1 Serverdienste bereitstellen und Administrationsaufgaben automatisieren (Lernfeld 10b)



Darüber werden wir sprechen, kompetent und aktiv uns beteiligen, genauer im Arbeitsbuch folgende Lernsituationen/Lernarrangements bearbeiten:

- Lernsituation 1: Wir unterscheiden Serverdienste und Plattformen.
- Lernsituation 2: Wir wählen Serverdienste und Plattformen gemäß Kundenanforderungen aus.
- Lernsituation 3: Wir planen Serverdienste und Plattformen.
- Lernsituation 4: Wir implementieren und testen Serverdienste.
- Lernsituation 5: Wir automatisieren Administrationsaufgaben.
- Lernsituation 6: Wir stellen Serverdienste und Plattformen vor.

Kaum ein Unternehmen kommt ohne eine entsprechende IT-Infrastruktur aus. Daher ist ein sicherer und ressourcenschonender Betrieb dieser Infrastruktur von großer Bedeutung. Serverdienste und Plattformen müssen möglichst einfach in großem Umfang in Betrieb genommen und zuverlässig an neue Anforderungen angepasst werden. Durch die Wahl geeigneter Tools können diese Ziele erreicht werden.

1.1 Unterschiede von Serverdiensten sowie Plattformen beschreiben

S **Sie sollen die Serverdienste und Plattformen für unterschiedliche Einsatzgebiete untersuchen.**

Es gibt einen grundlegenden Funktionsumfang, der in Betrieben unterschiedlicher Größe benötigt wird, auch bei der JIKU IT-Solutions GmbH und ihren Kunden. Hierzu gehören sowohl netznahe Dienste (z. B. IP-Adressvergabe, Namensauflösung) als auch anwendungsbezogene Dienste (z. B. E-Mail oder Webserver). Viele dieser Dienste arbeiten nach denselben grundlegenden Prinzipien hinsichtlich des Austauschs und der Ablage von Informationen.

Als IT-Systemhaus betreut JIKU IT-Solutions die IT-Infrastruktur verschiedener Kunden. Die Kunden haben verschiedene Aufgaben und Anforderungen, die durch die Mitarbeiter von JIKU IT-Solutions umgesetzt werden müssen. Die Auszubildenden begleiten die Mitarbeiter zu den Kundeneinsätzen und kommen mit den verschiedenen Techniken in Kontakt. Im folgenden Abschnitt werden exemplarisch einige dieser grundlegenden Dienste und Plattformen vorgestellt.

1.1.1 Serverdienste unterscheiden

S **Sie sollen verschiedene Serverdienste, die in den meisten Unternehmensnetzen im Einsatz sind, unterscheiden können. Dazu erarbeiten Sie sich die notwendigen Grundlagen.**

(1) Server

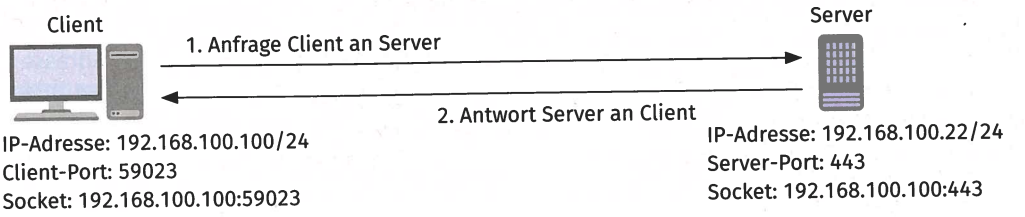
Server
Nach dem Client-Server-Modell stellt ein Server Dienste zur Verfügung, die von Clients genutzt werden.

In einem Client-Server-Netzwerk gibt es einen oder mehrere Server, die den Client-Systemen Dienste zur Verfügung stellen. Diese Clients müssen keine Endbenutzersysteme sein, sondern können auch vom Benutzer unabhängige Systeme sein.

In einem Netzwerk kommen eine Vielzahl von Diensten zum Einsatz. Beispielsweise stellt ein E-Mail-Server einen E-Mail-Dienst zur Verfügung und der E-Mail-Client nimmt den Dienst in Anspruch. Ein Webserver hingegen stellt beispielsweise eine Webseite bereit, die durch den Client, z. B. den Webbrowser, abgerufen werden kann.

Die meisten Computernetzwerke arbeiten mit dem TCP/IP-Protokollstapel. In einer solchen Umgebung verfügen Server und Client über IP-Adressen. Eine Adressierung über die IP-Adressen ist allerdings nicht ausreichend, denn ein Server kann unter einer IP-Adresse verschiedene Dienste anbieten. Außerdem kann ein Client mit einer IP-Adresse mehrere Dienste anfragen. Aus diesem Grund ist es nötig, dass zusätzlich zur IP-Adresse Ports (OSI-Schicht 4) für die Adressierung verwendet werden. Die Kombination aus IP-Adresse und Port wird **Socket** genannt.

Beispiel (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.1):



Kommunikation zwischen Client und Server

In der Abbildung bietet der Server auf dem Socket 192.168.100.22:443 einen Webserver-Dienst an. Der Port 443 ist der „Well-Known-Port" für das Protokoll HTTPS (HyperText Transfer Protocol Secure).

Der Client stellt eine Anfrage (Nachricht 1) an den Server. Dazu verwendet er als Quell-Socket den Socket 192.168.100.100:59023. Der Port 59023 ist ein freier Port aus dem Bereich der dynamisch zugewiesenen Ports (auch Client-Ports genannt). Als Ziel-Socket verwendet der Client den Socket des Servers. Der Server antwortet (Nachricht 2) auf die Anfrage des Clients und verwendet als Quell-Socket den Socket 192.168.100.22:443 und als Ziel-Socket den Socket 192.168.100.100:59023. Da der Socket beim Client zur Laufzeit einmalig vergeben ist, kann der Client die empfangenen Daten der Verbindung zum Server zuordnen.

	Quell-IP-Adresse	**Quell-Port**	**Ziel-IP-Adresse**	**Ziel-Port**
Anfrage	192.168.100.100	59023	192.168.100.22	443
Antwort	192.168.100.22	443	192.168.100.100	59023

Nachrichtenaustausch zwischen Client und Server

(2) Dynamic Host Configuration Protocol

Das **Dynamic Host Configuration Protocol (DHCP – RFC 2131)** wird verwendet, um in einem IP-basierten Netzwerk die Konfiguration von Netzwerkelementen durchzuführen. Neben der IP-Adresse können u.a. auch die Netzwerkmaske, die IP-Adressen des DNS-Servers, Zeitservers oder Gateways konfiguriert werden. Das Protokoll ist eine Weiterentwicklung des Bootstrap-Protokolls (BOOTP). Diese Bezeichnung findet sich noch hin und wieder, z.B. bei dem Mitschnitt und der Analyse von Netzwerkverkehr (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.1).

Das Protokoll basiert auf dem Client-Server-Modell und der Protokollablauf wird in einem vierstufigen Verfahren (sog. 4-Way-Handshake) durchgeführt. Auf der Serverseite besteht die Möglichkeit, das Vergabeverfahren der Konfiguration mit den folgenden drei Einstellungen zu konfigurieren:

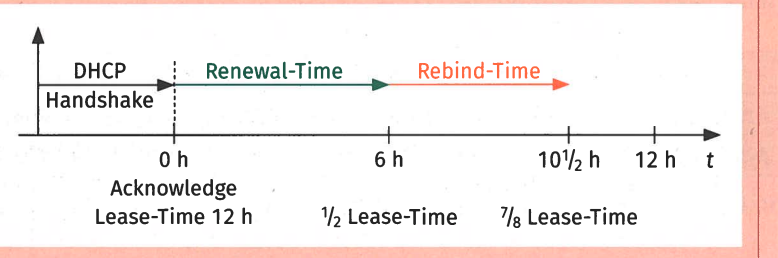
- manuelle Zuordnung,
- automatische Zuordnung,
- dynamische Zuordnung.

Adressvergabeverfahren bei DHCP

Manuelle Zuordnung (Static Allocation)
Bei der manuellen Zuordnung muss im Server eine Zuordnungsliste von IP- zu MAC-Adresse manuell konfiguriert werden. Konfigurierte MAC-Adressen erhalten bei der Anfrage die fest zugeordnete IP-Adresse zugewiesen.
Vorteil: Die Computer erhalten immer die gleiche IP-Adresse und nur bekannte MAC-Adressen erhalten IP-Adressen.
Nachteil: Jeder neue Computer, der eine IP-Adresse erhalten soll, muss erst manuell im DHCP-Server konfiguriert werden.

Automatische Zuordnung (Automatic Allocation)
Für die automatische Zuordnung wird im DHCP-Server ein IP-Adressbereich konfiguriert, der den Clients zugewiesen wird. Der Server merkt sich bei der Vergabe der IP-Adressen die zugehörige MAC-Adresse und reserviert die IP-Adresse nach der ersten Vergabe bis auf Weiteres für diese MAC-Adresse.
Vorteil: Auch hier erhalten Computer immer die gleiche IP-Adresse. Es muss keine manuelle Konfiguration neuer Computer durchgeführt werden.
Nachteil: Durch die feste Vergabe kann der Adressbereich vergeben sein und neue Computer erhalten dann keine Adresse mehr.

Dynamische Zuordnung (Dynamic Allocation)
Bei der dynamischen Zuordnung wird, wie bei der automatischen Zuordnung, ein IP-Adressbereich konfiguriert. Hier wird jedoch bei der Vergabe nicht die MAC-Adresse fest zugeordnet. Stattdessen wird eine sogenannte **Lease-Time** festgelegt, nach der die Vergabe der Konfiguration ausläuft. Mit dem Acknowledge teilt der Server dem Client mit, nach welcher Zeit seine Konfiguration die Gültigkeit verliert. Zusätzlich erhält er eine Renewal-Time und eine Rebind-Time.

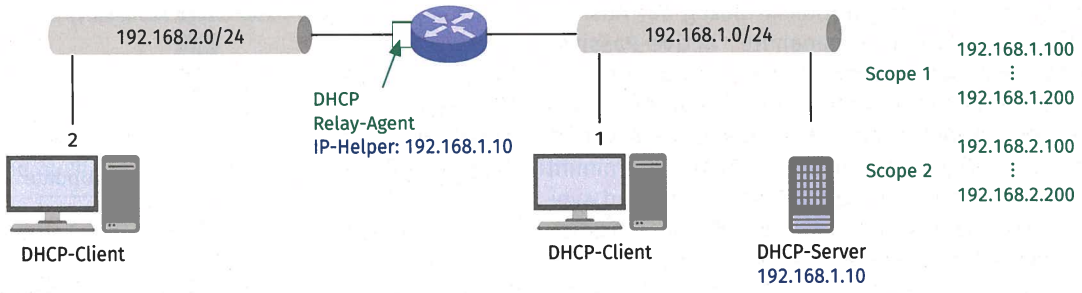


Renewal-Time = $^1/_2$ Lease-Time
Rebind-Time = $^7/_8$ Lease-Time

Nach der **Renewal-Time** beginnt der Client mit einem Request direkt (Unicast) an den Server gerichtet nachzufragen, ob er die Konfiguration verlängern kann. Der Server kann das mit einem Acknowledge bestätigen, damit wird die Lease-Time erneuert.
Wurde der Server beispielsweise abgeschaltet und antwortet nicht, beginnt der Client nach der **Rebind-Time** mit einem Broadcast einen Request in das Netz zu senden, um von einem anderen Server die Konfiguration bestätigt und verlängert zu bekommen. Findet das nicht statt, wird er nach der Lease-Time seine Konfiguration verwerfen und mit einem Discover wieder nach einer neuen Konfiguration fragen.

Neben der IP-Adresse erlaubt DHCP viele weitere Optionen (siehe RFC 2132 „DHCP Options and BOOTP Vendor Extensions“) zu konfigurieren. Typische Parameter sind z. B.:
- Gateway-Adresse,
- DNS-Server-Adresse,
- NTP-Server-Adresse.

DHCP-Relay
Um über Router-Grenzen hinweg die Konfiguration mithilfe von DHCP zu nutzen, wird die DHCP-Relay-Funktion genutzt. Im Router wird dazu ein Relay-Agent konfiguriert, der die DHCP-Pakete weiterleitet.



Beispiel für einen DHCP-Relay-Agent

Im Beispiel oben ist DHCP-Client 1 im gleichen Netz wie der DHCP-Server. Die Anfragen von DHCP-Client 2 werden nicht über den Router geleitet. Durch den Relay-Agent am Interface wird die Anfrage des DHCP-Clients 2 an den DHCP-Server geleitet. Im Relay-Agent wird die IP-Adresse des DHCP-Servers konfiguriert. Der Relay-Agent kennt das Netzwerk des angeschlossenen Interfaces und sendet die Netzadresse des Clients an den DHCP-Server. Für das Netzwerk des Clients ist ein eigener Adresspool konfiguriert, der als Scope bezeichnet wird. Der Server bietet also aus dem entsprechenden Scope des Interfaces eine IP-Adresse an.

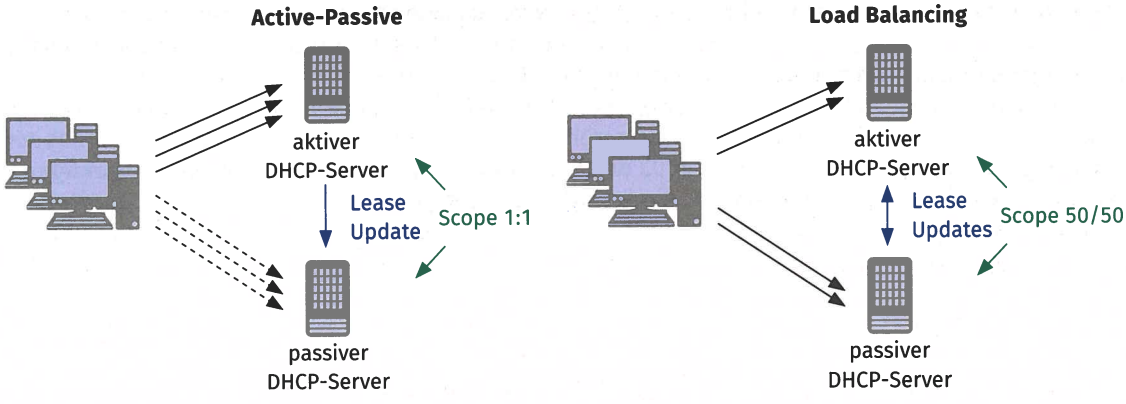
Ausfallsicherheit

Die Ausfallsicherheit kann z. B. durch zwei DHCP-Server erhöht werden. Dazu muss der zu vergebende Adressbereich auf die beiden Server aufgeteilt werden. Bei konkurrierendem **DHCP-Offer** der beiden Server wird der Client das ihn schneller erreichende Offer akzeptieren.

Die IETF (Internet Engineering Task Force, www.ietf.org) hatte an einer Protokollerweiterung gearbeitet, um Ausfallsicherheit zwischen DHCP-Servern zu ermöglichen (siehe auch https://datatracker.ietf.org/doc/html/draft-ietf-dhc-failover und https://datatracker.ietf.org/doc/html/rfc8156). Für DHCPv4 kam aus dieser Entwicklung kein RFC zustande. Trotzdem haben einige DHCP-Server die Funktion als proprietäre Implementierung erhalten. Zu finden ist die Funktion beispielsweise in der DHCP-Rolle des Windows-Servers, den DHCP-Server-Implementierungen von Cisco oder beim Kea-DHCP-Server. Für DHCPv6 wurde das Failover-Protokoll im RFC 8156 veröffentlicht.

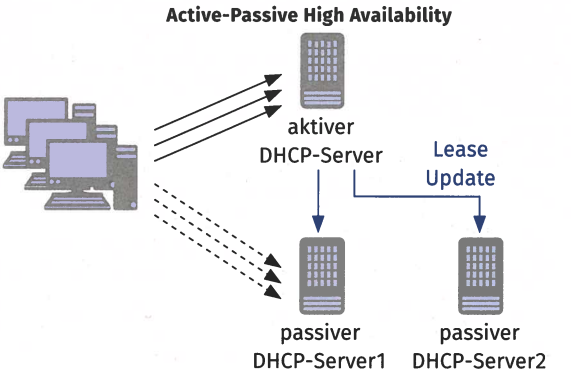
Der DHCPv6-Failover implementiert einen Protokollablauf, mit dem zwei DHCP-Server im Aktiv-Passiv-Modus (Hot-Standby-Model) betrieben werden können. Ein Server übernimmt den aktiven Modus und beantwortet DHCP-Anfragen von Clients. Über das Failover-Protokoll synchronisiert der aktive Server seinen Status mit dem passiven Server.

Ein Open-Source-DHCP-Server, der diese Funktionen implementiert, ist der Kea-DHCP-Server. Dieser Server unterscheidet die Failover-Funktion in Active-Passive und Load Balancing.



Unterschied zwischen Active-Passive und Load Balancing

Die einfachste Konfiguration in Bezug auf Ausfallsicherheit ist der sogenannte **Active-Passive-Modus**. Hier werden zwei Server im Aktiv-Passiv-Modus geschaltet. Beide Server bekommen den gleichen IP-Adressbereich als Vergabebereich (Scope) konfiguriert. Der aktive Server übernimmt die IP-Adressvergabe bei DHCP-Anfragen und synchronisiert seinen Status mit dem passiven Server. Fällt der aktive Server aus, übernimmt der passive Server die Aufgabe.



Active-Passive und High Availability

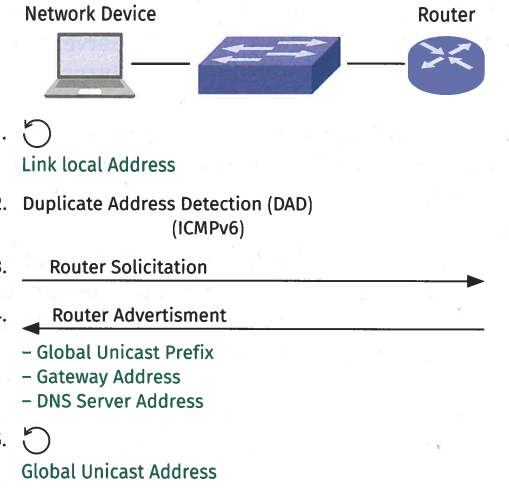
Diese Konfiguration kann für die Lastverteilung als **Load Balancing** geschaltet werden. Hierbei wird der Scope in den Verhältnissen 50/50, 80/20 oder 60/40 zwischen dem aktiven und dem passiven Server aufgeteilt. In dem Modus beantworten beide Server die DHCP-Anfragen der Clients. Dabei tauschen sie gegenseitig den Status der vergebenen Adressen aus. Fällt ein Server aus, übernimmt der andere dessen Scope.

Eine Steigerung des Active-Passive-Modus zu einer Hochverfügbarkeit ist durch die Nutzung von zwei oder mehreren passiven Servern, die alle mit dem aktiven Server synchronisiert werden, zu erreichen. Fällt der aktive Server aus, übernimmt der erste passive Server die Funktion und synchronisiert seine weitere Arbeit mit dem passiven Server 2.

DHCPv6

Stateless Address Auto Configuration (SLAAC) ist ein Mechanismus, mit dessen Hilfe sich ein Netzwerkknoten eine eigene IPv6-Adresse selbst zuweist. Da die Vergabe nirgends gespeichert wird, wie z. B. durch einen DHCP-Server, wird der Mechanismus als „stateless" bezeichnet. Der SLAAC-Mechanismus ist im RFC 4862 beschrieben.

Im ersten Schritt weist der Netzwerkknoten sich eine Link Local Address zu, mit der er nur im lokalen Netzwerk kommunizieren kann. Dadurch kann er aber andere Knoten erreichen und sendet ein **Router Solicitation (RS)**, um von einem Router ein **Router Advertisement (RA)** zu empfangen. Mit dem Router Advertisement erhält er das Präfix für eine **globale Unicast Address**, mit dessen Hilfe er sich eine eindeutige globale Adresse bildet.



