLSR is suitable for a large scale mobile ad hoc network with high node density. OLSR is deﬁned in RFC 3626 (Clausen & Jacquet, 2003).

### Aspects of Pervasive Computing and the Internet of Things

Pervasive computing aims to make computing available from any geographical location and electronic device, at any time. End nodes of pervasive computing include not only desktop or laptop computers, but also cellphone devices, sensors, actuators, home appliances, and wearable devices. An emerging application of pervasive computing is the IoT. Pervasive computing is also referred to as ubiquitous computing.

###### Properties of Pervasive Computing

The key properties of pervasive computing are as follows (Poslad, 2011):

* context awareness. Network nodes have to be context aware in order to optimize performance.
* distribution. The system should be deployed over a distributed network.
* autonomy. Network nodes should be self-governing, meaning that they should be able to operate without human intervention.
* human-computer interactions (HCI). Human-computer interactions should be mini- mized and hidden.
* intelligence. The network nodes should be intelligent, i.e., artiﬁcial intelligence should be integrated.

###### Internet of Things

One of the biggest outcomes of ubiquitous or pervasive computing is the IoT. Tradi- tional internet connects only computers, while the IoT connects computers along with a wide variety of electronics, giving it its name. In recent years, several protocol stand- ards, hardware platforms, and technologies have been introduced for IoT technology.

IoT protocol standards

The Internet Engineering Task Force (IETF) formed the Constrained RESTful Environ- ments (CORE) research group, which develops protocol standards for constrained IP networks. A constrained IP network includes nodes with limited memory size, low power, and low throughput capacity. This type of network may experience compara- tively more packet loss and may include a large number of nodes (sensors and actua- tors) which may turn on or off periodically. IoT is an example of constrained IP network (Constrained RESTful Environments (CoRE), n.d.). Some IoT protocol standards are intro- duced below.

###### Constrained Application Protocol (CoAP)

CoAP is a web transfer protocol designed for a wireless network with constrained nodes i.e., low power, low memory, and low data transfer rate. These nodes may have 8-bit microcontrollers and small ROM and RAM. CoAP operates on request/response archi- tecture. It includes basic ideas of the web, for example, uniform resource identiﬁers (URIs) and media types. CoAP is standardized in 7252, 7959, 8613, and 8974 (Shelby et al., 2014).

###### IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)

The low-power wireless personal area network (LoWPAN) comprises devices with low power, computation speed, memory, and bit rate. These devices follow properties given in IEEE 802.15.4-2003 standard. 6LoWPAN network assumptions, its problem statements and goals are standardized in RFC 4919 (Kushalnagar et al., 2007).

###### WebSocket

WebSocket is aimed at supporting browser-based applications that require bidirec- tional communication with servers that do not depend on opening multiple HTTP con- nections. WebSocket utilizes existing concepts, such as proxies, ﬁltering, and authenti- cation. WebSocket operates over port 80 and 443 to use the HTTP proxies. Websocket is deﬁned in RFC 6455 (Fette & Melnikov, 2011).

###### IoT Technologies

The development in system on chip (SoC) technology introduced a few low-power, low- cost, lightweight network nodes, which are being treated as the hardware platform of various IoT technologies. Some of the popular IoT technologies and relevant hardware platforms include radio frequency identiﬁcation (RFID) and low range (LoRa).

Transponders This is a network device that operates as both a transmitter and a responder.

Radio frequency identiﬁcation (RFID)

Radio frequency identiﬁcation (RFID) system uses an electromagnetic ﬁeld to identify or track objects, and record data (Jia et al., 2012). In an RFID setup, there are RFID tags (also called transponders) and RFID readers. An RFID tag is attached to the target object. In general, the RFID tag is passive and is powered by the radio wave released from the RFID reader. However, there are also active RFID tags which are powered by battery. RFID readers are transceivers, containing a radio frequency interface (RFI) mod- ule and a wireless switch.

LoRa

LoRa stands for low range. It is a spread spectrum modulation technique for low-power wide area network (LPWAN) technology. LoRa provides long-range communication using low-power network nodes. For example, using LoRa it is possible to communicate over

From Distributed Systems to Ubiquitous Computing

a distance of 160 meters using 86.5 megajoules of energy. LoRa operates on lower industrial, scientiﬁc, and medical (ISM) band frequencies (868 megahertz (MHz) and 433 MHz for the EU; 915 MHz and 433 MHz for the US). LoRaWAN is a MAC layer protocol that operates on a network with star topology (Bor et al., 2016).