Stateless Address Auto Configuration (SLAAC)

Die Nachricht Router Advertisement beinhaltet auch die Gateway-Adresse und in einer späteren Erweiterung (RFC 6106 wurde 2017 von RFC 8106 abgelöst) wurde 2010 auch die IP-Adresse des DNS-Servers ins Router Advertisement aufgenommen. Durch die zeitlich unterschiedliche Entwicklung wurde DHCP mit IPv6 nicht überflüssig. Daher gibt es seit 2003 den RFC 3315, in dem beschrieben ist, wie die Konfiguration von DHCP auch über IPv6 möglich ist. Die Konfiguration mit DHCP wird als **„stateful"** bezeichnet, da der DHCP-Server abspeichert, welcher Netzwerkknoten welche IP-Adresse zugewiesen bekommen hat.

(3) Domain Name System

Das **Domain Name System** dient zur Namensauflösung. Sind Server im Internet mit ihrer IP-Adresse erreichbar, ist für den menschlichen Nutzer ein aussagekräftiger Name besser verwendbar. So erhalten Server einen Namen und die zugehörige IP-Adresse wird vom Computer über eine DNS-Abfrage aufgelöst (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.1).

Das ursprüngliche System war eine Textdatei mit der Bezeichnung **„hosts"**, die auch heute noch verwendet wird. Unter Unix/Linux-Systemen liegt sie im Pfad „/etc". Bei Windows im Pfad „C:\Windows\System32\drivers\etc". Die Textdatei enthält pro Zeile einen Eintrag „IP-Adresse Hostname". Eine Beispieldatei kann folgendermaßen aussehen:

```
$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 antwerpen
8.8.8.8 dns.google.com

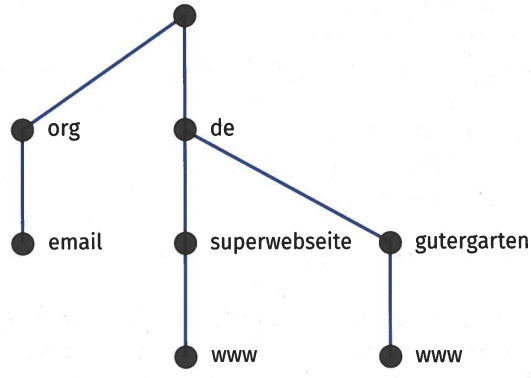
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Der Name „localhost" bekommt die IP-Adresse 127.0.0.1 zugeordnet. Der IP-Adresse 127.0.1.1 wird der Name „antwerpen" zugeordnet. Der Domänename „dns.google.com" wird der IP-Adresse 8.8.8.8 zugeordnet.

Jeder Computer, der eine DNS-Abfrage startet, durchsucht zuerst die eigene **Hosts**-Datei, bevor eine Anfrage an das DNS-System gestartet wird. Da dies einen möglichen Angriffspunkt darstellt, ist die Datei nur über das Administratorkonto editierbar.

Mit der Verbreitung des Internets war die Pflege einer einzelnen Datei nicht mehr sinnvoll und das System wurde auf ein hierarchisch verteiltes Datensystem mit Client-Server-Architektur ausgebaut. Ein Domänenname wird als **Fully Qualified Domain Name (FQDN)** bezeichnet und zeigt die Hierarchie des Systems.

Die Hierarchie wird durch Punkte getrennt, die Wörter der einzelnen Ebenen werden als Label bezeichnet. Zu beachten ist, dass nach der sogenannten **Top-Level-Domain „.de"** auch noch ein Punkt steht, der das Root-Label von dem Top-Level-Domain-Label abtrennt. Das Root-Label ist ein Leerzeichen und wird bei der Eingabe eines FQDN, z. B. im Browser, nie mit eingegeben.



Beispiel: FQDN www.superwebseite.de

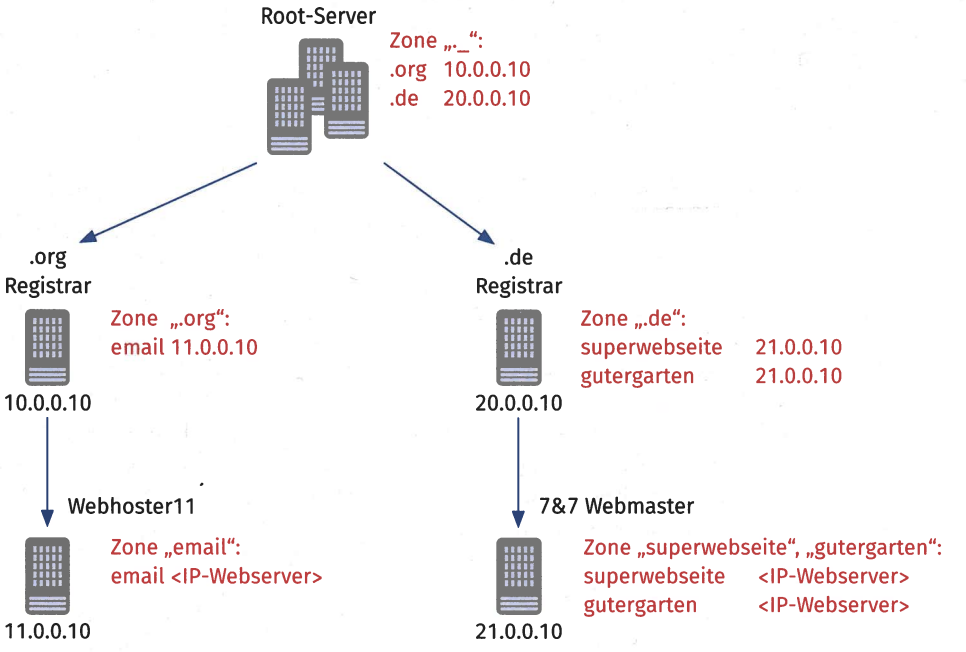
Abgrenzung FQDN von URL

Der Uniform Resource Locator (URL) ist eine Untermenge der Uniform Resource Identifier (URI). Eine URL beschreibt den Ort einer Ressource im Internet und ist folgendermaßen aufgebaut: „<scheme>:<scheme-specific-part>".

Beispiel: https://datatracker.ietf.org/doc/html/rfc1738
scheme: https – Nutze das Protokoll HTTPS
scheme-specific-part: datatracker.ietf.org/doc/html/rfc1738
→ datatracker.ietf.org – FQDN des Servers
→ /doc/html/rfc1738 – Pfad zum Dokument

Bei einer URL identifiziert der FQDN also den jeweiligen Server, auf dem die Ressource abgelegt ist. Hinzu kommt das Protokoll und der Pfad, wo die Ressource zu finden ist.

Umgesetzt wird die DNS-Hierarchie mit DNS-Servern, von denen jeder eine Autorität für einen bestimmten Bereich übernimmt. Die Autorität wird durch die **„Zones"** (Zonen) festgelegt. Die Abbildung zeigt die Umsetzung eines Teilbereichs der DNS-Hierarchie mit DNS-Servern.



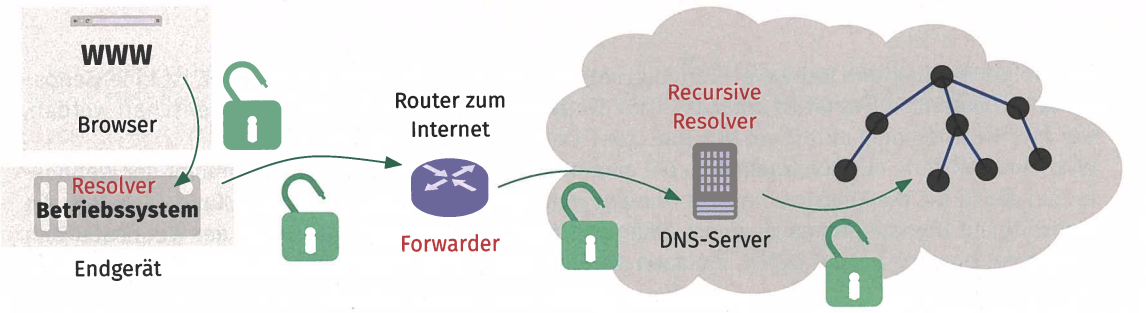
Umsetzung der DNS-Hierarchie mit Zonen

Die 13 **Root-Server** kennen die IP-Adressen der DNS-Server der Top-Level-Domänen (TLD). Die DNS-Server des jeweiligen TLD-Registrars kennen die DNS-Server der jeweiligen Domäne. Der TLD-Registrar für die „.de"-Domäne ist z. B. die DENIC. In dem obigen Beispiel sind die beiden FQDN „www.superwebseite.de" und „www.gutergarten.de" beim Webseitenhoster 7&7 gehostet. Der Hoster betreibt einen eigenen DNS-Server, in dem die IP-Adressen der **Webserver** für die jeweilige Domäne eingetragen sind. Der DNS-Server der DENIC („.de"-Registrar) wiederum verlinkt die beiden Webseiteneinträge mit der IP-Adresse des DNS-Servers des Webhosters.

„.de"-Registrar DENIC

Für die Verwaltung der Top-Level-Domain „.de" ist die DENIC zuständig. Die DENIC ist eine Genossenschaft mit Sitz in Frankfurt am Main. Laut Statuten kann Mitglied in der Genossenschaft werden, wer Internet-Domains unterhalb der Top-Level-Domain „.de" verwaltet.
Wird eine Webseite mit Domänenname bei einem Webseitenhoster bestellt, übernimmt der Webhoster bei der DENIC die Registrierung der Domäne und rechnet die Gebühr mit dem Kunden ab. Soll die Webseite mit Domänenname zu einem anderen Webhoster umziehen, beantragt der Webhoster den Umzug der Domäne bei der DENIC, die dann die Einträge in ihren DNS-Servern entsprechend anpasst.

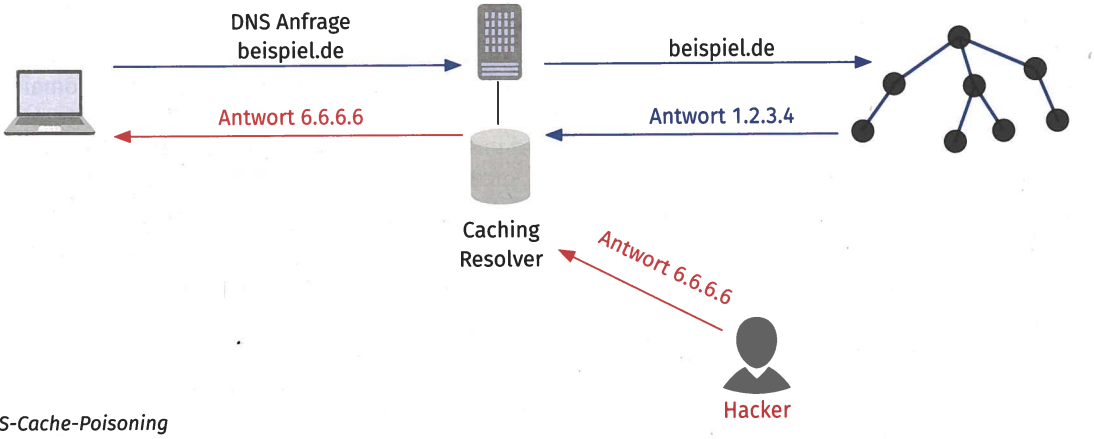
Erläuterungen zu DNS	
DNS-Root-Server	Root-Server stellen die oberste Ebene in der Hierarchie autoritativer DNS-Server dar. Bei einer DNS-Abfrage sind sie die erste Stelle und verweisen auf die entsprechenden DNS-Server der Top-Level-Domäne. Es gibt 13 DNS-Root-Server, deren IP-Adressen über Root Hint fest in DNS-Resolvern konfiguriert sind. Aus Sicherheit vor DDOS-Angriffen werden die DNS-Anfragen an die 13 Server über IP-Anycast weltweit verteilt.
Root Hint	Die 13 DNS-Root-Server haben festgelegte IPv4- und IPv6-Adressen. Jeder DNS-Server, der Abfragen an das DNS-System stellt, muss die IP-Adressen der Root-Server kennen. Diese werden als Root-Hint-Datei auf der Webseite der IANA zur Verfügung gestellt (siehe www.iana.org/domains/root/servers).
Autoritativer DNS-Server	Ein autoritativer DNS-Server enthält die DNS-Informationen unterhalb seiner Ebene in der DNS-Hierarchie. Die Root-Server kennen die IP-Adressen der Top-Level-Domain-Server, z. B. der DENIC, die für die Top-Level-Domain „.de" zuständig ist. Die DENIC-Server sind autoritativ für alle Domänennamen unter der Top-Level-Domain „.de" und kennen die IP-Adressen der entsprechenden DNS-Server. Die Konfiguration eines autoritativen DNS-Servers wird in einer sogenannten Zonen-Datei konfiguriert. Daher wird der autoritative Bereich auch Zone genannt.
DNS-Cache	Um wiederholte DNS-Abfragen zu reduzieren, hat z. B. der Resolver im Endgerät einen DNS-Cache, der durchgeführte Abfragen zwischenspeichert. Über das Kommando „ipconfig /displaydns" kann der Cache beispielsweise in Windows angezeigt werden. Nach der eingestellten Cache-Zeit, z. B. fünf Minuten, wird der Eintrag aus dem Cache gelöscht. Linux-Betriebssysteme speichern standardmäßig keine DNS-Abfragen.
DNS-Client	Der DNS-Client ist ein Endgerät, das eine Anfrage an einen DNS-Server stellt.
DNS-Resolver	DNS-Resolver ist eine Software, die DNS-Abfragen an einen DNS-Server weiterleitet. Ein Stub-Resolver wird beispielsweise in Endgeräten eingesetzt. Er hat eine eingeschränkte Abfragefunktionalität und leitet seine Abfrage in der Regel an einen Recursive-Resolver weiter. Ein Recursive-Resolver fragt die ganze DNS-Hierarchie nach einer Abfrage ab, bis er das Ergebnis hat und es dem abfragenden Client zurückliefert. Wenn der Recursive-Resolver auch das Abfrageergebnis in einem Cache speichert, ist er ein Full-Service-Resolver.
Forward-Lookup	Beim Forward-Lookup findet die Abfrage der IP-Adresse zu einem gegebenen Domänennamen statt.
Reverse-Lookup	Beim Reverse-Lookup findet die Abfrage des Domänennamens zu einer gegebenen IP-Adresse statt.
Forwarder	Ein Forwarder ist ein DNS-Server, an den Anfragen weitergeleitet werden, wenn ein Server die an ihn gerichtete Anfrage nicht selber beantworten kann.



Ablauf einer Namensauflösung

Die Abbildung zeigt eine Namensauflösung, ausgelöst durch die Eingabe einer URL im Browser. Die grünen Schlösser beschreiben unverschlüsselte Verbindungen. Der Computer als Endgerät ist der DNS-Client, in dem ein Stub-Resolver implementiert ist. Der Browser leitet die Namensauflösung an den Stub-Resolver. Dieser leitet die Anfrage an den Forwarding-DNS-Server weiter, der im Internet-Router implementiert ist. Der Forwarding-DNS-Server schickt seinerseits die Anfragen an einen Recursive-Resolver weiter. Der Recursive-Resolver, oft vom Netzbetreiber des Internetzugangs betrieben, führt die rekursive Abfrage bei den autoritativen DNS-Servern durch. In der Abbildung nicht eingezeichnet ist die Rückantwort, die an das Endgerät zurückgegeben wird.

DNS-Sicherheit



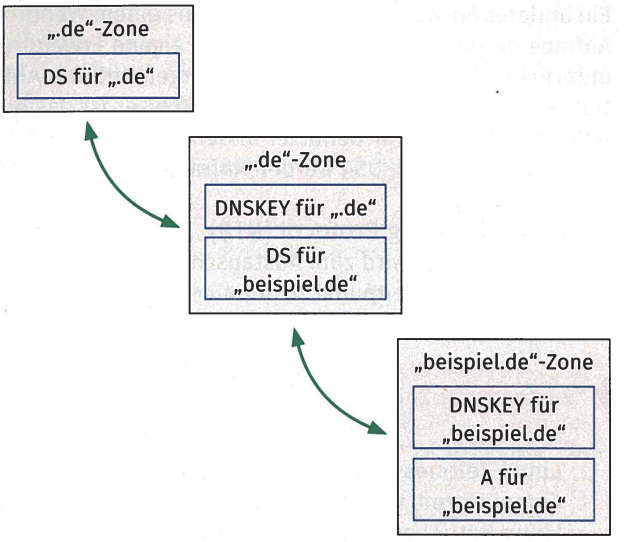
DNS-Cache-Poisoning

Bevor eine Webseite oder irgendein anderer Server angesprochen werden kann, muss der Domänenname über das DNS aufgelöst werden, damit der Server über seine IP-Adresse angesprochen werden kann. Hier ergeben sich verschiedene Angriffspunkte, um die endgültige Anfrage auf einen falschen Server umzuleiten, oder auch einfach datenschutzrechtliche Bedenken, da die herkömmlichen Abfragen alle unverschlüsselt stattfinden. Somit bedeutet jeder Zugriff auf die Netzwerkinfrastruktur auch einen Zugriff auf die gemachten Anfragen. Hier sind beispielsweise unverschlüsselte WLANs zu nennen oder auch Staaten mit eingeschränkten Freiheitsrechten.

Die Abfrage des Resolvers an die autoritativen DNS-Server erfolgt über UDP, einer verbindungslosen Übertragungsart. Ein Angriff aus dem Netz ist das sogenannte **Cache-Poisoning**. Hierbei wird an den Resolver von einem Angreifer eine Antwort mit einer gefälschten IP-Adresse gesendet. Der Resolver schreibt diese Information in seinen Cache. Für die Cachezeit werden alle zukünftigen Namensauflösungen für diese Domäne mit der gefälschten IP-Adresse beantwortet, unter der der Angreifer einen eigenen Server betreibt, der für den Angriff verwendet wird.

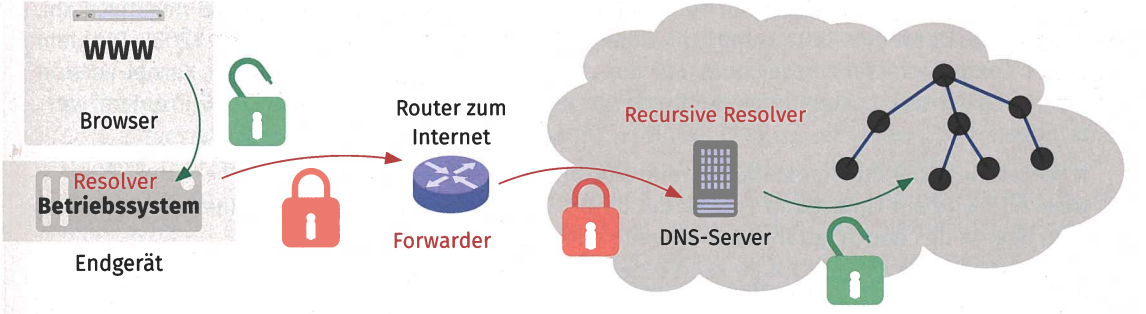
Um dem **Cache-Poisoning** vorzubeugen, wurde der DNS-Standard um **DNSSEC** erweitert. Mithilfe von DNSSEC kann der Resolver überprüfen, ob die Antwort wirklich vom angefragten DNS-Server kam.

Für jeden Zoneneintrag in einem autoritativen DNS-Server wird der Eintrag signiert. Neben der Signatur wird auch der öffentliche Schlüssel, als DNSKEY-Record, mit in der Zonendatei abgelegt. Der Hash-Wert des öffentlichen Schlüssels wird im übergeordneten Server der DNS-Hierarchie als sogenanntes DS-Record abgelegt. Dadurch wird eine sogenannte **Chain-of-Trust** erzeugt. Der abfragende Resolver kann jetzt über den öffentlichen Schlüssel die Signatur überprüfen. Den öffentlichen Schlüssel kann er über eine Abfrage beim übergeordneten DNS-Server überprüfen.

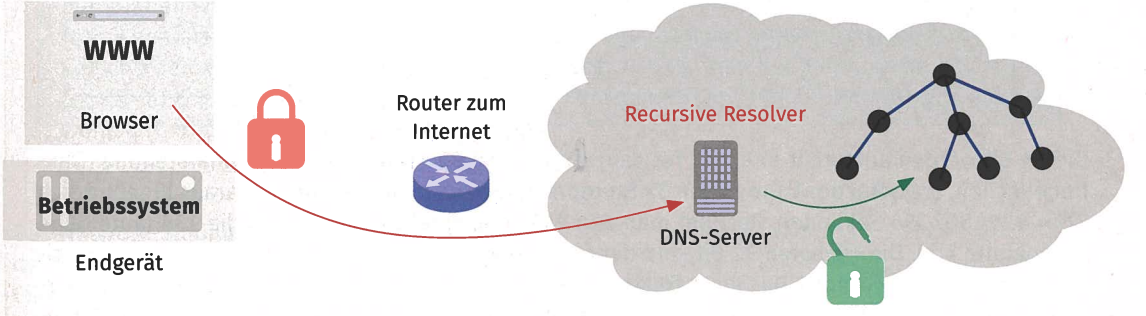


Beispiel für DNSSEC in der Zonenkonfiguration

Obwohl die RFCs für DNSSEC bereits in den 2000er-Jahren veröffentlicht wurden, hat sich die Technik nicht vollständig durchgesetzt. DNSSEC zielt darauf ab, die Abfrage des Recursive Resolvers an die autoritativen DNS-Server abzusichern. Zwei Verbesserungen, die auf die Sicherheit der Abfrage vom DNS-Client zum Recursive Resolver abzielen, sind **DNS over TLS** (DoT) und **DNS over HTTPS** (DoH).



Mit DNS over TLS (DoT) verschlüsselte DNS-Anfrage



Mit DNS over HTTPS (DoH) verschlüsselte DNS-Anfrage

Beim **DNS over TLS** wird die DNS-Anfrage vom DNS-Client, also dem Resolver auf dem Endgerät, zum Recursive Resolver, also dem Resolver beim ISP oder Cloud-Provider, mit TLS verschlüsselt. Ziel ist es hier, die Abfrage gegen unbeabsichtigtes Mitlesen, z. B. über ein unverschlüsseltes WLAN, oder jeglichen anderen Zugriff auf das Netzwerk bis zum Recursive Resolver zu unterbinden.

Ein anderer Ansatz für DNS-Anfragen aus einem Webbrowser ist das **DNS over HTTPS**. Hier wird die DNS-Anfrage direkt vom Browser über eine eigene Erweiterung durchgeführt und nicht an den Resolver des unterliegenden Betriebssystems weitergeleitet. Die Anfrage wird über HTTPS an einen voreingestellten DoH-Provider geleitet. Beim Chrome-Browser ist das ein eigener DNS-Server von Google. Firefox hat im Jahr 2022 DoH nur für Benutzer in den USA, Kanada, Russland und der Ukraine aktiviert. Weitere Länder sollen folgen. In den USA werden Anfragen an Cloudflare gestellt.

(4) Network Timing Protocol (NTP)

Das Protokoll NTP wird zum Austausch von exakten Zeitpunkten genutzt. Es gibt allerdings unterschiedliche Arten, einen Zeitpunkt zu definieren. Im Folgenden werden die wichtigsten Systeme zur Zeitmessung betrachtet.

Systeme zur Zeitmessung

Linux-Zeitmessung mit timestamp (32-Bit seit 1. Januar 1970)

Computerzeit wird häufig als Zahlenwert basierend auf einem Anfangsdatum gespeichert. Im Linux-Betriebssystem ist das ein 32-bit-signed-Integer-Wert seit dem 1. Januar 1970. Der 1. Januar 1970 wird als **epoch** bezeichnet. Es ist der Startzeitpunkt, auf den sich der **timestamp** bezieht. Da es sich um einen vorzeichenbehafteten Wert handelt (signed Integer), sind positive Werte die Sekunden nach dem 1. Januar 1970 und negative Werte entsprechend davor.

Ein Wert von 3600 beschreibt also 3600 Sekunden nach dem 01.01.1970, also 01:00 Uhr am 01.01.1970 in der Zeit Coordinated Universal Time (UTC). Der Wert $24 \cdot 60 \cdot 60 = 86400$ Sekunden nach dem 01.01.1970, also 00:00 Uhr 02.01.1970 (UTC).

Mit dem Zahlenwert können Zeiten bis zum 19. Januar 2038 dargestellt werden. Danach würde für den „signed Integer" ein Überlauf stattfinden und dieser repräsentiert dann eine negative Zahl, die dem 13. Dezember 1901 entspricht. Das Problem des Überlaufs wird auch als Y2038-Problem in der Computertechnik bezeichnet. Für das „Jahr-2038-Problem" wurde mit der Kernel-Version 5.10 ab 2020 eine Lösung implementiert, die das Problem zu einem „Jahr-2486-Problem" verschiebt.

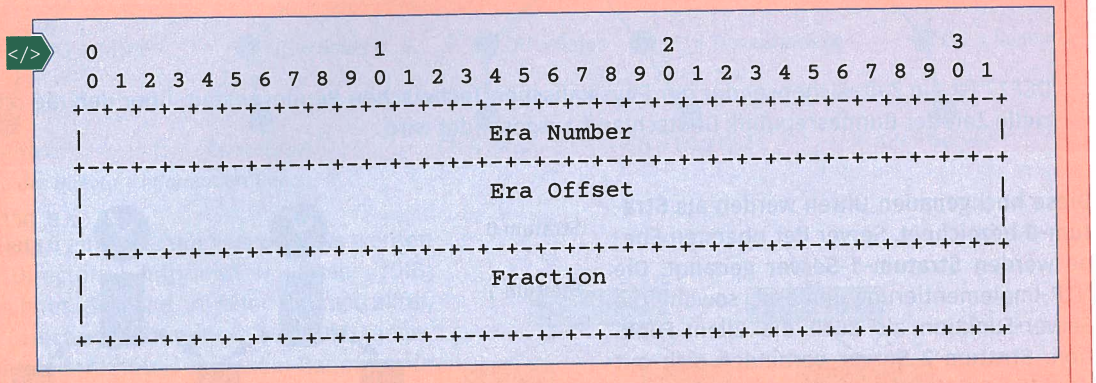
NTP-Zeitmessung im Timestamp-Format (64-Bit)

Beim NTP wird ein Timestamp-Format mit 64-Bit-Datenlänge verwendet. Der Timestamp wird seit dem Beginn des epoch gezählt. Der aktuelle epoch begann am 1. Januar 1900.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Seconds                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Fraction                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Die 64 Bit werden aufgeteilt in 32 Bit für ganzzahlige und 32 Bit für Nachkommadarstellung (fraktioneller Teil, engl. fraction). Die ersten 32 Bit repräsentieren ganze Sekunden. Daraus ergeben sich $2^{32} = 4294967296$ Sekunden. Rechnet man den Wert um, ergeben sich daraus die 136 Jahre (eine sogenannte Era). Die nächsten 32 Bit sind der fraktionelle Teil. Hier wird die Wertigkeit mit negativen Exponenten berechnet. Das erste fraktionelle Bit ist $2^{-1} = \frac{1}{2^1} = \frac{1}{2} = 0{,}5$. Das 32. Bit hat die Wertigkeit $2^{-32} = \frac{1}{2^{32}} = 232{,}83$ Pikosekunden. Mit diesem Wert wird die Genauigkeit festgelegt. Das **Timestamp-Format** wird für den Austausch zwischen dem NTP-Client und dem NTP-Server verwendet.

NTP-Zeitmessung im Datestamp-Format (128-Bit)
Das zweite Format ist das **Datestamp-Format**. Es wird im Client und Server für die Verarbeitung verwendet. Hier werden die **Era Number** mit „32-Bit signed Integer", die Sekunden seit Beginn der Era (**Era Offset**) als „64-Bit signed Integer" und schließlich noch 64-Bit **Fraction** (kleinste Zeiteinheit) mit den Nachkommastellen zu den Sekunden gespeichert. Zusammen werden für die Speicherung 128 Bit verwendet.

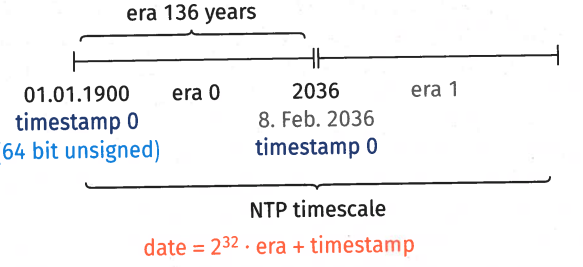


Die **Era Number** wird mit Nummern festgelegt. Die Ur-Era ist null und startete am 1. Januar 1900. Ihr folgt die Era 1, die am 8. Februar 2036 startet. Die Era vor dem 1. Januar 1900 ist die Era-1.

Beispiel:
- Era 0: 01.01.1900
- Era 1: 08.02.2036
- Era -3: 15.10.1582

Durch die **Era Number** ist NTP gegenüber Überläufen der Zähler, bei denen der Zeitwert auf den Anfangswert zurückspringt, gesichert.
Um beispielsweise log-Einträge von Computern zu vergleichen, die an verschiedenen Stellen der Erde aufgestellt sind, wird die Computerzeit auf **Coordinated Universal Time (UTC, koordinierte Universalzeit)** eingestellt. Hier gab es in der Abkürzungsfindung zwischen verschieden Sprachen einen Konflikt. Als Kompromiss wurde die Abkürzung auf UTC festgelegt, obwohl die englische Abkürzung zur Beschreibung CUT wäre. Der Begriff UTC wurde erst 1972 eingeführt. Zuvor wurde der Begriff Greenwich Meant Time (GMT) verwendet, da der Ursprung der Zeitmessung in der Sternwarte von Greenwich, einem Vorort von London in Großbritannien, stattfand. Der Nullmeridian ist eine Linie vom Nordpol bis zum Südpol, die durch die Sternwarte von Greenwich verläuft. Basierend auf diesem Meridian werden die Zeitzonen festgelegt, die jedoch nicht stur an Meridianen wechseln. So wird die Mittel-Europäische-Zeit (MEZ) von Spanien bis Polen und von Norwegen bis Italien verwendet. Portugal, Irland und Großbritannien verwenden die West-Europäische-Zeit (WEZ). Die Mittel-Europäische-Zeit ist UTC+1. Dies bedeutet, wenn UTC = 12:00 Uhr ist, ist MEZ = 13:00 Uhr. Die Mittel-Europäische-Sommer-Zeit (MESZ) ist UTC+2. Dies bedeutet, wenn UTC = 12:00 Uhr ist, ist MESZ = 14:00 Uhr.

Das **Network Timing Protocol (NTP)** ist eines der ältesten Protokolle im Internet. Nach dem Client-Server-Prinzip erlaubt es die Verteilung von Zeitinformationen im Netzwerk. Die Zeiteinstellungen finden dazu parallel zu der in Computern üblichen Hardwarezeit statt. Das bedeutet, ein Computer startet erst mit der Hardwarezeit, bis der NTP-Client startet und über einen NTP-Server die Zeit abfragt. Von da an arbeitet er mit der durch NTP eingestellten Zeit weiter.



Era verhindert Zeitüberlauf

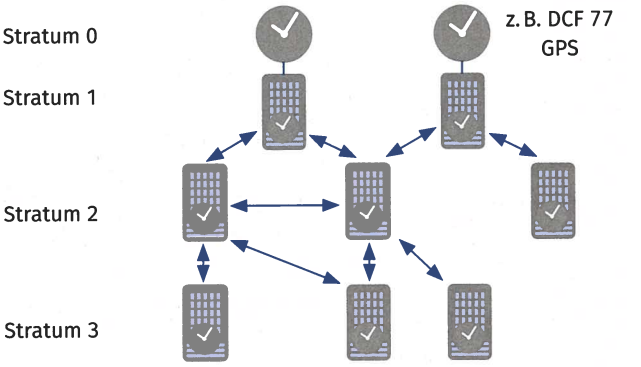
Aktuell ist NTP in der Version 4 (RFC 5905 Network Time Protocol Version 4: Protocol and Algorithms Specification) von der IETF spezifiziert.

Das System der Zeitserver ist in einer Hierarchie angeordnet. Die oberste Ebene sind Server, die mit hochgenauen Hardwareuhren versehen sind. Das können Uhren z. B. auf Basis eines GPS- oder DCF77-Signals sein.

DCF77

DCF77 ist ein Zeitzeichensender der Physikalischen Technischen Bundesanstalt, über den die offizielle Zeit der Bundesrepublik Deutschland ausgesendet wird.

Diese hochgenauen Uhren werden als **Stratum-0** bezeichnet. Server der obersten Ebene werden **Stratum-1-Server** genannt. Die NTP-Implementierung umfasst sowohl die Server-Funktion als auch die Client-Funktion. **Stratum-2-Server** verbinden sich mit der Client-Software zu Stratum-1-Servern. Sollte die Verbindung gestört sein, können sie sich mit Servern auf der gleichen Stratum-Ebene verbinden, um einen Zeitabgleich durchzuführen. Je höher die Stratumzahl, desto entfernter ist der Server von der hochgenauen Uhr und umso ungenauer ist das Zeitsignal. Das Protokoll misst die Paketlaufzeiten zwischen Client und Server und korrigiert die empfangene Zeit damit. Eine vereinfachte Variante ist das Simple Network Timing Protocol (SNTP). Eine Implementierung verlangt nur nach einem Upstream-Server (vgl. Abbildung Server-Hierarchie) und keine abhängigen Clients.



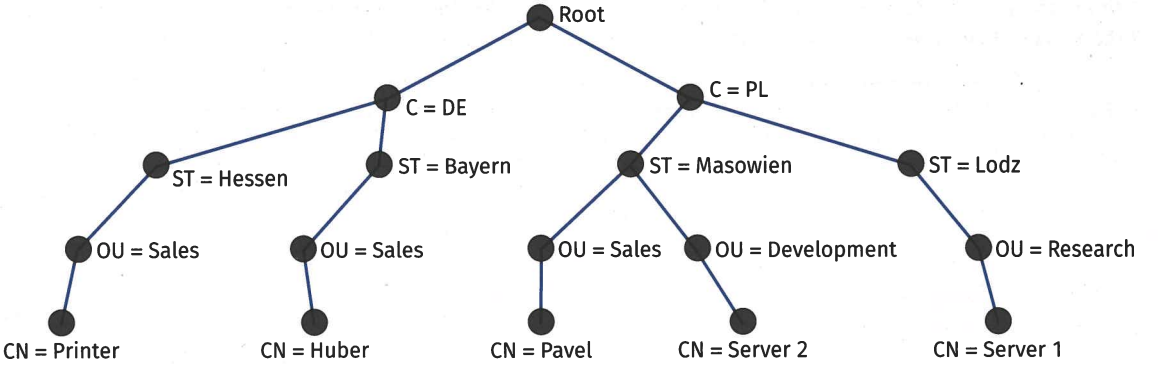
Server-Hierarchie für das NTP-Protokoll

(5) Verzeichnisdienste

Die ITU-T hat in ihrer Empfehlung X.500 Verzeichnisdienste in Netzwerken beschrieben. Die einfachste Vorstellung eines Verzeichnisdienstes ist die eines elektronischen Kontaktinformationsservers, von dem beispielsweise die Informationen von Mitarbeitern einer Organisation mit Telefonnummer und E-Mail-Adresse abrufbar sind. Ein Verzeichnisdienst geht aber über diese einfache Funktion hinaus und nimmt auch die Netzwerkressourcen auf. Wenn Mitarbeiter und Ressourcen in dem Verzeichnisdienst aufgenommen sind, können Zuordnungen und Zugriffsrechte darüber verwaltet werden.

Folgende Erklärung beschreibt zunächst allgemein, wie ein Verzeichnisdienst aufgebaut ist, und konzentriert sich dann auf den von Microsoft umgesetzten Verzeichnisdienst **Active Directory**.

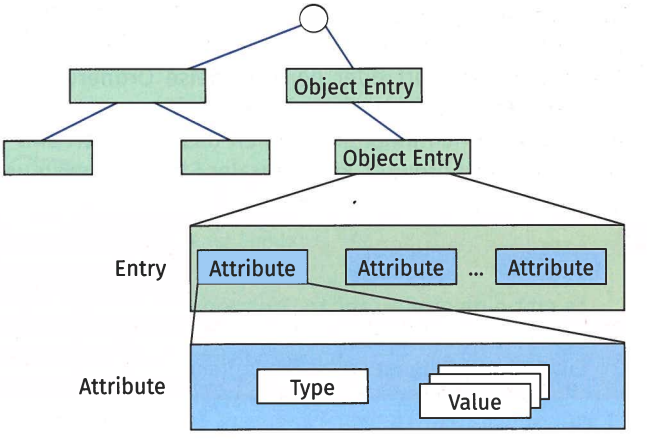
Verzeichnisstruktur
Ein Verzeichnisdienst ist nicht als relationale Datenbank aufgebaut, sondern in einer hierarchischen Struktur. Die Daten werden mithilfe eines objektorientierten Datenmodells abgebildet und gespeichert. Da der Verzeichnisdienst geografisch verteilte Informationen enthalten kann, ist es möglich, die Daten über entsprechend verteilte Server zur Verfügung zu stellen.



Aufbau des Directory Information Tree

Alle Daten eines Verzeichnisdienstes werden als Directory Information Base (DIB) bezeichnet. Sie sind in einer Baumstruktur, dem Directory Information Tree (DIT), angeordnet. Jeder Knoten in diesem Baum ist ein Object Entry. Jeder dieser Einträge enthält ein oder mehrere Attribute. Ein Attribut wiederum besteht aus einem Type/Value-Paar.

Folgende Attributstypen werden verwendet:
- cn – common name
- sn – surname
- ou – organizational unit
- st – state
- c – country
- mail – E-Mail-Adresse



Aufbau des Object Entry

Protokoll

Die Abfrage im Verzeichnis findet nach dem Client-Server-Modell statt. Der Client stellt Anfragen an den Server über das Lightweight-Directory-Access-Protocol (LDAP) (RFC 4510, RFC 4511 und RFC 4532).

Active Directory

Das Active Directory (AD) ist ein Verzeichnisdienst im Windows-Netzwerk. Dieser Verzeichnisdienst erfüllt zwei Aufgaben:

1. Auf der administrativen Seite stellt er, wie alle Verzeichnisdienste, sämtliche Funktionen zur Verfügung, mit denen die Ressourcen des Netzwerks organisiert, verwaltet und gesteuert werden können.
2. Auf der Benutzerseite regelt er die Zugriffe auf die Ressourcen des Netzwerks.

Hier ist anzumerken, dass die Verwaltung nur in Bezug auf Windows-Betriebssysteme funktioniert. Zwar gibt es mit dem Samba-Projekt eine auf Linux basierte Implementierung eines Domain Controllers und ab der Version 4 auch eine Implementierung des Active Directory, jedoch können damit keine Linux-Clients verwaltet werden.

Leistungsmerkmale des Active Directory sind die zentrale Verwaltung aller Netzwerkressourcen durch eine einheitliche Verwaltungsschnittstelle und eine transparente Darstellung der Ressourcen für die Benutzer. Dazu führt der Verzeichnisdienst eine logische Sicht auf die physikalischen Gegebenheiten ein. Dadurch erlaubt er die Abbildung von Firmen- oder Verwaltungsstrukturen unabhängig von der Netzwerktopologie. Die Abbildung von physikalischen Gegebenheiten, wie Lokation und Netzwerktopologie, sowie logischen Einheiten, z. B. Firmen- oder Verwaltungsstruktur, erfolgt im Active Directory über die folgenden Komponenten:

Domäne (Windows-Domäne)	Eine Windows-Domäne ist die eigentliche und wesentliche Einheit von Active Directory. Eine Domäne stellt eine einzelne Verwaltungseinheit dar.
Organisationseinheit (OU)	Eine Organisationseinheit ist ein Container, der dazu dient, Objekte zu gruppieren. Eine OU erlaubt die Delegation von Verwaltungsaufgaben.
Objekt	Ein Objekt ist die kleinste Einheit, um Netzwerkressourcen abzubilden. Objekte sind Geräte, Dienste, Datenbestände, Benutzerkonten und Organisationseinheiten.

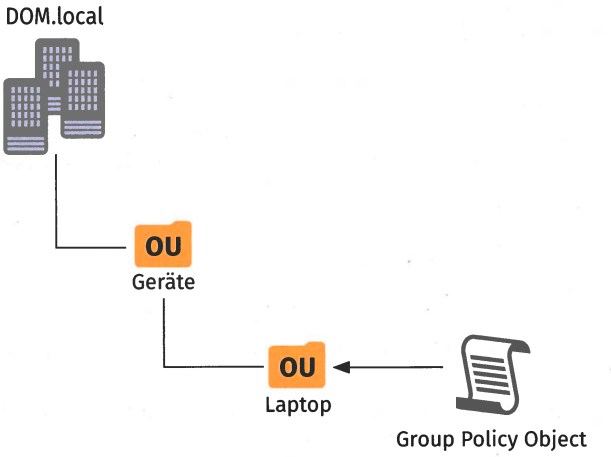
Gruppenrichtlinien

Gruppenrichtlinien sind auf allen Windows-Betriebssystemen vorhanden. Sie können lokal oder zentral über die Domänenverwaltung vom Active Directory konfiguriert und dann auf die Clientcomputer übertragen werden. Gruppenrichtlinien sind Sammlungen von Benutzer- und Computerkonfigurationseinstellungen, die mit Computern, Standorten, Domänen oder Organisationseinheiten verknüpft werden, um das Verhalten des Benutzerdesktops zu steuern. Darüber hinaus werden in ihnen Aspekte, wie Sicherheitseinstellungen, An- und Abmeldeskripte sowie Skripte für Starten sowie Herunterfahren eines Computers, definiert oder beispielsweise Ordnerumleitungen festgelegt. Mit Gruppenrichtlinien kann das Verhalten des Betriebssystems bestimmt und dessen Optionen eingeschränkt werden. Es gibt aber auch Gruppenrichtlinien, mit denen das Verhalten und die Optionen von Anwendungen, z. B. Microsoft Office oder Webbrowser, von zentraler Stelle aus gesteuert werden können.

Gruppenrichtlinieneditor

Es gibt eine große Zahl an Gruppenrichtlinien. Über den Gruppenrichtlinieneditor und die zugehörige Hilfe einer Gruppenrichtlinie können die vorhandenen Gruppenrichtlinien erkundet werden. Eine gute Quelle ist auch die deutschsprachige Webseite www.gruppenrichtlinien.de.

Unter einem **Group Policy Object (GPO)** versteht Microsoft eine Zusammenstellung von Gruppenrichtlinieneinstellungen. GPO betreffen Computer (bzw. Computergruppen) oder Benutzer (bzw. Benutzergruppen) an Standorten, in einzelnen Domänen oder in Organisationseinheiten. Das bedeutet beispielsweise, dass eine OU mit der Bezeichnung „Laptops" erstellt werden kann. Für diese OU wird ein neues Gruppenrichtlinienobjekt angelegt und in diesem GPO werden all diejenigen Richtlinien definiert, die nur Laptops betreffen. Andere Systeme mit einer ständigen Netzwerkverbindung sind nicht von diesen Richtlinien betroffen. Die für dieses GPO geltenden Richtlinieneinstellungen wirken sich auf alle Computerobjekte aus, die in die OU „Laptops" verschoben werden.



Anwendung eines Group Policy Object (GPO)

Reihenfolge der Richtlinienvererbung

Wenn mehrere OUs ineinander verschachtelt sind, so wirkt die Gruppenrichtlinie einer OU nicht nur auf deren Computer- oder Benutzerobjekte, sondern über die Vererbung auch auf Objekte, die in untergeordneten OUs liegen. Gibt es beispielsweise eine OU „Standort Frankfurt" mit den Sub-OU „Vertrieb Frankfurt", „Produktion Frankfurt" und „Einkauf Frankfurt" und sind für die OU „Standort Frankfurt" Gruppenrichtlinien definiert, so wirken diese auf alle Objekte in tiefer gelegenen OUs.

Wenn für die Sub-OU „Einkauf Frankfurt" ein neues GPO erstellt wird und dort eine Gruppenrichtlinie andere Einstellungen vornimmt als in der übergeordneten Gruppenrichtlinie schon festgelegt wurde, so

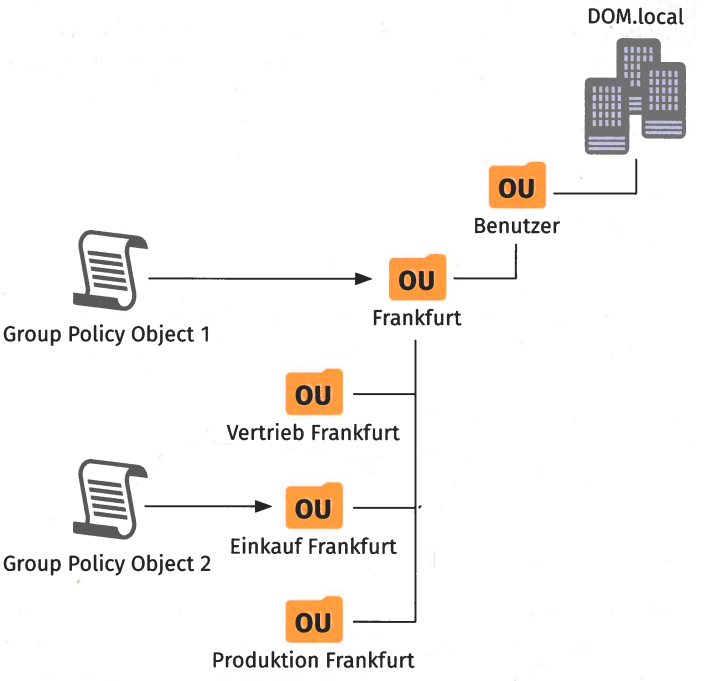
überschreibt die untergeordnete Gruppenrichtlinie die vorgegebenen Einstellungen. Richtlinien in beiden OUs, die nicht im Widerspruch zueinander stehen, werden kumulativ übernommen. Es werden also alle Einstellungen in Summe angewendet.

Aktiviert, nicht konfiguriert, deaktiviert

Wird eine Richtlinie auf **aktiviert** gestellt, so wird sie auf alle Objekte der OU angewendet. Ist eine Richtlinie im Zustand **nicht konfiguriert**, findet sie keine Anwendung. Durch die Vererbung kann die Richtlinie in einer übergeordneten OU aktiviert sein und findet dann auch an dieser Stelle Anwendung.

Wenn die Richtlinie auf **deaktiviert** gesetzt wird, macht sie genau das Gegenteil ihrer beschriebenen Funktion.

Die Vererbung kann unterbunden werden, wenn in einer übergeordneten Richtlinie eine Funktion aktiviert ist und untergeordnet nicht deaktiviert werden soll. Dann kann die Richtlinie auf **kein Vorrang** gesetzt werden.



Reihenfolge der Richtlinienvererbung

Wirksamkeit von Gruppenrichtlinien

Gruppenrichtlinien für Benutzerobjekte wirken nach Anmeldung des entsprechenden Benutzers. Für Computerobjekte wirken die Richtlinien, nachdem der Rechner neu gestartet wurde. Es gibt den Kommandozeilenbefehl „gpupdate", über den alle Gruppenrichtlinien unverzüglich erneut angewendet werden. Der Parameter „/force" sorgt dafür, dass alle Einstellungen erneut angewendet werden, auch wenn die Richtlinie nicht verändert wurde.

Fehlersuche, wenn eine Richtlinie nicht wirkt

Grundsätzlich kann es von Umgebung zu Umgebung verschiedene Ursachen dafür geben, dass eine aktivierte Gruppenrichtlinie auf dem betreffenden Client oder unter der betreffenden Kennung nicht das gewünschte Resultat zeigt. Folgende Ansatzpunkte können bei der Fehlersuche helfen:

- Verbindung von Client-Server überprüfen: Bei einem Ping-Test ist eine aktive Firewall zu beachten, die den Ping unterbinden kann.
- Funktionierende Namensauflösung vom Client mithilfe von „nslookup" überprüfen.
- Im Active Directory überprüfen, ob das Computer- oder Benutzerobjekt in der richtigen OU mit der angehefteten Gruppenrichtlinie liegt.
- In der Ereignisanzeige des Clients nach entsprechenden Fehlermeldungen suchen.
- In der Ereignisanzeige des Servers nach entsprechenden Fehlermeldungen suchen.
- Mithilfe des Kommandozeilentools „gpresult" im Client überprüfen, welche Gruppenrichtlinien wirken.

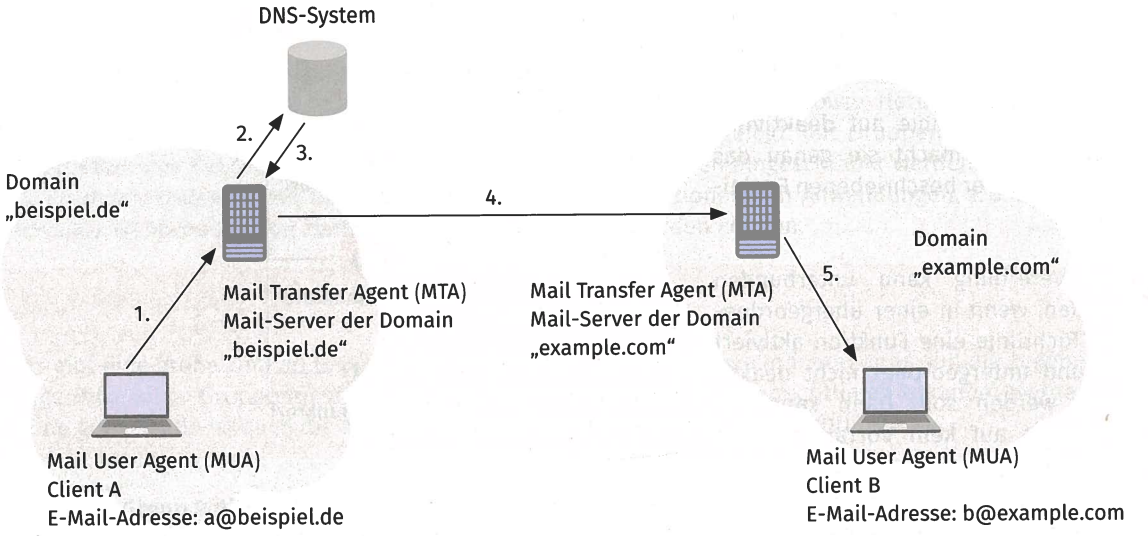
(6) E-Mail-System

E-Mail-System

Ein E-Mail-Server bietet einem E-Mail-Client die Möglichkeit E-Mails (Electronic Mail) zu versenden und zu empfangen. Dabei kommen verschiedene Systeme und Protokolle zum Einsatz.

Für das Versenden und Empfangen sind der E-Mail-Server und der E-Mail-Client nur Teilsysteme. Es ist sinnvoll, das Gesamtsystem inklusive der verwendeten Protokolle anhand eines Beispiels zu betrachten.

Beispiel:



E-Mail-Systeme im Überblick

Der Client A mit der E-Mail-Adresse „a@beispiel.de“ möchte an den Client B mit der E-Mail-Adresse „b@example.com“ eine E-Mail versenden.

Dazu wird auf dem Client mit einem E-Mail-Programm, auch Mail User Agent (MUA) genannt, die E-Mail verfasst und abgesendet (Nachricht 1). Der MUA sendet die E-Mail an den E-Mail-Server der eigenen Domain, im Beispiel den E-Mail-Server der Domain „beispiel.de“. Der eigene E-Mail-Server wird Mail Transfer Agent (MTA), manchmal auch Mail Submission Agent (MSA), genannt. Beim Senden kommt das **Simple Mail Transfer Protocol Secure (SMTPS)** oder das **Simple Mail Transfer Protocol (SMTP)** zum Einsatz.