Zusammenfassung

A distributed ledger technology (DLT) is a decentralized distributed network infra- structure in which a ledger, i.e., database, is deployed over multiple distributed machines with no central authority to maintain the database. DLT provides a plat- form for developing decentralized and distributed applications for registering, shar- ing, and synchronizing transactions on digital assets. The key properties of DLT are: decentralized, distributed, immutability, irreversibility, data consistency, data prove- nance, distributed consensus, accountability, and transparency.

There are two types of distributed ledger: public and private. Two data structures that prevent DLT are blockchain and direct acyclic graph. DLT protocol stack con- tains following layers: application, contract, consensus, network, data, and device. Some popular DLT platforms are Bitcoin, Ethereum, Multichain, and EOS.

Mobile computing considers mobile computing nodes, such as user mobility, device mobility, network mobility, and host mobility. The IP of a mobile node is managed by implementing a home agent. ad hoc on-demand distance vector (AODV), dynamic source routing protocol (DSR), and optimized link state routing protocol (OLSR) are some well-known routing protocols for mobile network. One of the big- gest outcomes of ubiquitous or pervasive computing is Internet of Things (IoT). Tra- ditional internet connects computers, while the IoT also connects a wide variety of electronics, such as home appliances, handheld devices, sensors, switches, and actuators.

Constrained application protocol (CoAP), constrained RESTful environments (CoRE), and IPv6 over low-power wireless personal area networks (6LoWPAN) are some well-known protocols for IoT. There are various hardware and software platform for IoT. Some of the popular platforms are RFID, Zigbee, NodeMCU, and LoRa.



# Anhang 1

## Quellenangaben

Quellenangaben

Abed, G. A., Ismail, M., & Jumari, K. (2011). A survey on performance of congestion control mechanisms for standard TCP versions. *Australian Journal of Basic and Applied Scien- ces*, *5*(12), 1345—1352. <http://www.ajbasweb.com/old/ajbas/2011/Decem-> ber-2011/1345-1352.pdf

Ahmed, H., Talukder, A., & Yavagal, R. (2010). *Mobile computing: Technology, applications and service creation*. McGraw Hill.

Antal, C., Cioara, T., Anghel, I., Antal, M., & Salomie, I. (2021). Distributed ledger technol- ogy review and decentralized applications development guidelines. *Future Internet*, *13*(3), 62—94. https://doi.org/10.3390/ﬁ13030062

Arkko, J., & Pignataro, C. (2009, April). *IANA allocation guidelines for the address resolu- tion protocol (ARP)* (RFC 5494). Internet Engineering Task Force. https://data- tracker.ietf.org/doc/html/rfc5494

Baran Ivo. (2007, October 22). FTP cable3. *Wikimedia Commons*. https://commons.wiki- media.org/wiki/File:FTP\_cable3.jpg

Baresi, L., & Garriga, M. (2020). Microservices: The evolution and extinction of web serv- ices? In A. Bucchiarone, N. Dragoni, S. Dustdar, P. Lago, M. Mazzara, V. Rivera, & A. Sado- vykh (Eds.), *Microservices* (pp. 3—28). Springer. https://doi.org/ 10.1007/978-3-030-31646-4\_1

Benzekki, K., El Fergougui, A., & Elbelrhiti Elalaoui, A. (2016). Software-deﬁned network- ing (SDN): A survey. *Security and Communication Networks*, *9*(18), 5803—5833. https:// onlinelibrary.wiley.com/doi/epdf/10.1002/sec.1737

Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O’Connor, B., Radoslavov, P., Snow, W., & Parulkar, G. (2014, August 22). ONOS: Towards an open, dis- tributed SDN OS. *HotSDN ’14: Proceedings of the third workshop on hot topics in soft- ware deﬁned networking* (pp. 1—6). Association for Computing Machinery. https:// doi.org/10.1145/2620728.2620744

Birrell, A. D., & Nelson, B. J. (1984). Implementing remote procedure calls. *ACM Transac- tions on Computer Systems (TOCS)*, *2*(1), 39—59. https://web.eecs.umich.edu/~mosharaf/ Readings/RPC.pdf

Bor, M., Vidler, J. E., & Roedig, U. (2016). LoRa for the Internet of Things. *EWSN ’16 Pro- ceedings of the 2016 international conference on embedded wireless systems and net- works* (pp. 361—366). Junction Publishing.