```
                -$ nslookup -type=mx beispiel.de
Server:         192.168.178.1
Address:        192.168.178.1#53

Non-authoritative answer:
beispiel.de     mail exchanger = 10 mx00.udag.de.
beispiel.de     mail exchanger = 20 mx01.udag.de.

Authoritative answers can be found from:
```

DNS-Namensauflösung des E-Mail-Servers

Die E-Mail liegt jetzt beim eigenen E-Mail-Server. Dieser muss die E-Mail zum MTA der Empfänger-Domain senden. Dazu wertet der MTA den Domain-Anteil der Ziel-E-Mail-Adresse aus. Im Beispiel muss die E-Mail an den E-Mail-Server der Domäne „example.com“ weitergeleitet werden. Dazu benötigt der MTA die IP-Adresse des E-Mail-Servers von „example.com“. Um den Domain-Namen in eine IP-Adresse umzuwandeln, wird das DNS-System verwendet. Der MTA des Empfängers fragt nach dem **MX-Record** (Mail Exchange-Record) der Empfänger-Domäne (Nachricht 2). Der MX-Record ist die Zuordnung zwischen dem Domain-Namen des E-Mail-Servers und dessen IP-Adresse. Das DNS-System sendet als Antwort die IP-Adresse des empfangenden E-Mail-Servers der Domain „example.com“ (Nachricht 3).

Jetzt kennt der MTA die IP-Adresse des E-Mail-Servers der Domäne „example.com“ und leitet die E-Mail weiter. Dabei kommt wieder das Protokoll SMTP(S) zum Einsatz (Nachricht 4).

Die E-Mail liegt nun beim empfangenden MTA. Dieser wertet den **Benutzeranteil** der E-Mail-Adresse aus. Der Benutzer im abgebildeten Beispiel ist „b“. Anschließend wird die E-Mail in das Postfach des Benutzers sortiert. Dies übernimmt eine Komponente des Mail-Servers, die Mail Delivery Agent (MDA) genannt wird. Manchmal kommt als MDA eine separate Software zum Einsatz.

Im letzten Schritt ruft der E-Mail-Client (MUA) des Empfängers sein Postfach ab. Dabei kommt meistens das **Internet Message Access Protocol Secure (IMAPS)** zum Einsatz (Nachricht 5). Alternativ kann auch das **Post Office Protocol 3 Secure (POP3S)** verwendet werden. Beim Einsatz von Microsoft Exchange in Verbindung mit Microsoft Outlook wird oftmals das **Messaging Application Programming Interface (MAPI)** benutzt.

Beim E-Mail-System eingesetzte Protokolle	
Abkürzung	**Erläuterung**
SMTP	Das Simple Mail Transfer Protocol wird zum Versenden von E-Mails verwendet. Es arbeitet auf TCP-Port 25. SMTP arbeitet unverschlüsselt.
SMTPS	Das Simple Mail Transfer Protocol Secure wird zum Versenden von E-Mails verwendet. Es arbeitet auf TCP-Port 465. Bei SMTPS wird die Kommunikation mit TLS (Transport Layer Security) verschlüsselt.
IMAP	Das Internet Message Access Protocol dient zum Abrufen von E-Mails. Die E-Mails verbleiben dabei auf dem Server. Dadurch kann auf ein Postfach von mehreren Endgeräten zugriffen werden. Auch ein Zugriff über ein Webinterface ist möglich. IMAP arbeitet standardmäßig unverschlüsselt auf Port 143.
IMAPS	Das Internet Message Access Protocol Secure ist die verschlüsselte Variante von IMAP. Sie verwendet TLS für die Verschlüsselung und arbeitet auf Port 993.
POP3	Das Post Office Protocol 3 dient ebenfalls dem Abrufen von E-Mails. Allerdings werden die E-Mails auf den Client übertragen und vom Server gelöscht. Eine Synchronisation mehrerer Clients ist so nicht möglich. POP3 arbeitet unverschlüsselt auf Port 110.
POP3S	Das Post Office Protocol 3 über SSL/TLS oder STARTTLS ist die verschlüsselte Variante des POP3. Als Port wird 995 eingesetzt.
MAPI	Das Messaging Application Programming Interface ist eine von Microsoft entwickelte, proprietäre Schnittstelle für den Zugriff auf E-Mails.

Authentifizierung im E-Mail-System

Möchte ein Mail User Agent (MUA) eine E-Mail versenden, muss er sich zunächst gegenüber dem E-Mail-Server authentifizieren. Damit wird verhindert, dass ein unbefugter MUA E-Mails über den E-Mail-Server, z. B. als Spam, verschicken kann.

! Eine einfache Implementierung des SMTP sieht keine Authentifizierung der E-Mail-Server untereinander vor. Diese Authentifizierung muss durch andere Protokolle umgesetzt werden.

Wenn ein E-Mail-Server (MTA) eine E-Mail von einem anderen E-Mail-Server empfängt, ist es aber sinnvoll, dass die Identität des anderen, sendenden Servers überprüft wird. Dies ist nötig, da der wirkliche Absender einer E-Mail gefälscht werden kann. Für die Überprüfung des Absenders haben sich drei Verfahren durchgesetzt.

Beim E-Mail-System eingesetzte Authentifizierungsverfahren	
Verfahren	**Arbeitsweise**
DKIM	Beim Verfahren DomainKeys Identified Mail soll sichergestellt werden, dass der Inhalt einer E-Mail auf dem Weg zwischen Sender und Empfänger nicht verändert wurde. DKIM verwendet dazu das Verfahren der digitalen Signatur. Über die komplette E-Mail (Header und Body) wird ein Hash-Wert H gebildet und anschließend mit dem privaten Schlüssel des Absenders verschlüsselt. E-Mail und verschlüsselter Hash-Wert werden an den Empfänger gesendet. Dieser entschlüsselt den empfangenen Hash-Wert mit dem zum Absender passenden öffentlichen Schlüssel. Er bildet einen eigenen Hash-Wert H' über die komplette E-Mail. Abschließend vergleicht er die beiden Hash-Werte H und H'. Wenn diese identisch sind, ist die Integrität gewährleistet.
SPF	Beim Sender Policy Framework wird der Domain-Teil der Absender-E-Mail-Adresse einer empfangenen E-Mail geprüft (z. B. lautet bei der E-Mail-Adresse „test@example.com" der Domain-Teil „example.com"). Der empfangende E-Mail-Server befragt das DNS-System, ob es einen SPF-Eintrag zu der Domain gibt. In diesem SPF-Eintrag ist eine IP-Adresse hinterlegt, von der E-Mails mit entsprechendem Domain-Teil kommen dürfen. Diese IP-Adresse wird mit der Absender-IP-Adresse verglichen. Nur wenn die beiden IP-Adressen übereinstimmen, gilt der Absender als verifiziert.
DMARC	Die Spezifikation Domain-based Message Authentication, Reporting and Conformance kann zusätzlich zu DKIM und SPF festlegen, was mit E-Mails passiert, bei denen SPF und/oder DKIM fehlschlägt. Dazu muss es im DNS-System eine DMARC-Richtlinie geben, die anzeigt, wie mit der E-Mail zu verfahren ist.

SMTP-Relay

Unter einem SMTP-Relay versteht man zunächst einen E-Mail-Server, der E-Mails entgegennimmt und an das Ziel weiterleitet. Problematisch ist es, wenn es sich um ein **offenes SMTP-Relay** handelt, das E-Mails ohne Authentifizierung entgegennimmt und weiterleitet, für die es nicht der empfangende oder absendende MTA ist. Dann kann ein SMTP-Relay für das Versenden von Spam verwendet werden. E-Mail-Server, die als offenes SMTP-Relay konfiguriert sind, werden oftmals auf den Blocklists der anderen Provider geführt.

Aufbau einer E-Mail

Der Empfang von Spam-E-Mails und Phishing-E-Mails, gerade in Bezug auf Ransomware, ist ein großes Problem. Für die Definition erfolgreicher und genauer Filterregeln ist es sinnvoll, den Aufbau einer E-Mail zu verstehen. Dieser ist im RFC 5322 beschrieben. Bevor der eigentliche E-Mail-Header (Kopfzeile; s. nachfolgende Tabellen) bearbeitet wird, werden die SMTP-Daten ausgetauscht.

SMTP-Befehle	
Befehl	**Bedeutung**
HELO	Dienst zum Aufbau der SMTP-Verbindung. Als Parameter wird normalerweise der Domain-Name (FQDN) oder alternativ die IP-Adresse des Clients angegeben.
MAIL FROM	Hiermit wird die E-Mail-Adresse des Senders angegeben. Wenn die E-Mail-Adresse akzeptiert wird, beantwortet der Empfänger die Nachricht mit einem 250-OK-Code. Mit diesem Befehl wird dem Empfänger signalisiert, dass ein neuer Datenaustausch beginnt und alle Speicher zurückgesetzt werden müssen.
RCPT TO	Mit RCPT TO wird die E-Mail-Adresse des Empfängers angegeben. Wenn der Befehl mehrfach angegeben wird, kann eine E-Mail an mehrere Empfänger zugestellt werden. Wenn die Empfänger-E-Mail-Adresse akzeptiert wird, beantwortet der Empfänger der RCPT-TO-Nachricht, ein E-Mail-Server, die Nachricht mit einem 250-OK-Code.

SMTP-Befehle	
Befehl	**Bedeutung**
DATA	Der Befehl DATA signalisiert, dass der Transport der Daten beginnt. Der Empfänger, ein MTA, beantwortet den DATA-Befehl mit „354 Send message content;", anschließend können die Daten gesendet werden. Das Ende des Transfers der Nutzdaten wird durch einen „." in einer einzelnen Zeile angegeben. Der empfangende E-Mail-Server bestätigt den Empfang mit einem 250-OK-Code.
RSET	Der RSET-Befehl (für Reset) setzt eine geöffnete noch nicht komplett durchgeführte SMTP-Verbindung zurück.
VRFY	Durch den VRFY-Befehl (Verify) kann beim E-Mail-Server nachgefragt werden, ob das E-Mail-Postfach des Empfängers existiert.
NOOP	Für die Überprüfung, ob die SMTP-Verbindung zum E-Mail-Server noch aufgebaut wird, kann der NOOP-Befehl (No operation) verwendet werden. Der angesprochene E-Mail-Server antwortet mit einem 250-OK-Code.
QUIT	Signalisiert, dass die Verbindung beendet werden soll. Wenn der E-Mail-Server die Verbindung schließen kann, sendet er einen 221-Bye-Code als Bestätigung.
EHLO	Der Extended-Hello-Befehl (EHLO-Befehl) wird statt des HELO-Befehls zum Verbindungsaufbau eingesetzt, um beim Austausch erweiterte SMTP-Befehle zu verwenden.
AUTH	Der AUTH-Befehl (Authentication) wird genutzt, um sich beim Server zu authentifizieren. Es gibt verschiedene Authentifizierungsverfahren. Wird das Verfahren PLAIN verwendet, werden Benutzername und Passwort Base64-codiert an den Server übertragen.
STARTTLS	Mit dem Befehl STARTTLS (Start Transport Layer Security) wird eine TLS-Verbindung gestartet und der komplette E-Mail-Verkehr zwischen den beiden beteiligten Parteien verschlüsselt. Andere E-Mail-Server auf dem Weg zum Ziel können die E-Mail aber noch immer unverschlüsselt weitersenden.

Die SMTP-Daten werden nach der Zustellung einer E-Mail – auch die Empfängeradresse aus RCPT TO – verworfen. Eine E-Mail lässt sich in zwei Abschnitte einteilen: Kopf (engl. Header) und Körper (engl. Body) der E-Mail. Für die Zustellung einer E-Mail sind die Informationen des E-Mail-Headers teilweise nicht relevant. Aus diesem Grund lassen sich diese Informationen auch leicht verfälschen.

Felder in E-Mail-Header (Auszug)	
Feld	**Inhalt**
Return-Path	Dieses optionale Feld steht zu Beginn des E-Mail-Headers. Es enthält die Absender-E-Mail-Adresse aus dem „Envelope-Feld" des SMTP. Dieses Feld kann allerdings beliebig beschrieben werden.
Authentication-Results	Dieses Feld enthält das Authentifizierungsverfahren zwischen den E-Mail-Servern.
Received	Eine E-Mail kann mehrere Received-Felder enthalten. Ein Received-Feld wird vor dem Weiterleiten einer E-Mail vorangestellt. Der eigene E-Mail-Server hat das oberste Received-Feld angefügt. Jedes Received-Feld gibt an, wer die E-Mail von wem empfangen hat.
Message-ID	Dieses Feld ist eine eindeutige Kennung einer E-Mail. Damit die Kennung einmalig ist, wird oftmals ein Zeitstempel und eine Prozess-ID oder eine codierte Benutzerkennung verwendet. Dieser ersetzte Teil wird durch ein „@" mit nachfolgendem Domänen-Namen ergänzt.
Date	Dieses Feld beinhaltet das Absendedatum und wird vom absendenden Client eingetragen.

Felder in E-Mail-Header (Auszug)	
Feld	**Inhalt**
MIME-Type	Mit diesem Feld wird angegeben, um welche Art von Daten es sich handelt. Eine Übersicht der verschiedenen MIME-Types kann bei der Internet Assigned Numbers Authority eingesehen werden.
User-Agent	Dieses optionale Feld gibt an, mit welchem E-Mail-Programm die E-Mail versendet wurde.
Envelope-To	Das Feld beinhaltet die Empfänger-E-Mail-Adresse. Diese wird normalerweise aus dem SMTP-Datenverkehr (RCPT-TO) übernommen, kann aber gefälscht werden.
„X"-Felder	Die Felder, die mit einem X beginnen, sind optional. Ein mögliches Feld ist beispielsweise „X-UI-Filterresults". Dieses gibt an, ob eine E-Mail als Spam klassifiziert ist.

Beispiel: Aufbau einer E-Mail