Brownell, D. (1996). *Dynamic RARP extensions for automatic network address acquisi- tion.* Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc1931

Quellenangaben

Cain, B., Deering, S., Kouvelas, I., Fenner, B., & Thyagarajan, A. (2002, October). *Internet group management protocol, version 3*. Internet Engineering Task Force. https://data- tracker.ietf.org/doc/html/rfc3376

Chowdhury, M. J. M., Ferdous, M. S., Biswas, K., Chowdhury, N., Kayes, A. S. M., Alazab, M., & Watters, P. (2019). A comparative analysis of distributed ledger technology platforms. *IEEE Access*, *7*, 167930—167943. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnum- ber=8902067

Cisco. (2020). *New Cisco annual internet report forecasts 5G to support more than 10% of global mobile connections by 2023*. The Newsroom. https://newsroom.cisco.com/ press-release-content?articleId=2055169#:~:text=By%202023%3A,of%20those%20will

%20be%20mobile

Clausen, T., & Jacquet, P. (2003). *Optimized link state routing protocol (OLSR)* (RFC 3626). RFC Editor. https://doi.org/10.17487/RFC3626

Carnegie Mellon University (CMU). (n.d.). *ASCII table*. Carnegie Mellon University. https:// [www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html](http://www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html)

Cohen-Almagor, R. (2013). *Moral, ethical, and social dilemmas in the age of technology: Theories and practice*. IGI Global. https://doi.org/10.4018/978-1-4666-2931-8.ch002

Comer, D. E. (2015). *Computer networks and internets* (6th ed., global ed.). Pearson Edu- cation.

Comer, D. E. (2018). *The internet book: Everything you need to know about computer net- working and how the internet works* (5th ed.). CRC Press.

Conformance Speciﬁcation Radio. (2011). *3rd generation partnership project; Technical speciﬁcation group radio access network; Evolved universal terrestrial radio access (E- UTRA); User equipment (UE) conformance speciﬁcation radio transmission and recep- tion*. European Telecommunication Standards Institute (ETSI).

Constrained RESTful Environments (CoRE) (n.d.). *Charter for working group*. Internet Engineering Task Force. https://datatracker.ietf.org/wg/core/about/

Docker. (n.d.). *Orientation and setup*. https://docs.docker.com/get-started/

Department of Defence High Performance Computing Modernization (DoD HPC). (n.d.). *Neighbor discovery protocol attacks*. U.S. Department of Defense High Performance Computing Modernization Program. https://[www.hpc.mil/program-areas/networking-](http://www.hpc.mil/program-areas/networking-) overview/2013-10-03-17-24-38/ipv6-knowledge-base-security/neighbor-discovery-proto- col-attacks

Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustaﬁn, R., & Saﬁna,

L. (2017). Microservices: Yesterday, today, and tomorrow. In M. Mazzara & B. Meyer (Eds.),

*Present and ulterior software engineering* (pp. 195—216). Springer.

Elson, J., Girod, L., & Estrin, D. (2002). Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review*, *36*(SI), 147—163. https:// doi.org/10.1145/844128.844143

Endrei, M., Ang, J., Arsanjani, A., Chua, S., Comte, P., Krogdahl, P., Luo, M., & Newling, T. (2004). *Patterns: Service-oriented architecture and web services*. IBM Corporation.

Etzkorn, L. H. (2017). *Introduction to middleware: Web services, object components, and cloud computing*. Chapman and Hall/CRC.

Farahani, B., Firouzi, F., & Lücking, M. (2021). The convergence of IoT and distributed ledger technologies (DLT): opportunities, challenges, and solutions. *Journal of Network and Computer Applications*, *177*, 102936. https://doi.org/10.1016/j.jnca.2020.102936

Fette, I., & Melnikov, A. (2011). *The Websocket protocol* (RFC 6455). RFC Editor. https:// datatracker.ietf.org/doc/html/rfc6455

Finlayson, R., Mann, T., Mogul, J., & Theimer, M. (1984, June). *A reverse address resolution protocol* (RFC 903). Internet Engineering Task Force. https://datatracker.ietf.org/doc/ html/rfc903

Ford-Hutchinson, P. (2005, October). *Securing FTP with TLS* (RFC 4217). Internet Engineer- ing Task Force. https://datatracker.ietf.org/doc/html/rfc4217

Forouzan, B. A. (2013). *Data Communications and networking* (5th ed.). McGraw-Hill.

Fox, A., Gribble, S. D., Chawathe, Y., Brewer, E. A., & Gauthier, P. (1997, October). Cluster- based scalable network services*. Proceedings of the sixteenth ACM symposium on oper- ating systems principles* (pp. 78—91). https://doi.org/10.1145/268998.266662

Gabhart, K., & Bhattacharya, B. (2008). *Service oriented architecture ﬁeld guide for exec- utives*. Wiley & Sons.

Google. (n.d.). *Statistics*. Google IPv6. https://[www.google.com/intl/en/ipv6/statis-](http://www.google.com/intl/en/ipv6/statis-) tics.html

Internet Assigned Numbers Authority (IANA). (n.d.). *Service name and transport protocol port number registry.* https://[www.iana.org/assignments/service-names-port-numbers/](http://www.iana.org/assignments/service-names-port-numbers/) service-names-port-numbers.xhtml?&page=1

Inductiveload. (2007, October 26). EM Spectrum properties edit. *Wikimedia Commons*. https://commons.wikimedia.org/wiki/File:EM\_Spectrum\_Properties\_edit.svg

Information Sciences Institute. (1981). *Transmission control protocol: DARPA internet program protocol speciﬁcation* (RFC 793)*.* Internet Engineering Task Force. https://data- tracker.ietf.org/doc/html/rfc793

Quellenangaben

Jia, X., Feng, Q., Fan, T., & Lei, Q. (2012, April). RFID technology and its applications in Internet of Things (IoT). *2012 2nd international conference on consumer electronics, communications and networks (CECNet)* (pp. 1282—1285). IEEE.

Khan, W. Z., Ahmed, E., Hakak, S., Yaqoob, I., & Ahmed, A. (2019). Edge computing: a sur- vey. *Future Generation Computer Systems*, *97*, 219—235. https://doi.org/10.1016/ j.future.2019.02.050

Khatri, A., & Khatri, V. (2020). *Mastering service mesh: Enhance, secure, and observe cloud-native applications with Istio, Linkerd, and Consul*. Packt.

Kurose, J., & Keith R. (2017). *Computer networking: A top-down approach* (7th ed., global edition). Pearson Education.

Kushalnagar, N., Intel Corp, Montenegro, G., Microsoft Corporation, & Schumacher, C. (2007). *IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals* (RFC 4919). RFC Editor. https://www.rfc-edi- tor.org/rfc/pdfrfc/rfc4919.txt.pdf

Lamport, L. (1979). How to make a multiprocessor computer that correctly executes mul- tiprocess program. *IEEE Transactions on Computers*, *C-28*(9), 690—691. https://doi.org/ 10.1109/TC.1979.1675439

Leiner, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., Postel, J., Rob- erts, L. G., & Wolff, S. (2009). A brief history of the Internet. *ACM SIGCOMM Computer Communication Review*, *39*(5), 22—31. https://doi.org/10.1145/1629607.1629613

Lin, S. D., Lian, Q., Chen, M., & Zhang, Z. (2004, February). A practical distributed mutual exclusion protocol in dynamic peer-to-peer systems. In G. M. Voelker & S. Shenker (Eds.), *International workshop on peer-to-peer systems* (pp. 11—21). Springer.

Liu, F., Tang, G., Li, Y., Cai, Z., Zhang, X., & Zhou, T. (2019). A survey on Edge computing systems and tools. *Proceedings of the IEEE, 107*(8), 1537—1562. https://doi.org/10.1109/ JPROC.2019.2920341

Luo, Y., Chen, Y., Chen, Q., & Liang, Q. (2018, November). A new election algorithm for DPos consensus mechanism in blockchain. *2018 7th international conference on digital home (ICDH)* (pp. 116—120). IEEE.