```
01 Return-Path: <absender@beispiel.de>
02 Authentication-Results: beispiel.de; dkim=none
03 Received: from mx.beispiel.de ([212.227.126.133]) by mx-mtu.example.com
04 (mxbeispiel [12.227.17.5]) with ESMTPS (Nemesis) id 1MdvW2-1oM2Xx20ZS-00az9Z
05 for <absender@beispiel.de>; Wed, 04 May 2022 09:15:35 +0200
06 Received: from [192.168.100.100] ([85.17.204.25]) by mxrelay.beispiel.de
07 (mxeu [22.27.15.17]) with ESMTPSA (Nemesis) id
08 1 mmlGg-1oDCrJ0XTS-00jtoI for <absender@beispiel.de>; Wed, 04 May 2022 09:15:35
09 +0200
10 Message-ID: <deeu-4871-0815-4711-06757c9bbc70@example.com>
11 Date: Wed, 4 May 2022 09:15:34 +0200
12 MIME-Version: 1.0
13 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101
14 Thunderbird/91.8.0
15 Content-Language: de-DE
16 To: Name des Empfängers <empfaenger@example.com>
17 Cc: Name des CC-Empfängers <cc-empfaenger@example.com>
18 From: "Name des Abenders" <absender@beispiel.de>
19 Subject: Betreff der E-Mail
20 Content-Type: text/plain; charset=UTF-8; format=flowed
21 Content-Transfer-Encoding: 7bit
22 Envelope-To: <empfaenger@example.com>
23
24 Body-Text
```

Erklärung: Eine E-Mail von dem Absender „absender@beispiel.de" an den Empfänger „empfaenger@example.com" wurde empfangen. Das Feld „Return-Path" (Zeile 01) zeigt den Absender „absender@beispiel.de" an. Der Absender wird aus dem SMTP-Datenverkehr übernommen, kann aber gefälscht werden. Das Feld „Authentication-Results" (Zeile 02) zeigt an, dass zwischen den E-Mail-Servern von „beispiel.de" und „example.com" keine Authentifizierung der Server untereinander stattgefunden hat. In Zeile 03 wird durch das „Received" angegeben, dass der letzte empfangende E-Mail-Server „mx-mtu.example.com" ist und die E-Mail vom E-Mail-Server „mx.beispiel.de" abgesendet wurde. Der Namensteil „mx" als Kennzeichnung für „mail exchange" findet sich oftmals bei E-Mail-Servern zur besseren Zuordnung. Das „Received" in Zeile 06 wurde vom E-Mail-Server „mxrelay.beispiel.de" angelegt. Es zeigt an, dass die E-Mail von diesem E-Mail-Server empfangen und von einem Client mit der privaten IP-Adresse 192.168.100.100 gesendet wurde. In Zeile 10 wird eine Message-ID angegeben, gefolgt vom Absendedatum in Zeile 11 und der verwendeten MIME-Version in Zeile 12. Das Feld „User-Agent" (Zeile 14) zeigt an, dass Thunderbird unter Linux als Mail User Agent (MUA) zum Einsatz kam. Die Daten in den Feldern „To" (Zeile 16), „Cc" (Zeile 17) und „From" (Zeile 18) lassen sich beliebig manipulieren. Das wird besonders beim Phishing ausgenutzt, um falsche Daten vorzutäuschen. Das Feld „Subject" in Zeile 19 enthält den Betreff der E-Mail. Die Felder in den Zeilen 20 und 21 enthalten Meta-Angaben zum Inhalt der E-Mail. Im

Beispiel enthält die empfangene E-Mail im Kodierungsschema UTF-8 kodierten Text. Das Feld „Envelope-To:“ (Zeile 22) enthält nochmals die E-Mail-Adresse des Empfängers. Normalerweise wird der Inhalt dieses Feldes vor dem stattfindenden SMTP-Datenaustausch aus dem Feld RCPT-TO kopiert.

(7) Webserver

Webserver

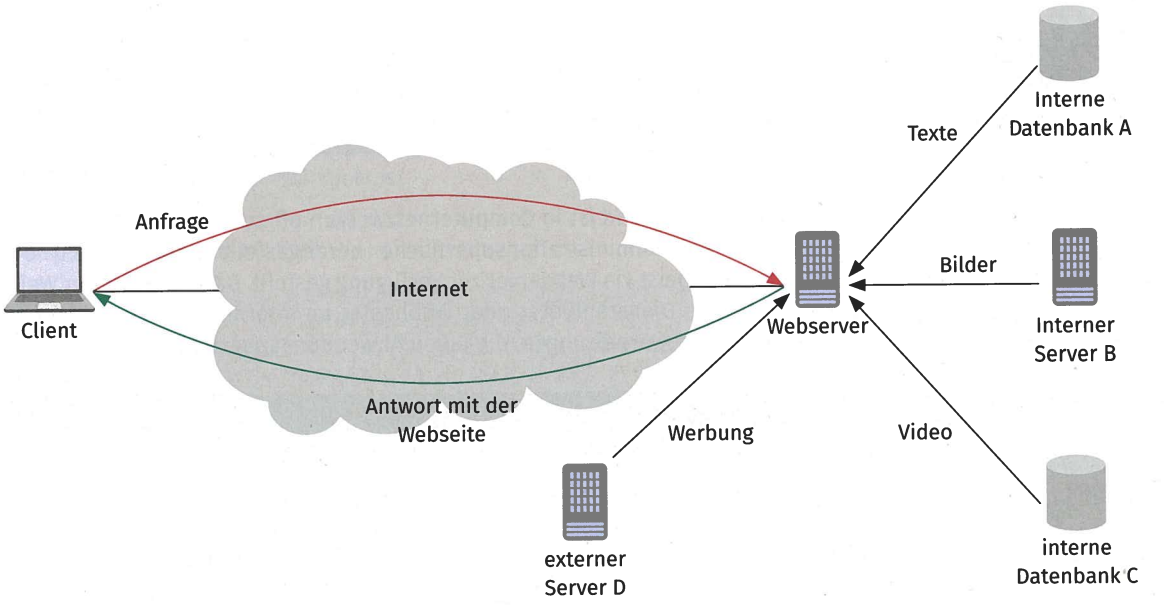
Ein Webserver ist die umgangssprachliche Bezeichnung für einen Server, der einen HTTPS-Dienst (oder HTTP-Dienst) zur Verfügung stellt. Mit diesem Dienst lassen sich statische oder dynamische Webseiten an einen Webbrowser übertragen.

Ein Webserver gehört zu den Standardservern und ist in Computernetzwerken oft zu finden. Über ihn kann beispielsweise auf Koppelelementen die Administrationsoberfläche bereitgestellt werden, auch die firmeninterne Kooperationssoftware wird meist via Webserver zur Verfügung gestellt. Damit gehört der Webserver zum Softwareprogramm der großen Dienstanbieter oder Webhoster im Internet. Dementsprechend gibt es eine große Anzahl verschiedener Softwarelösungen, die jeden Anwendungszweck abdecken.

Auswahl bekannter Webserver im Vergleich

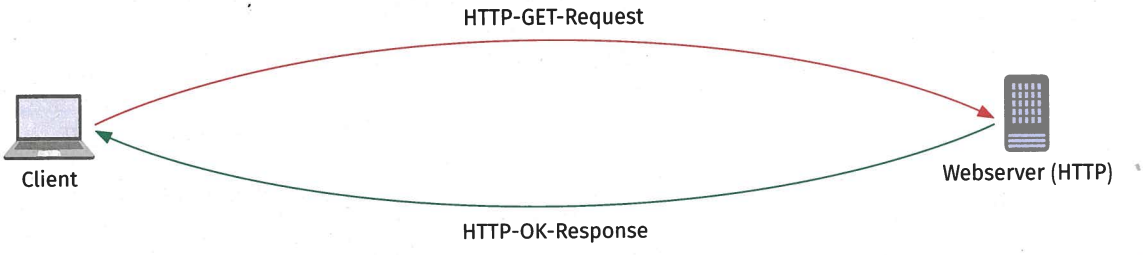
Webserver	Beschreibung	Vorteile	Nachteile
Nginx	einer der verbreitetsten Webserver im Internet; kann auf Unix- oder Microsoft Betriebssystemen eingesetzt werden	modular aufgebaut; gut erweiterbar; verbraucht weniger Ressourcen, z. B. im Vergleich zum Apache HTTP Server	Konfiguration auf Verzeichnisebene wird nicht unterstützt
Microsoft Internet Information Services	in Microsoft Server integrierte modulare Kommunikationsplattform, die u. a. auch HTTPS unterstützt	in vielen Windows-Versionen ein integrativer Bestandteil des Betriebssystems; unterstützt verschiedene Protokolle; durch die hohe Integration einfach zu installieren bzw. zu aktivieren; bietet viele Funktionen, besonders die Integration der Technik „ASP“ (active server pages) ist sehr gut	als Bestandteil eines Microsoft-Betriebssystems benötigt die Verwendung Lizenzen, z. B. die Server-Lizenz und bei öffentlicher Nutzung eine „External Connector Lizenz“
Apache HTTP Server	bekannter Webserver, der schon 1995 eingeführt wurde; stetig weiterentwickelt; weit verbreitet; steht als quelloffene Software unter vielen Betriebssystemen zur Verfügung	kann durch Module erweitert werden; dadurch viele Einsatzszenarien; bietet die Möglichkeit auf Verzeichnisebene (durch die .htaccess-Datei) die Konfiguration anzupassen, um z. B. Zugriffsberechtigungen umzusetzen	benötigt bei vielen Anfragen auch sehr viele Ressourcen, da für jede Anfrage ein eigener Prozess gestartet wird; durch die vielen Möglichkeiten ist Konfiguration, je nach Anwendungsfall, komplex
Lighttpd	performanter Webserver; arbeitet auf Unix-Betriebssystemen; kann kostenfrei verwendet werden	kann durch Module erweitert werden; installierbar auf nahezu allen Linux-Repositories; verbraucht sehr wenige Ressourcen; arbeitet performant und startet nicht für jede Anfrage einen eigenen Prozess	an Lighttpd arbeiten weniger aktive Entwickler

Ein Webserver stellt Webseiten zum Abruf bereit. Diese können als statische Seiten vorliegen, deren Inhalt sich nicht verändert. Sie werden z. B. durch die Beschreibungssprache HTML (Hypertext Markup Language) einmalig erstellt. Viele Seiten werden allerdings dynamisch erzeugt. Bei einem Seitenabruf werden dabei Daten aus Datenbanken abgerufen und die Webseite zusammengestellt. Die Daten für eine Webseite können von unterschiedlichen Systemen geliefert werden. Datenbank A liefert beispielsweise die Texte, Server B die Bilder, Datenbank C die Videos und Server D die Werbung für die Webseite. Aus den Daten wird die Webseite erzeugt und an den Browser ausgeliefert.



Zusammenbau einer dynamischen Webseite

Für das Abrufen einer Webseite durch den Browser wird das Protokoll HTTPS (Hypertext Transfer Protocol Secure) verwendet. Das unsichere HTTP (Hypertext Transfer Protocol) wird nur noch selten eingesetzt. Der Nachrichtenaustausch bei HTTPS ist verschlüsselt und kann nicht ohne Weiteres als Klartext in einer Netzwerkanalysesoftware wie „Wireshark" dargestellt werden, sodass im Verlauf der unverschlüsselte, identische HTTP-Ablauf besprochen wird.



Kommunikation zwischen Client mit Webbrowser und Webserver

Der Webbrowser, in diesem Fall der Client, sendet dem Webserver einen HTTP-GET-Request. In dieser Anfrage werden verschiedene Parameter an den Server weitergegeben, z. B. mit welchem Webbrowser die Webseite aufgerufen oder welche Sprache bevorzugt wird.

```
Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
    Host: www.example.com\r\n
  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
  Accept-Language: de,en-US;q=0.7,en;q=0.3\r\n
  Accept-Encoding: gzip, deflate\r\n
  Connection: keep-alive\r\n
  Upgrade-Insecure-Requests: 1\r\n
  If-Modified-Since: Thu, 17 Oct 2019 07:18:26 GMT\r\n
  If-None-Match: "3147526947"\r\n
  Cache-Control: max-age=0\r\n
```

Wireshark-Mitschnitt eines HTTP-GET-Request

Durch die vom Client an den Server gesendeten Parameter ist es dem Server möglich, dem Client eine bestimmte Version einer Webseite auszuliefern, z. B. die Webseite in deutscher Sprache (siehe das Feld „Accept-Language“) oder eine an den Browser (siehe das Feld „User-Agent“) angepasste Version. Der Server antwortet im Normalfall mit der Statusmeldung „200 OK“ und überträgt mit dieser Nachricht die Webseite.

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Content-Encoding: gzip\r\n
  Age: 569739\r\n
  Cache-Control: max-age=604800\r\n
  Content-Type: text/html; charset=UTF-8\r\n
  Date: Sun, 08 May 2022 17:46:53 GMT\r\n
  Etag: "3147526947+gzip"\r\n
  Expires: Sun, 15 May 2022 17:46:53 GMT\r\n
  Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT\r\n
Line-based text data: text/html (46 lines)
  <!doctype html>\n
  <html>\n
  <head>\n
      <title>Example Domain</title>\n
```

Wireshark-Mitschnitt der HTTP-Statusmeldung „200 OK“

Die Statusmeldungen bei HTTPS sind standardisiert und besitzen festgelegte Codes.

Gängige Statuscodes (kleine Auswahl)	
Statusmeldung	**Bedeutung**
200 OK	Die Anfrage war erfolgreich und das Ergebnis wird übermittelt.
301 Moved Permanently	Die Adresse ist nicht mehr gültig, aber die angeforderte Ressource steht unter einer anderen Adresse zur Verfügung.
403 Forbidden	Der Client hat nicht die nötige Berechtigung, die Ressource aufzurufen.
404 Not Found	Die Ressource wurde nicht gefunden.

Damit der Datenverkehr zwischen Client und Server abgesichert ist, wird dieser authentifiziert und verschlüsselt. Dazu wird das Protokoll Transport Layer Security (TLS) verwendet.

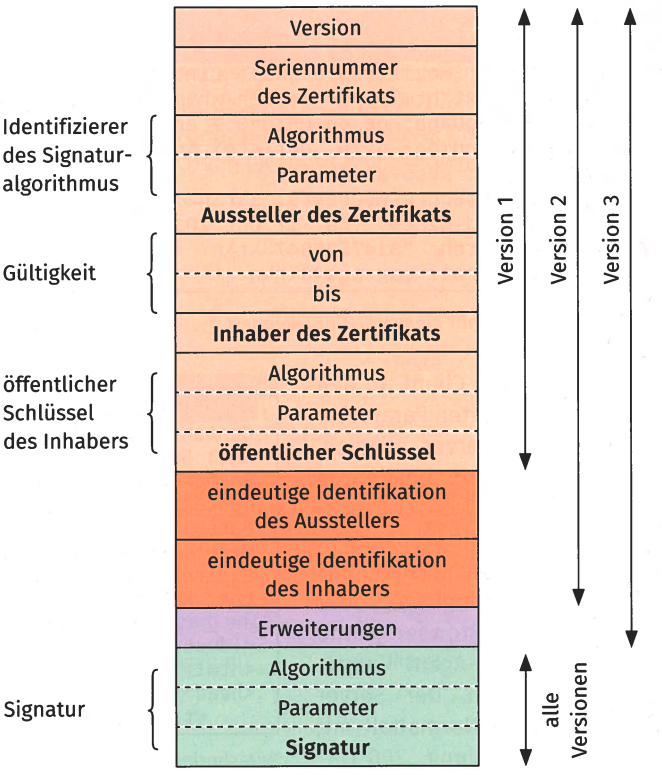
Public Key Infrastructure (PKI) mit Zertifikaten

Zu Beginn einer Verbindung zwischen dem Webbrowser (Client) und dem Webserver (Server) sendet der Webserver dem Client ein **Zertifikat**. Ein Zertifikat ist ein Nachweis. Erhält man z. B. für eine abgelegte Prüfung ein Zertifikat, so ist dies der Nachweis, dass man über bestimmte, in der Prüfung abgefragte Inhalte Kenntnis hat. Für ein Zertifikat ist es wichtig, dass man der Person oder der Institution vertraut, die das Zertifikat ausgestellt hat. Ein von einem WG-Mitbewohner ausgestelltes Zertifikat über IT-Sicherheit ist sicherlich nicht sehr nützlich, außer der Mitbewohner ist ein ausgewiesener IT-Sicherheitsfachmann. Es ist also wichtig, dass man auch dem Aussteller bzw. der ausstellenden Institution vertraut. In der Kommunikation zwischen Webbrowser und Webserver ist das genauso.