Mills, D. L. (1984). *Exterior gateway protocol formal speciﬁcation* (RFC 904). Internet Engi- neering Task Force. https://datatracker.ietf.org/doc/html/rfc904

Mills, D. L. (1991). Internet time synchronization: The network time protocol. *IEEE Trans- actions on communications*, *39*(10), 1482—1493.

Mills, D., Martin, J. (Ed.), Burbank, J., & Kasch, W. (2010). *Network time protocol version 4: Protocol and algorithms speciﬁcation* (RFC 5905). RFC Editor. https://[www.rfc-editor.org/](http://www.rfc-editor.org/) info/rfc5905

Monolithic Power Systems (MPS). (n.d.). *Analog signals vs. digital signals*. Monolithic Power Systems. https://[www.monolithicpower.com/en/analog-vs-digital-signal](http://www.monolithicpower.com/en/analog-vs-digital-signal)

Namiot, D., & Sneps-Sneppe, M. (2014). On micro-services architecture. *International Journal of Open Information Technologies*, *2*(9), 24—27. https://[www.researchgate.net/](http://www.researchgate.net/) publication/265292970\_On\_Micro-services\_Architecture

Newpaltz. (n.d.). *Notes on different types of delays*. <http://cs.newpaltz.edu/>

~easwaran/CN/Module2/delays.html

Oksman, V., Strobel, R., Wang, X., Wei, D., Verbin, R., Goodson, R., & Sorbara, M. (2016). The ITU-T’s new G.fast standard brings DSL into the gigabit era. *IEEE Communications Maga- zine*, *54*(3), 118—126. https://doi.org/10.1109/MCOM.2016.7432157

Othman, M. (2007). *Principles of mobile computing and communications*. CRC Press.

Papastergiou, G., Fairhurst, G., Ros, D., Brunstrom, A., Grinnemo, K. J., Hurtig, P., & Mangi- ante, S. (2016). De-ossifying the internet transport layer: A survey and future perspec- tives. *IEEE Communications Surveys & Tutorials*, *19*(1), 619—639. <http://dx.doi.org/> 10.1109/COMST.2016.2626780

Perkins, C. (2010). *IP mobility support for IPv4, revised* (RFC 5944). RFC Editor. https:// [www.rfc-editor.org/info/rfc5944](http://www.rfc-editor.org/info/rfc5944)

Perkins, C., Belding-Royer, E., & Das, S. (2003). *Ad hoc on-demand distance vector (AODV) routing* (RFC 3561). RFC Editor. https://[www.rfc-editor.org/info/rfc3561](http://www.rfc-editor.org/info/rfc3561)

Polese, M., Chiariotti, F., Bonetto, E., Rigotto, F., Zanella, A., & Zorzi, M. (2019). A survey on recent advances in transport layer protocols. *IEEE Communications Surveys & Tutorials*, *21*(4), 3584—3608. https://doi.org/10.1109/COMST.2019.2932905

Poslad, S. (2011). *Ubiquitous computing: smart devices, environments and interactions*. Wiley & Sons.

Postel, J. (1980). *User datagram protocol* (RFC 768). Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc768

Postel, J. (1981, September). *Internet control message protocol: DARPA internet program protocol speciﬁcation* (RFC 792). Internet Engineering Task Force. https://data- tracker.ietf.org/doc/html/rfc792

Qiu, L., Padmanabhan, V. N., & Voelker, G. M. (2001, April). On the placement of web server replicas. *Proceedings IEEE INFOCOM 2001: Conference on computer communica- tions—Twentieth annual joint conference of the IEEE Computer and Communications Society*, *3*, 1587—1596. IEEE.

Quellenangaben

Radoslavov, P., Govindan, R., & Estrin, D. (2002). Topology-informed internet replica placement. *Computer Communications*, *25*(4), 384—392. https://doi.org/10.1016/ S0140-3664(01)00410-8

Ricart, G., & Agrawala, A. K. (1981). An optimal algorithm for mutual exclusion in com- puter networks. *Communications of the ACM*, *24*(1), 9—17. https://citeseerx.ist.psu.edu/ viewdoc/download?doi=10.1.1.507.9382&rep=rep1&type=pdf

Rimal, B. P., Choi, E., & Lumb, I. (2009, August 25—27). A taxonomy and survey of cloud computing systems. *2009 ﬁfth international joint conference on INC, IMS and IDC* (pp. 44

—51). IEEE. https://doi.org/10.1109/NCM.2009.218

Rosen, E., Vishwanathan, A., & Callon, R. (2001). *Multiprotocol label switching architec- ture* (RFC 3031)*.* Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/ rfc3031

Santiago, B. (2020). *Concurrency and parallelism for beginners*. Github. https:// github.com/santiagobasulto/pycon-concurrency-tutorial-2020

Sharma, R., & Singh, A. (2020). *Getting started with Istio service mesh: Manage microser- vices in Kubernetes*. Apress. https://doi.org/10.1007/978-1-4842-5458-5

Shelby, Z., Hartke, K., Bormann, C., & Frank, B. (2014). *The constrained application proto- col (CoAP)* (RFC 7252). RFC Editor. https://[www.rfc-editor.org/info/rfc7252](http://www.rfc-editor.org/info/rfc7252)

Shen, N., & Smit, H. (2004, October). *Calculating interior gateway protocol (IGP) routes over trafﬁc engineering tunnels* (RFC 3906). International Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc3906

Silberschatz, A., Galvin, P. B., & Gagne, G. (2013). *Operating system concepts essentials*. Wiley.

Solomon, M., & Kim, D. (2021). *Fundamentals of communications and networking* (3rd ed.). Jones & Barlett Learning.

Srlefﬂer. (2009, June 15). Optical ﬁber cable. *Wikimedia Commons*. https:// commons.wikimedia.org/wiki/File:Optical\_ﬁber\_cable.jpg

Stallings, W. (2017). *Cryptography and network security: Principles and practice* (7th ed.). Pearson.

Szymaniak, M., Pierre, G., & van Steen, M. (2006). Latency-driven replica placement. *IPSJ Digital Courier*, *2*, 561—572. <http://www.globule.org/publi/LDRP_ipsj2006.pdf>

Tanenbaum, A. S., & Wetherall, D. J. (2014). *Computer networks: New international edition*

(5th ed.). Pearson.

Thurlow, R. (2009). *RFC 5531 on RPC: Remote procedure call protocol speciﬁcation ver- sion 2* (RFC 5531). RFC Editor. https://[www.rfc-editor.org/pipermail/rfc-dist/2009-May/](http://www.rfc-editor.org/pipermail/rfc-dist/2009-May/) 002287.html

Tkgd2007. (2008, May 31). Coaxial cable cutaway. *Wikimedia Commons*. https:// commons.wikimedia.org/wiki/File:Coaxial\_cable\_cutaway.svg

Traiger, I. L., Gray, J., Galtieri, C. A., & Lindsay, B. G. (1982). Transactions and consistency in distributed database systems. *ACM Transactions on Database Systems (TODS)*, *7*(3), 323

—342.

Van Steen, M., & Tanenbaum, A. S. (2017). *Distributed systems* (3rd ed.). Maarten van Steen.

Venaas, S., & Retana, A. (2020, February). *PIM message type space extension and reserved bits* (RFC 8736). Internet Engineering Task Force. https:// datatracker.ietf.org/doc/html/rfc8736

Ward, J. S., & Barker, A. (2013). *A cloud computing survey: Developments and future trends in infrastructure as a service computing*. arXiv. https://arxiv.org/abs/1306.1394v1

Wireshark Foundation. (n.d.). Wireshark wiki. *GitLab*. https://wiki.wireshark.org/Sample- Captures

Yih-Chun, H. U., Maltz, D. A., & Johnson, D. B. (2007). *The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4* (RFC 4728). RFC Editor. https://www.rfc-edi- tor.org/rfc/rfc4728.txt

Zhu, Q., Loke, S. W., Trujillo-Rasua, R., Jiang, F., & Xiang, Y. (2019). Applications of distrib- uted ledger technologies to the Internet of Things: A survey. *ACM Computing Surveys (CSUR)*, *52*(6), 1—34.