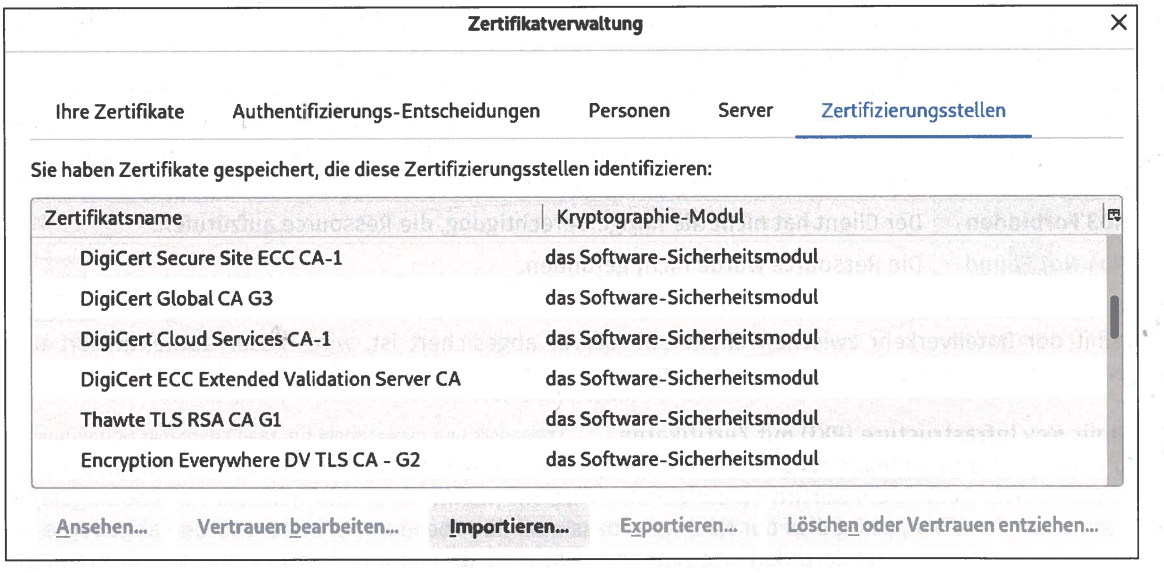
Ein Zertifikat für die sichere Kommunikation folgt dem Standard X.509. Für die Verschlüsselung und Authentifizierung sind die folgenden fett gedruckten Parameter entscheidend.

Um ein öffentlich anerkanntes Zertifikat, z. B. für einen Webserver, zu beantragen, wird ein **Certificate Signing Request (CSR)** erstellt. Dazu erzeugt der Benutzer im ersten Schritt ein **Schlüsselpaar** bestehend aus einem **privaten** und einem **öffentlichen** Schlüssel. Der private Schlüssel ist geheim und darf niemals weitergegeben werden. Der öffentliche Schlüssel hingegen darf jedem bekanntgegeben werden. Anschließend wird der eigentliche CSR erzeugt. Dieser beinhaltet z. B. den öffentlichen Schlüssel, den Domain-Namen und die E-Mail-Adresse.

Im zweiten Schritt wird der mit dem privaten Schlüssel des Antragstellers signiert. Im dritten Schritt wird der signierte CSR an die Zertifizierungsstelle (Certificate Authority – CA) gesendet. Diese prüft die Angaben des CSR und kontrolliert, mit dem angefügten öffentlichen Schlüssel die Signatur. Wenn alle Angaben rechtmäßig sind, fügt die Zertifizierungsstelle weitere Informationen, wie z. B. die Seriennummer des Zertifikats, hinzu, signiert abschließend das Zertifikat und hängt die Signatur diesem an. Das Zertifikat kann nun verwendet werden.



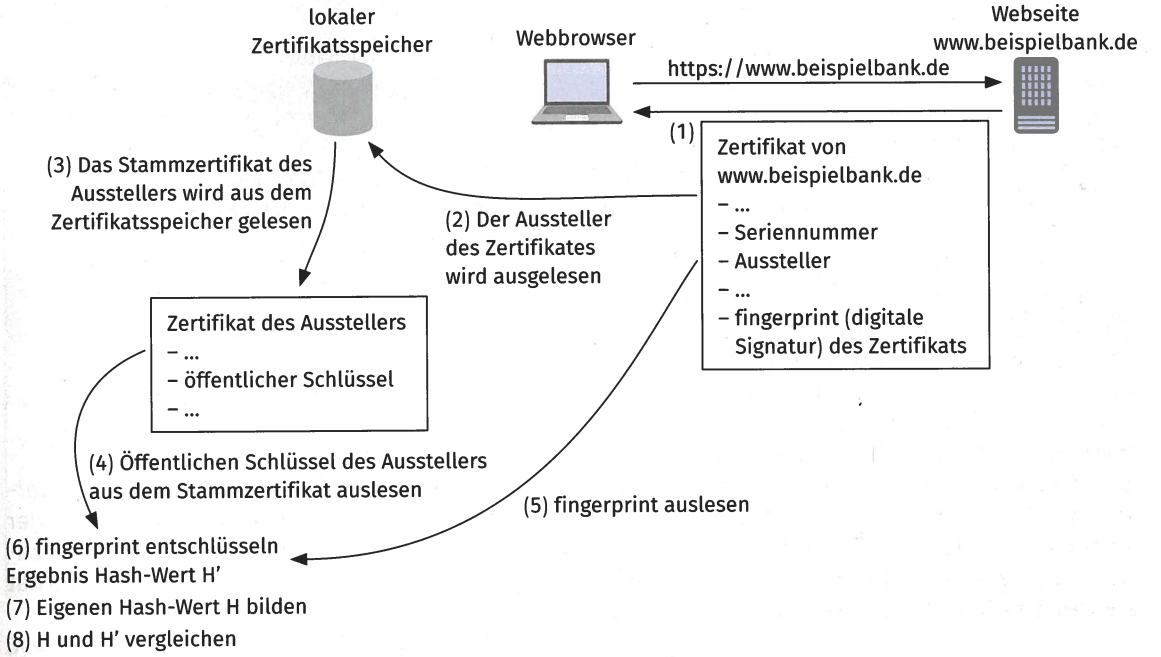
X.509 Zertifikatsformat für öffentliche Schlüssel



Beispiel für einen Zertifikatspeicher

Wenn der Webbrowser (Client) das Zertifikat erhält (vgl. (1) in der Abbildung auf der folgenden Seite), muss er das Zertifikat auf dessen Gültigkeit überprüfen. Dazu liest der Client aus dem Zertifikat den Aussteller (2) heraus. Aus dem Zertifikatsspeicher des Betriebssystems oder des Browsers liest er das passende Stammzertifikat (3).

Der Zertifikatsspeicher beinhaltet Zertifikate, denen der Hersteller des Betriebssystems oder des Browsers vertraut. Um das Vertrauen zu erwerben, muss eine Zertifizierungsstelle bestimmten Kriterien genügen. In Deutschland muss eine solche Stelle von der Bundesnetzagentur akkreditiert sein. Mittels Betriebssystem- oder Browserupdate können neue Zertifizierungsstellen in den Zertifikatsspeicher geladen oder nicht mehr vertrauenswürdige Stellen gelöscht werden.

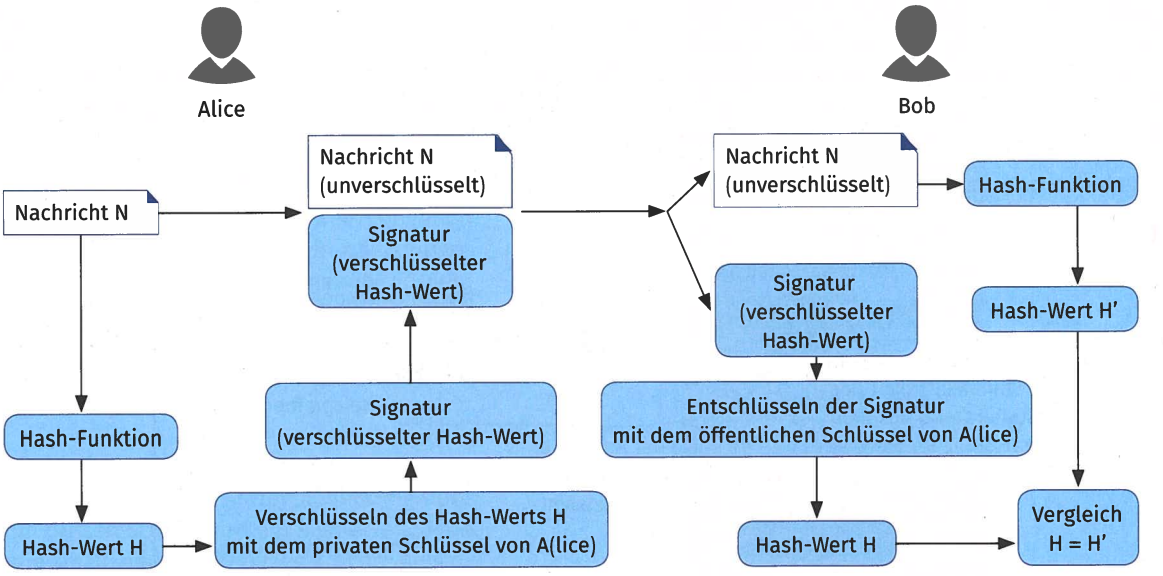


Validierung eines Zertifikats im Browser

Ein Teil des Zertifikats aus dem Zertifikatsspeicher ist der öffentliche Schlüssel der Zertifizierungsstelle. Hat der Webbrowser das Zertifikat der Webseite und den öffentlichen Schlüssel (4) muss er im nächsten Schritt analysieren, ob das Zertifikat der Webseite auch von der angegebenen Zertifizierungsstelle unterschrieben ist. Dazu überprüft der Webbrowser die digitale Signatur des empfangenen Zertifikats der Webseite. Zunächst entschlüsselt er den **Fingerprint** des erhaltenen Zertifikats mit dem öffentlichen Schlüssel der Zertifizierungsstelle und erhält den Hash-Wert H' (5) (6). Anschließend bildet er einen eigenen Hash-Wert H über das Zertifikat (7). Zuletzt vergleicht er den selbst erstellten Hash-Wert H mit dem entschlüsselten Hash-Wert H' (8). Wenn beide Hash-Werte gleich sind, ist sichergestellt, dass das Zertifikat der Webseite von der Zertifizierungsstelle unterschrieben wurde. Vertraut der Browser der Zertifizierungsstelle, kann er dem Zertifikat der Webseite vertrauen.

Digitale Signatur

Mit der **digitalen Signatur** wird das Schutzziel der Authentizität und der Integrität sichergestellt. Der Sender A möchte nachweisen, dass Empfänger B sicher sein kann, dass die Nachricht N (oder die Daten) auch von ihm ist und während der Übertragung nicht verändert wurde. Dazu bildet er mit einer Hash-Funktion einen Hash-Wert H über die zu sendende Nachricht. Anschließend verschlüsselt er den Hash-Wert mit dem eigenen privaten Schlüssel. Dieser Wert wird Signatur genannt. Den verschlüsselten Hash-Wert (also die Signatur) und die Nachricht (unverschlüsselt!) sendet A an den Empfänger B. Der Empfänger B entschlüsselt den verschlüsselten Hash-Wert mit dem öffentlichen Schlüssel des Senders A und erhält den unverschlüsselten Hash-Wert H. Nun bildet der Empfänger B selbst einen Hash-Wert H' über die Nachricht. Abschließend vergleicht er H und H'. Wenn beide Werte gleich sind, kann der Empfänger sicher sein, dass die Nachricht unverändert von Absender A kommt, da nur dieser im Besitz des privaten Schlüssels ist.



Ablauf bei Prüfung der digitalen Signatur

Ablauf der Verbindung mit dem Protokoll Transport Layer Security (TLS)
Nachdem das Zertifikat überprüft wurde, kann die eigentliche verschlüsselte Verbindung aufgebaut werden. Für die Verschlüsselung wird das Transport-Layer-Security-Protokoll verwendet. TLS arbeitet auf der fünften OSI-Schicht, der Sitzungsschicht. TLS wird zurzeit in der Version 1.3 eingesetzt und ist der Nachfolger des Secure Socket Layers (SSL). Durch den Einsatz von TLS soll Vertraulichkeit und Authentizität gewährleistet werden (siehe auch Jahrgangsband 1, Lernfeld 4, Kapitel 4.1.3).

TLS besteht aus verschiedenen Teilprotokollen. Im ersten Schritt kommt das **TLS Handshake Protocol** zum Einsatz. Dabei authentifiziert sich der Server gegenüber dem Client mit einem Zertifikat. Anschließend werden die Verschlüsselungsparameter ausgehandelt. Der Client legt einen Sitzungsschlüssel fest und sendet diesen verschlüsselt an den Server. Dazu verwendet er den öffentlichen Schlüssel des Servers aus dessen Zertifikat. Im zweiten Schritt wird das **TLS Change Cipher Spec Protocol** eingesetzt. Es signalisiert, dass ab nun die Daten der Sitzung verschlüsselt werden. Anschließend werden folgende Schritte abgearbeitet: Das **TLS Record Protocol** ist für die eigentliche Verschlüsselung zuständig und ist der Kern des TLS. Durch das **TLS Application Data Protocol** werden die Daten an den Record Layer weitergeleitet. Mit dem **TLS Alert Protocol** werden Warnungen und Fehler des TLS übermittelt. Fehler führen zu einem Verbindungsabbruch.

(8) Videokonferenzsysteme und WebRTC

Videokonferenzsysteme werden seit Anfang der 2000er-Jahre eingesetzt. Anfangs waren diese Systeme sowohl in der Anschaffung als auch im Betrieb verhältnismäßig teuer. Die notwendigen Übertragungsraten waren nicht flächendeckend verfügbar und darüber hinaus teuer.

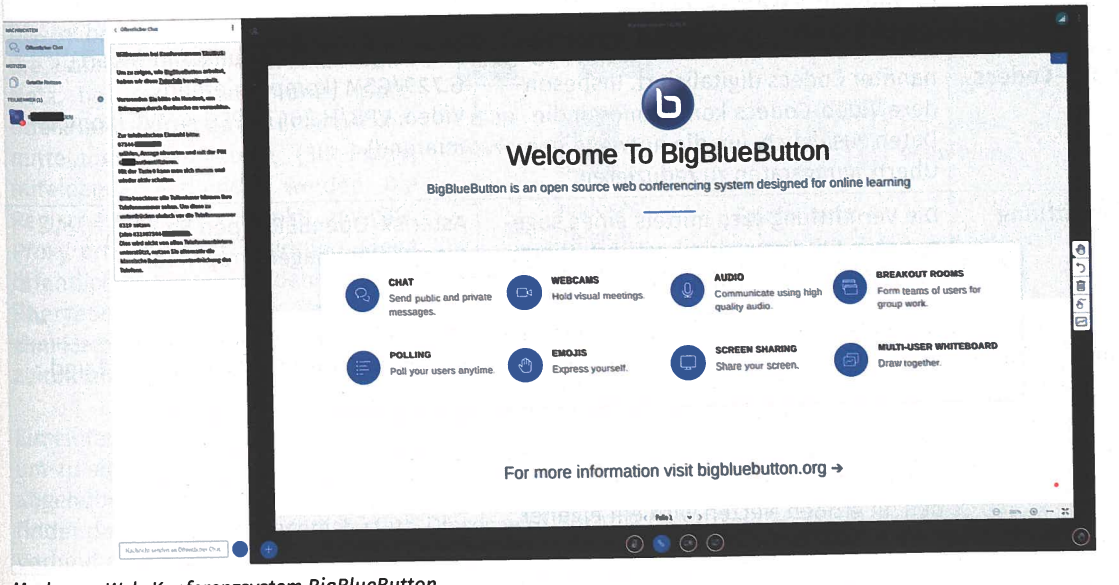
Zu Beginn der 2020er-Jahre haben die Videokonferenzsysteme eine sehr große Verbreitung erfahren. Sie werden neben dem betrieblichen Einsatz unter anderem im Bereich Home-Schooling und für virtuelle Vereinssitzungen eingesetzt.

Videokonferenzsystem

Ein Videokonferenzsystem wird zur zeitgleichen Übertragung von Audio- und Videoinformationen der Teilnehmer genutzt. Auf der Eingabeseite kommen mindestens Mikrofon und Kamera zum Einsatz. Zur Ausgabe werden Lautsprecher und Bildschirme verwendet. Ein Videokonferenzsystem verfügt in der Regel nicht über die Möglichkeit des direkten Verbindungsaufbaus zu einer beliebigen Gegenstelle (Anruf nicht möglich, keine Signalisierung). Bei Videokonferenzen sind meist mehr als zwei Teilnehmer beteiligt.

Komponenten in einem Videokonferenzsystem		
Komponente	**Erklärung**	**Beispiel**
Videoaufnahme	HD-Kamera zur Aufzeichnung der teilnehmenden Personen	USB-Webcam
Display	zur Wiedergabe der Videodaten	PC-Monitor/Tablet
Mikrofon	zur Aufzeichnung der Gespräche	Headset, Freisprechmikrofon in Webcam
Lautsprecher/ Kopfhörer	zur Wiedergabe der Gespräche	Headset/Lautsprecher
Videokonferenz-Client	Software zur Steuerung der Videokonferenz	Web-Anwendung wie Jitsi/BBB (Open Source), Zoom/Teams/Discord (proprietär)
Videokonferenz-Server	zentraler Server, über den alle Teilnehmer verbunden sind	Jitsi/BBB (Open Source), Zoom/Teams/Discord (proprietär)

Systeme wie Jitsi und BigBlueButton (BBB) gehören zu den sogenannten WebRTC-Anwendungen (Web Real-Time Communication, Web-Echtzeitkommunikation). Sie verwenden Webbrowser als Clients und nutzen einen Backend-Server zur Verteilung der Audio-/Videostreams. Die Teilnehmer benötigen daher außer den ohnehin notwendigen technischen Voraussetzungen (Mikrofon, Webcam, Display) keine zusätzlich im Vorfeld installierte Software. Wie bei proprietären Lösungen können WebRTC-Anwendungen neben den reinen Audio-/Videostreams auch für Dateitransfers, Desktopfreigaben, Chats oder geteilte Notizen genutzt werden (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.1 und 4.5.2).



Modernes Web-Konferenzsystem BigBlueButton

(9) VoIP-Dienste (Voice over IP)

Im Gegensatz zu Videokonferenzsystemen, bei denen man sich vorab, z. B. per E-Mail, zu einem Gespräch verabreden muss, kann man bei Voice over IP (VoIP) den Gesprächspartner direkt anrufen. Hierzu wird ein Gerät oder eine Software benötigt, die kontinuierlichen Kontakt zum vermittelnden Server aufrechterhält.

Voice over IP (VoIP)

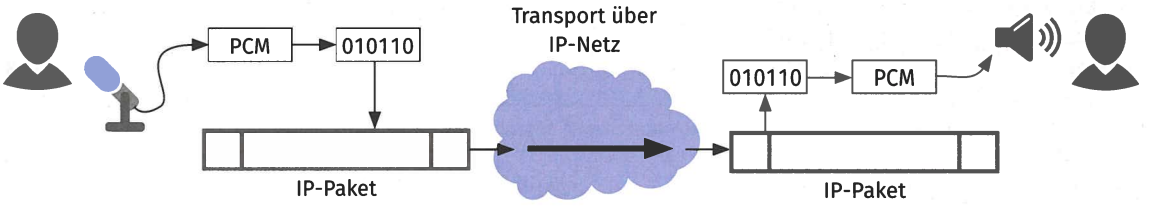
Voice over IP bezeichnet die Übertragung von Sprach- oder Videodaten in Echtzeit über eine paketvermittelnde (meist IP-)Infrastruktur (Echtzeitdatentransport). In den meisten Fällen kommunizieren genau zwei Teilnehmer miteinander. Werden Videodaten übertragen, wird auch von Bildtelefonie gesprochen. Darüber hinaus enthält VoIP zusätzlich die Möglichkeit, einen Gesprächspartner direkt anzurufen (Signalisierung).

Um eine Verbindung über ein VoIP-System aufbauen zu können, werden unterschiedliche Komponenten benötigt.

Komponenten in einem VoIP-System		
Komponente	**Erklärung**	**Beispiel**
Hardphone	VoIP-fähiges Telefon, welches als Tischgerät oder schnurlos ausgeführt sein kann	Avaya IP-Telefon J139, Cisco Unified IP Phone 7900 Series
Softphone	VoIP-fähige Software, die meist im Büro bzw. in Callcentern eingesetzt wird	PhonerLite, X-Lite
Signalisierung	Mittels der Signalisierung kann eine Verbindung zu einem Gesprächspartner aufgebaut werden.	SIP (Session Initiation Protocol), SDP (Session Description Protocol), MGCP (Media Gateway Control Protocol), H.323 (veraltet), Skinny/SCCP (Skinny Client Control Protocol) (proprietär)
Echtzeitdaten	Die Echtzeitdaten können Sprach- oder Videodaten enthalten. Diese werden auf einzelne IP-Pakete verteilt.	RTP (Real-Time Protocol)
Audio-/ Video-Codecs	Die Echtzeitdaten werden mittels sogenannter Codecs digitalisiert. Insbesondere Video-Codecs komprimieren die Daten zusätzlich, um die notwendigen Übertragungsraten zu reduzieren.	Audio: G.711/G.722 (unkomprimiert), G.729/GSM (komprimierend); Video: VP8/H.264/MPEG-4/AVC (komprimierend)
Vermittlung	Die Vermittlung wird mittels eines sogenannten SIP-Proxy durchgeführt. Hierzu muss dieser alle Teilnehmer und deren aktuelle IP-Adresse kennen.	Asterisk/OpenSER (Open Source), Cisco Callmanager (proprietär)
Anmeldung	Die Teilnehmer müssen sich wie bei einem E-Mail-Server zunächst authentifizieren, um an Gesprächen teilnehmen zu können. Diese Aufgabe kann in kleineren Netzen vom SIP-Proxy übernommen werden. In großen Netzen wird ein eigener sogenannter SIP-Registrar eingesetzt.	benutzername@mytestdomain.de und Passwort

Bereits im klassischen ISDN-Netz (Integrated Service Digital Network) wurde das analoge Audiosignal zunächst mittels einer Abtastung und Quantisierung digitalisiert. Die dabei entstandenen Binärwerte wurden mithilfe eines allgemeinen Audio-Codecs in übertragbare Daten gewandelt. Im klassischen ISDN-Netz wurden die einzelnen Daten Bit für Bit über eine zuvor aufgebaute Verbindung (Sprachkanal) zum Empfänger transportiert. Dabei wurden die Bits synchron vom Sender zum Empfänger gesendet. Es fand keine erneute Wegefindung (Routing) statt.

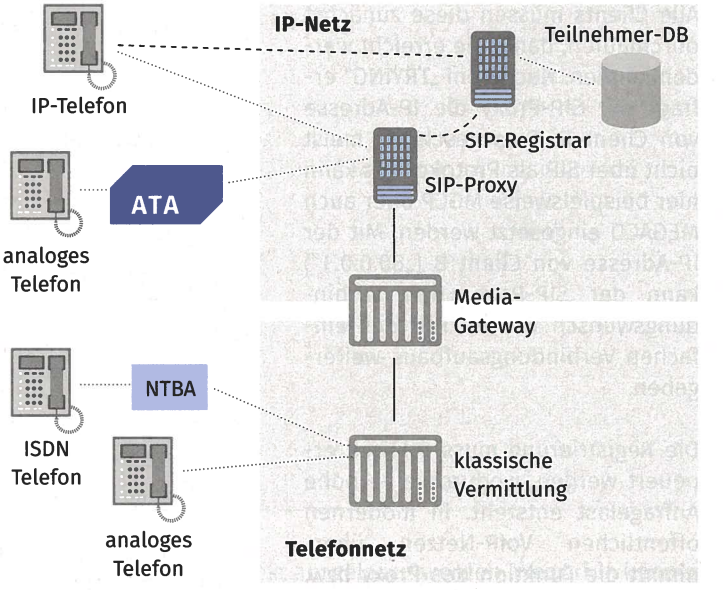
Bei VoIP werden nun die Digitalwerte nicht mehr Bit für Bit über einen festgelegten Kanal transportiert, sondern kontinuierlich auf IP-Pakete verteilt und diese jedes für sich zum Empfänger „geroutet". Da das IP-Protokoll keine Sicherung kennt (siehe TCP, Jahrgangsband 1, Lernfeld 3, Kapitel 3.3.4), kann es hierbei zu Zeitverzögerungen und sogar Verlusten kommen.



Prinzip Echtzeitdatentransport über IP-Verbindung

Hauptgrund für die Einführung von VoIP ist die vereinfachte Infrastruktur. Es wird nur noch eine IP-Infrastruktur benötigt. Die Infrastruktur im ISDN-Netz hingegen war auf die Übertragung von Sprachdaten optimiert und für hohe Übertragungsraten zu komplex.

Wenn anfangs die Nutzung eines Rechners mit Softphone und Headset noch als umständlich empfunden wurde, so gehört dies aktuell als selbstverständlich zum betrieblichen Umfeld. In vielen VoIP-Softphones sind weitere Features wie Presence Management (Signalisierung des Onlinestatus) und Versand von Kurznachrichten implementiert.

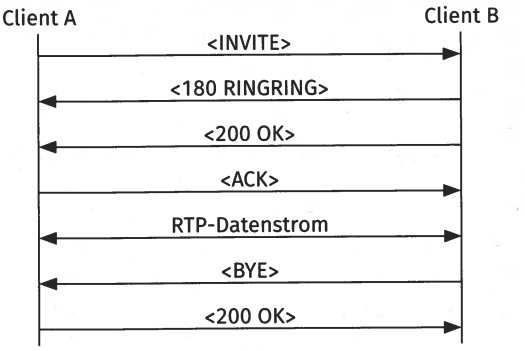


Prinzipieller Netzaufbau mit VoIP und klassischem Telefonnetz

Die nebenstehende Abbildung zeigt die einfachste Variante in einem VoIP-Netz, bei der die **IP-Phones (Soft- oder Hardphone)** direkt über ein vermittelndes Netzelement **(SIP-Proxy)** miteinander verbunden werden. Die Registrierung kann direkt am SIP-Proxy erfolgen oder in großen, meist öffentlichen Netzen an den Registrar übergeben werden. Die Verbindung zum klassischen Telefonnetz wird über ein Media-Gateway hergestellt, welches die Signalisierung und Echtzeitdaten auf die jeweiligen Protokolle umsetzt.

Ein einfacher Verbindungsaufbau beginnt mit einem „INVITE". Der Angerufene antwortet mit „180 RINGING", um zu signalisieren, dass das Telefon nun klingelt und auf einen Benutzer gewartet wird. Wird der Hörer abgehoben, so wird das mit „200 OK" signalisiert und der Anrufer quittiert dies mit „ACK". Im Folgenden findet das eigentliche Gespräch statt. Dieses wird per RTP (Real Time Protocol) über eine getrennte UDP-Verbindung transportiert. Um die Art der übertragenen Nutzdaten (Sprach-/Videodaten) zu signalisieren, wird das SDP (Session Description Protocol) verwendet. Am Ende des Gesprächs zeigt die Seite, die zuerst

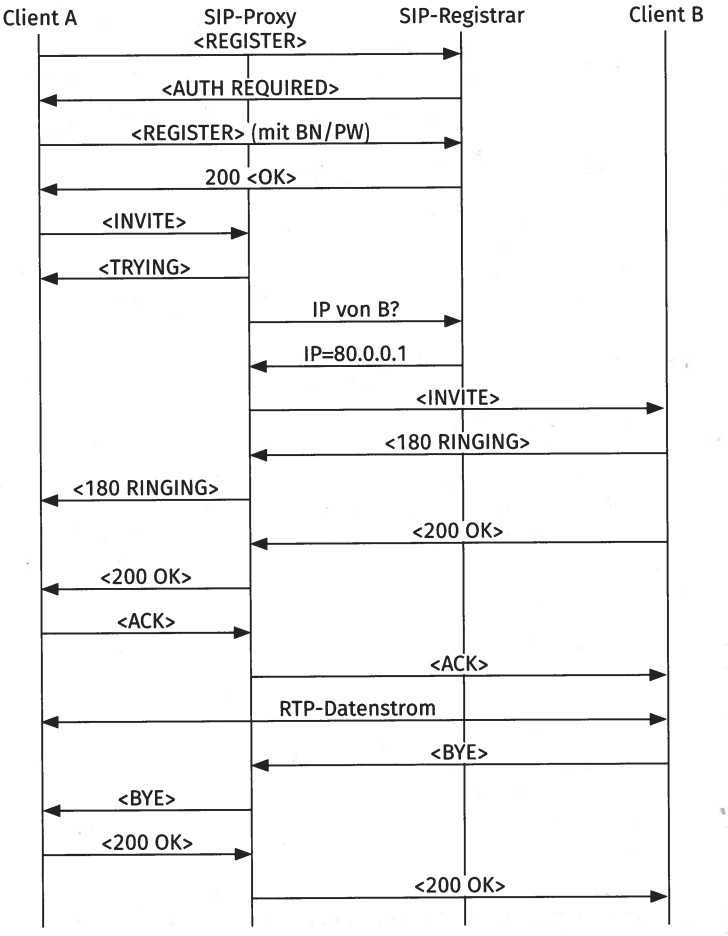
auflegt, dies mittels „BYE“ an. Die Gegenseite quittiert das mit „200 OK“. Die Zahlen (z. B. „200“) in den Statusmeldungen stammen aus der HTTP-Spezifikation (siehe HTTP-Statuscodes, S. 33). Sie wurden durch weitere Codes für SIP ergänzt (z. B. „180“).



Einfacher Verbindungsaufbau im VoIP-Netz zwischen zwei Clients

Diese Art von Verbindung setzt voraus, dass die beiden Clients die jeweilige öffentliche IP-Adresse des Gegenübers kennen. Da diese Adressen meist per Zwangstrennung durch den Provider täglich geändert werden, wird der SIP-Proxy benötigt, der beide Partner kennt und so einen Verbindungswunsch ausführen kann. Um sicherzustellen, dass die Clients zugangsberechtigt sind, wird zusätzlich der SIP-Registrar zur Authentifizierung der Account-Daten eingesetzt.

Der Verbindungsaufbau inklusive vorherige Authentifizierung kann wie folgt ablaufen (siehe nebenstehende Abbildung): Mit BN/PW sind Benutzername und Passwort gemeint. Die Registrierung ist hier nur exemplarisch für Client A gezeigt. Alle Clients müssen diese zunächst durchlaufen, damit sie erreicht werden können. Nach dem „TRYING“ erfragt der SIP-Proxy die IP-Adresse von Client B. Dies geschieht meist nicht über SIP als Protokoll. Es kann hier beispielsweise MGCP oder auch MEGACO eingesetzt werden. Mit der IP-Adresse von Client B („80.0.0.1“) kann der SIP-Proxy den Verbindungswunsch wie im Fall des einfachen Verbindungsaufbaus weitergeben.

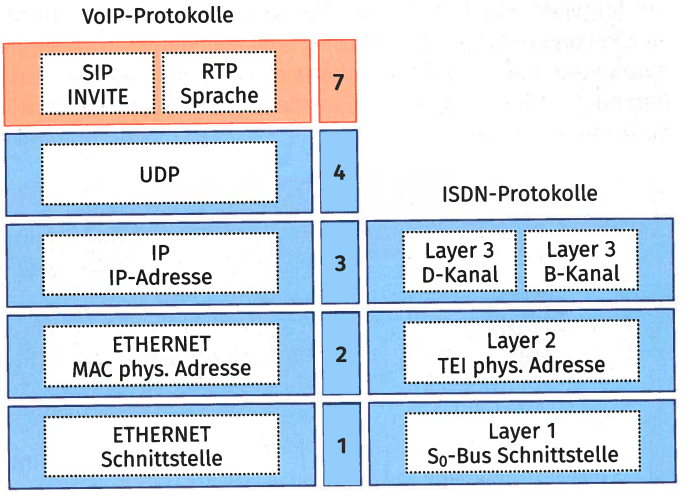


Vollständiger Verbindungsaufbau im VoIP-Netz mit Proxy und Registrar

Die Registrierung muss zyklisch erneuert werden, wodurch eine hohe Anfragelast entsteht. In modernen öffentlichen VoIP-Netzen übernimmt die Funktion des Proxy bzw. des Registrars der SBC (Session Border Controller) oder der MSAN (Multi-Service Access Node), die nah beim Kunden betrieben werden. Dies ist notwendig, um die Anfragelast auf zentrale Netzelemente gering zu halten.

VoIP verwendet die entsprechenden Protokolle bis OSI-Schicht 7 (Protokoll-Stack). ISDN kommt mit den ersten drei OSI-Schichten aus. Bei beiden Protokoll-Stacks werden die Nutzdaten von den Signalisierungsdaten getrennt verarbeitet. Bei ISDN kommen hier zwei unterschiedliche Kanäle (D-Kanal: Signalisierung und B-Kanal: Nutzdaten) zum Einsatz. Wie bei fast allen Echtzeitdatentransporten wird auch bei VoIP UDP als Transportprotokoll verwendet.

Die Verbindungsdaten, also die Information darüber, welche Endteilnehmer ein Gespräch führen, sowie die eigentlichen Sprachdaten werden durch das im Grundgesetz verankerte Fernmeldegeheimnis geschützt. Im ISDN-Netz sind diese Daten grundsätzlich durch die Kanäle vor unbefugtem Zugriff gesichert. Nur der Telekommunikationsanbieter hat im öffentlichen Netz überhaupt Zugriff auf diese Daten. Zum Schutz dieser Informationen sind Erbringer von öffentlichen Telekommunikationsdiensten durch das Telekommunikationsgesetz (§ 165 TKG) verpflichtet. SIP bieten grundsätzlich keine Möglichkeit, die Verbindungsdaten zu schützen. Alle Nachrichten werden im Klartext und sogar für Menschen „lesbar" transportiert.



Protokoll-Stacks von VoIP (SIP) und ISDN im Vergleich

Beispiel:

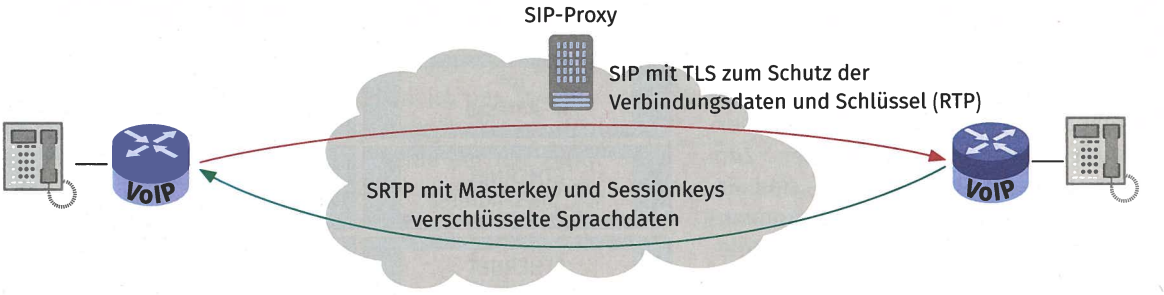
```
INVITE sip:charlie@10.0.2.25:5060 SIP/2.0
Via: SIP/2.0/UDP 10.0.2.21:5060;branch=dw3g3bK-1977-1-0
From: "PCMU/8000" <sip:zippy@10.0.2.21:5060>;tag=1
To: charlie <sip:charlie@10.0.2.25:5060>
Call-ID: 1-1977@10.0.2.21
CSeq: 1 INVITE
Contact: sip:zippy@10.0.2.21:5060
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 123

v=0
o=- 42 42 IN IP4 10.0.2.21
s=-
c=IN IP4 10.0.2.21
t=0 0
m=audio 6000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=recvonly
```

Hier können die beiden Endteilnehmer „charlie" und „zippy" direkt ausgelesen werden. Weiterhin können die Kodierung und der UDP-Port des RTP-Streams im unteren Teil ausgelesen werden. Hier: PCM my-Law mit einer Abtastrate von 8000 Hz (a=rtpmap:0 PCMU/8000) auf UDP-Port 6000 (m = audio 6000 RTP/AVP 0). Die möglichen Parameter sind in RFC 4566 spezifiziert.

Somit ist der RTP-Stream sehr leicht decodierbar und kann mit dem Programm Wireshark oder einer anderen Netzwerkanalysesoftware direkt abgespielt werden. Werden SIP oder RTP über öffentliche Netze transportiert, die nicht von einem Telekommunikationsanbieter kontrolliert werden (heterogene Umgebungen), können prinzipiell alle Daten eingesehen werden. Durch die Verschlüsselung der SIP-Kommunikation mittels TLS (SIPS) bzw. die Verwendung von SRTP kann das verhindert werden.

Die folgende Abbildung zeigt diesen Schutz. Der für den SRTP benötigte Masterkey sowie die beiden Sessionkeys müssen verschlüsselt zwischen den Endteilnehmern ausgetauscht werden, damit die Sprachdaten vor Entschlüsselung sicher sind. Die Schlüssel werden innerhalb des SIP-Protokolls ausgetauscht. Demnach muss SIP gesichert werden, damit auch die im RTP-Stream verwendeten Schlüssel gesichert übertragen werden können.



Mit TLS und SRTP gesicherte VoIP-Verbindung (RFC 3711)

Kompetenzcheck

1. Benennen Sie die Kombination aus IP-Adresse und Port.
2. Geben Sie den Bereich der Well-Known-Ports an.
3. Geben Sie die drei IP-Adressvergabeverfahren und die jeweiligen Vor-/Nachteile von DHCP an.
4. Erklären Sie die Aufgabe eines DHCP-Relay-Agent.
5. Beschreiben Sie die hierarchische Struktur eines Verzeichnisdienstes in eigenen Worten.
6. Geben Sie mindestens drei Protokolle und deren Ports an, die beim E-Mail-Dienst eingesetzt werden.
7. Erklären Sie in eigenen Worten die Validierung eines Webserver-Zertifikats.
8. Benennen Sie die Komponenten eines Videokonferenzsystems sowie die jeweiligen Aufgaben, die die Komponenten innerhalb des Systems übernehmen.
9. Bearbeiten Sie die Aufgaben 1, 2 und 3 der Lernsituation 1 im Arbeitsbuch zur Einordnung von IT-Systemen.

A1
A2
A3

1.1.2 Plattformen unterscheiden

S **Sie sollen im folgenden Abschnitt die Unterschiede von Plattformen kennenlernen.**

(1) Plattform

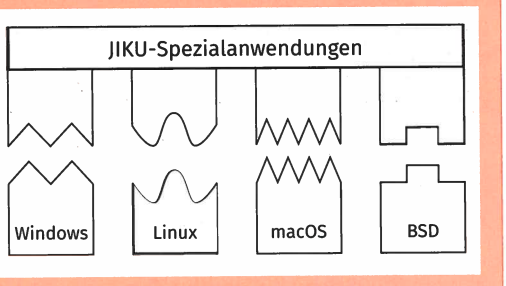
Plattform

Eine Plattform stellt eine Arbeitsbasis zur Verfügung. Die Grundlage der Plattform ist für deren Benutzer zunächst nicht relevant.

Bei vielen Tätigkeiten rund um die IT, z. B. beim Programmieren oder Bereitstellen von Diensten, kommen Plattformen zum Einsatz. Für die Benutzer vereinfacht eine Plattform die Verwendung von Programmen oder Schnittstellen, da eine Konfiguration der Hardware oder der zur Bereitstellung der Anwendersoftware benötigten Programme nicht vorgenommen werden muss. Dadurch wird die Verwendung der plattformbasierten Software oder Hardware weniger komplex.

Prinzip einer Plattform

Eine Plattform kann dem Benutzer unabhängig von der verwendeten Hardware oder vom Betriebssystem die gleiche Arbeitsumgebung zur Verfügung stellen.
In der nebenstehenden Abbildung sieht man, dass die gleiche JIKU-Spezialanwendung mit verschiedenen Betriebssystemen funktioniert. Dabei ist es notwendig, dass zu den entsprechenden Betriebssystemen spezifische Schnittstellen bestehen.



Es gibt eine ganze Reihe unterschiedlicher Plattformarten.

(2) Hardwareplattformen

Ein Beispiel für eine Hardwareplattform ist die Prozessorarchitektur „x86". Die Prozessoren dieser Architektur teilen sich beispielsweise grundlegende Befehlssätze. Der erste Prozessor mit dieser Architektur war Intels 8086 CPU im Jahr 1978. Der Befehlssatz wurde immer erweitert. Es gab beispielsweise eine 32-Bit-Erweiterung für den 80386 (1985) und eine 64-Bit-Erweiterung im Jahr 2003. Die 32-Bit- und 64-Bit-Befehlssätze unterscheiden sich.

Neben der CISC-Architektur (Complex Instruction Set Computer), zu der die x86-Prozessoren gehören, gibt es die RISC-Architektur (Reduced Instruction Set Computer). Diese enthält einen reduzierten Befehlssatz.

Architekturtyp	Bemerkungen
CISC	großer Befehlsumfang; vergleichsweise langsame Befehlsausführung; komplexere Befehle (pro Befehl mehrere Takte möglich); wenn nur Teile des Befehlsumfangs verwendet werden, gilt ein CISC als ineffizient
RISC	nur wenige elementare Befehle; schnell geladen; für jeden Befehl gibt es eine separate Schaltung (ein Takt pro Befehl); günstiger herzustellen

Ein Programm für einen RISC-Prozessor kann nicht auf einem CISC-Prozessor ausgeführt werden. x86- und x64-Systeme basieren auf einer CISC-Architektur. Allerdings wird heute ein Funktionswerk der Verarbeitung der Befehle vorgeschaltet, sodass komplizierte Befehle zerlegt werden, um diese in RISC-Befehle umzuwandeln. Tablets oder Smartphones mit einem Android-Betriebssystem basieren auf einer RISC-Architektur. Ebenso basiert Apples M2-Prozessor auf dieser Architektur.

(3) Betriebssystem als Plattform

Das Betriebssystem, z. B. Microsoft Windows oder Debian GNU/Linux, ist Grundlage für die Verwendung eines Computersystems. Es ist die Schnittstelle zwischen Anwendungsprogrammen wie LibreOffice Writer oder Microsoft Outlook und der Hardware. Das Betriebssystem ist für die Ausführung von Programmen zuständig und verwaltet beispielsweise den Speicher und die Prozessorauslastung. Dabei spielt die Hardware, auf der das Betriebssystem installiert wurde, eine untergeordnete Rolle.

Das Betriebssystem (Operating System, OS) läuft auf unterschiedlicher Hardware (CPU, GPU, Mainboard, Speichermedien), stellt aber dieselbe Funktionalität für die darauf ausgeführten Anwendungen zur Verfügung. Beispielsweise ist es für die Verwendung eines Libre-Office-Programms (meistens) egal, welche CPU zum Einsatz kommt. Die Funktionalität bleibt gleich (auch wenn es natürlich je nach Hardware unterschiedliche Ladezeiten gibt), solange das Betriebssystem alle erforderlichen Funktionen zur Verfügung stellt.

Betriebssysteme im Überblick

Betriebssystem	Einsatzgebiet	Lizenzmodell	Vor- und Nachteile
Windows	kommt in den meisten Fällen als Client-Betriebssystem zum Einsatz; wird als Betriebssystem für Büroaufgaben ebenso eingesetzt wie als Betriebssystem für Spezialanwendungen	große Anzahl verschiedener Lizenzierungsmodelle; z. B. Volumenlizenzen, OEM-Lizenzen oder Zugriffslizenzen (CAL, Client Access License; Anzahl der Zugriffe auf einen Server muss lizensiert werden)	*Vorteile:* sehr viele verfügbare Programme; verfügbare Sicherheitsupdates; gute Unterstützung von Treibern; oftmals bekannter Umgang *Nachteile:* viel Malware; nicht quelloffen; Lizenzgebühren
Windows Server	wird oft als Betriebssystem für Server eingesetzt; beinhaltet eine Vielzahl vorgegebener Rollen und somit Möglichkeiten; Microsoft bietet die Server-Lösungen auch als Cloud-Dienst an	Unterscheidung zwischen verschiedenen Betriebssystemversionen, die sich wiederum im Lizenzierungsmodell unterscheiden; z. B. Essentials, Standard- und Datacenter-Versionen, die eine Server-Lizenz oder Core-basierte Lizenz (benötigt zusätzliche Zugriffslizenzen) voraussetzen	*Vorteile:* sehr viel zusätzliche Software, z. B. für die Wartung der Server verfügbar; verfügbare Sicherheitsupdates; gute Unterstützung von Hardware durch Treiber; oftmals bekannter Umgang; durch den häufigen Einsatz stehen viele Problemlösungen zur Verfügung *Nachteile:* viel Malware; nicht quelloffen; Lizenzgebühren; oft hoher Ressourcenbedarf
Linux	steht in sehr vielen verschiedenen Distributionen zu Verfügung; bekannte Distributionen z. B. Debian, SUSE Linux, Red Hat, Ubuntu und Android; Distributionen teilen sich einen gemeinsamen Kern, unterscheiden sich jedoch in der Handhabung, der grafischen Darstellung und in der mitgelieferten Software; je nach Anwendung kann eine spezialisierte Distribution eingesetzt werden; Einsatz hauptsächlich im Unternehmensumfeld als Serverbetriebssystem	zunächst lizenzfrei und Open Source; im Unternehmenseinsatz kann es aber sinnvoll sein, eine kommerzielle Distribution zu verwenden (bieten z. B. erweiterten Support und langfristige Sicherheitsupdates)	*Vorteile:* zunächst kostenfrei; Open Source; viele Konfigurationsmöglichkeiten; niedriger Ressourcenbedarf *Nachteile:* oftmals zeitaufwendige Einarbeitung; geringere Softwareauswahl

Betriebssysteme im Überblick			
Betriebssystem	**Einsatzgebiet**	**Lizenzmodell**	**Vor- und Nachteile**
macOS	Betriebssystem von Apple; wird oft für Grafik- oder Videoanwendungen eingesetzt aber auch für Bürotätigkeiten; eine eigene Serverversion ist nicht mehr verfügbar; Serverfunktionalitäten können mittels Paketen installiert werden	wird zusammen mit Apple-Hardware vertrieben	*Vorteile:* intuitives Bedienkonzept; abgestimmter Softwaremarkt; integratives Gesamtkonzept; weniger Malware *Nachteile:* verschiedene Software nicht für macOS verfügbar; hohe Preise
BSD	verschiedene Distributionen mit verschiedenen Schwerpunkten; bekannte Vertreter sind NetBSD, FreeBSD, OpenBSD oder OPNsense (macOS basiert ebenfalls auf BSD); Einsatz im Unternehmensumfeld hauptsächlich als Serverbetriebssystem	kostenfrei und quelloffen	*Vorteile:* kostenfrei; Open Source; viele Konfigurationsmöglichkeiten; niedriger Ressourcenbedarf; sehr sicher *Nachteile:* zeitaufwendige Einarbeitung; wenig Support durch die Entwickler; im Vergleich wird weniger Hardware unterstützt

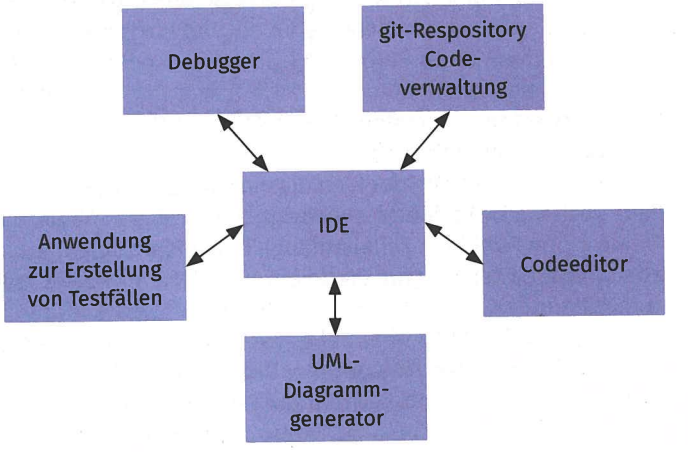
(4) Integrierte Plattform

Durch eine integrierte Plattform können alle Teilaufgaben zu einem übergeordneten Thema abgedeckt werden. Ein Beispiel für eine integrierte Plattform ist eine Mobile-Device-Management-Plattform (MDM-Plattform). Hiermit werden die mobilen Geräte einer Firma administriert. Dabei können nicht nur firmeneigene Geräte gepflegt werden, sondern es besteht auch die Möglichkeit, mitgebrachte Hardware (Bring Your Own Device, BYOD) in der Plattform zu administrieren. Für das Unternehmen hat dies den Vorteil, dass die Mitarbeiter und Gäste die Geräte verwenden können, mit denen sie vertraut sind.

Durch eine MDM-Plattform können Berechtigungen für bestimmte Dienste und Zugänge vorgegeben und die Verwendung bestimmter Anwendungen erlaubt oder verboten werden. Es ist möglich, den Datenbestand auf dem Gerät zwischen privaten Daten und Firmendaten zu trennen. Die Firmendaten können zum Erreichen des Schutzziels Vertraulichkeit verschlüsselt werden. Außerdem ist es möglich, bei einem Verlust der Hardware, z. B. durch Diebstahl, Daten, Anwendungen oder das Gerät zu sperren. Sind alle Geräte in einem Unternehmen im MDM registriert, wird auch eine Inventarisierung und daraus folgend eine Lizenzierung bestimmter Software vereinfacht.

(5) IDE als Plattform

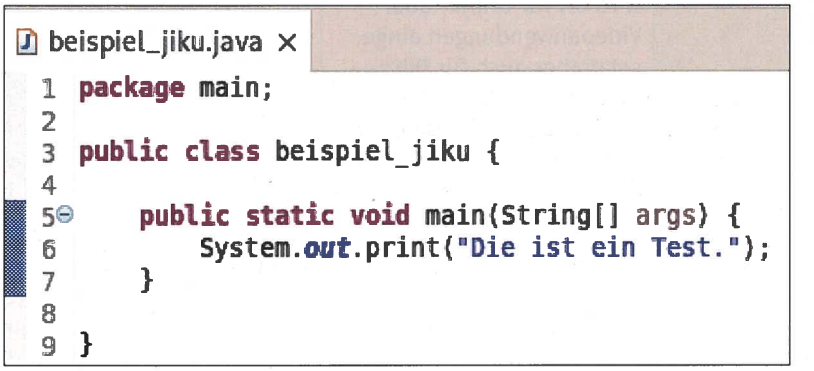
Integrated Development Enviroment (IDE) ist eine Entwicklungsumgebung für Software. Alle Werkzeuge für die Entwicklung einer Software können unter einem Programm oder einer Benutzeroberfläche vereint werden. Das hat für den Anwender den Vorteil, dass er nur eine Benutzeroberfläche bedienen muss. Die einzelnen Werkzeuge als Teil der Entwicklungsumgebung können untereinander Informationen austauschen. Dadurch können Aufgaben und Aufgabenketten vereinfacht und automatisiert werden.



Mögliche Komponenten einer IDE

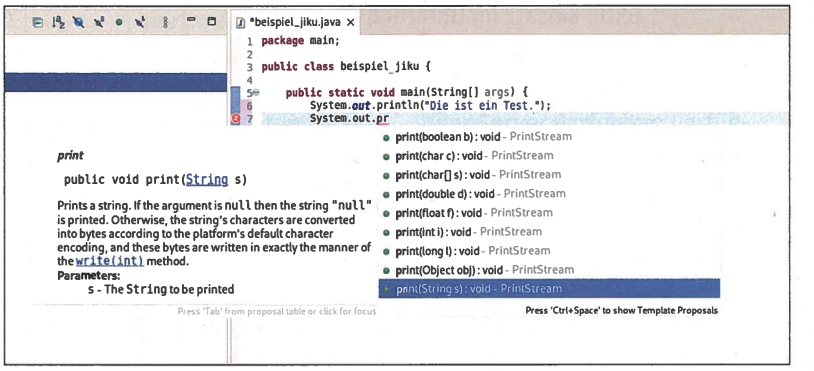
Beispiel: Mittels IDE wird Programmcode erstellt (vgl. vorangegangene Abbildung). Aus dem Programmcode wird automatisch ein UML-Sequenzdiagramm erzeugt. Der Code wird mittels Codeverwaltung versioniert und automatisch auf vereinbarte Code Conventions (Vorgaben, wie der geschriebene Code formell aussehen muss) überprüft.

Die meisten IDE unterstützen mehrere Programmiersprachen. Zu jeder IDE gehört ein **Editor**, mit dem Programmcode eingegeben werden kann. Der Editor unterstützt den Anwender durch das Hervorheben (Syntax-Highlighting) verschiedener Codeelemente, siehe z. B. in der Abbildung die farbliche Kennzeichnung von Schlüsselwörtern („public").



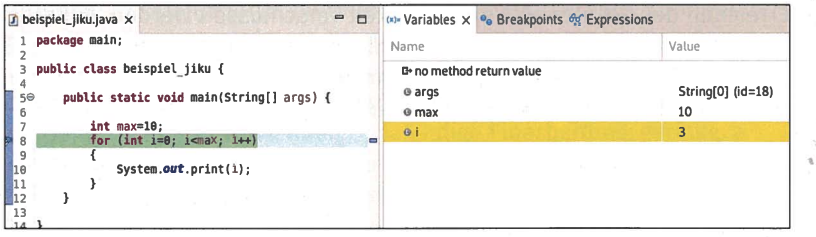
Unterstützung durch Syntax-Highlighting

Der Editor einer IDE unterstützt den Benutzer außerdem durch eine Funktion zur Textergänzung (Autovervollständigung). Dabei erhält der Benutzer Vorschläge, die begonnene Eingabe zu vervollständigen. Die nebenstehende Abbildung zeigt eine Autovervollständigung unter der IDE „Eclipse". Dabei werden dem Benutzer verschiedene Möglichkeiten für die Methode „print" (als überladene Methode) vorgeschlagen.



Autovervollständigung in der IDE

Mit dem **Debugger** kann der Ablauf des Programms untersucht werden, um z. B. zu analysieren, an welcher Stelle im Programm ein Fehler auftritt, oder um einfach das Verhalten eines Programms mit bestimmten Variablenwerten zu testen. Dazu werden im Programmcode Haltepunkte festgelegt. An diesen wird beim Debugging-Vorgang das Programm angehalten und der Wert bestimmter Variablen angezeigt. In der obigen Abbildung ist in Zeile 8 ein Haltepunkt angefügt (der kleine grüne Punkt am Zeilenanfang). Bei jedem Schleifendurchlauf wird an dieser Stelle der Programmablauf angehalten und die Werte der Variablen angezeigt (hier z. B. i = 3).



Haltepunkte im Debugger

Neben dem Editor und dem Debugger gehören zu einer IDE je nach Programmiersprache auch ein Compiler oder Interpreter. Ein **Compiler** übersetzt den Programmcode in eine maschinenlesbare Sprache und macht so das Programm ausführbar. Ein **Interpreter** führt den Programmcode direkt aus. Zusätzlich zu Editor, Debugger und Compiler/Interpreter kann eine IDE noch weitere Module oder Programmteile beinhalten. Bekannte IDE sind beispielsweise Eclipse, NetBeans, PyCharm oder Microsoft Visual Studio.

(6) Cloud als Plattform beschreiben

Nach dem National Institute of Standards and Technology (NIST) sind folgende essenzielle Charakteristiken nötig, um von Cloud-Computing zu sprechen:

Charakteristiken von Cloud-Computing

On-demand Self-service
Ein Kunde kann sich selbstständig Dienste hinzu- oder abbuchen, z. B. Rechenleistung oder Speicher, ohne mit dem Diensteanbieter persönlich in Kontakt treten zu müssen.

Broad Network Access
Der Zugang zu den Diensten ist über ein Netzwerk möglich und für ein breites Spektrum an Endgeräten verfügbar, z. B. Smartphone, Tablets oder Computer.

Resource Pooling
Die Rechenleistung des Anbieters wird zusammengefasst mehreren Kunden angeboten. Dadurch sind variable Skalierung und Zuordnung möglich. Kunden kennen den genauen Standort der Speicher- und Rechenleistung nicht, können aber aus Datenschutzgründen auf einer abstrakten Ebene den Standort der Daten auf Regionen oder Länder eingrenzen. Aufgrund von Datensicherheit werden in den gewählten Regionen georedundante Standorte der Datenspeicher mit Mindestabstand festgelegt. Das BSI legt einen Mindestabstand von 200 km für georedundante Rechenzentren fest (vgl. „Kriterien für die Standortwahl von Rechenzentren", Version 2.0, Bundesamt für Sicherheit in der Informationstechnik, 2019).

Rapid Elasticity
Kapazitäten können elastisch provisioniert und wieder freigegeben werden, basierend auf festgelegten Metriken.

Measured Service
Ein grundlegendes Bezahlmodel von Cloud-Computing ist **pay-per-use**. Es wird nur für die Ressourcen bezahlt, die auch genutzt wurden. Eine Grundlage dafür sind die Überwachung und Berichterstattung über die genutzten Ressourcen.

(siehe auch Jahrgangsband 1, Lernfeld 2, Kapitel 2.4.13 und Lernfeld 3, Kapitel 3.3.5)

Neben der grundlegenden Definition von Cloud-Computing beschreibt das NIST in der genannten Publikation **Deployment Models** und **Service Models**. Ein Deployment Model ist die **Public Cloud**. Die großen Anbieter wie Amazon, Google oder Microsoft haben dafür eigene Implementierungen erstellt. Um sich mit dem jeweiligen Anbieter vertraut zu machen und eine Aussage über die Kenntnisse zu dem Cloud-System des jeweiligen Anbieters zu machen, werden Kurse angeboten. Diese erlauben den Wissensstand über Zertifikate zu dokumentieren. Grundlagenkurse mit Zertifikat, die auf einer nicht technischen Ebene starten und einen Überblick über den Dienst geben, sind z. B.:

- Amazon Web Services (AWS): Cloud Practitioner
- Microsoft Azure: AZ900 Fundamentals
- Google Cloud Platform (GCP): Cloud Digital Leader

Weitere Zertifikate auf mittlerer und fortgeschrittener Ebene werden zunehmend technischer.

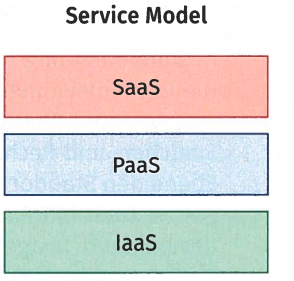
Die Frage, welcher Anbieter der richtige für ein Projekt ist, kann von vielen Aspekten abhängen. Sinnvoll ist es, durch ein Pilotprojekt zu ermitteln, ob der jeweilige Anbieter die nötigen Ressourcen anbietet, die Dokumentation passend, das Angebot robust verfügbar ist und ob der Dienst bedienbar ist.
Die Möglichkeit einen **Vendor Lock-in** zu vermeiden ist, eine Abstraktion einzuführen, die auf allen Plattformen gleichermaßen umgesetzt werden kann. Die Containerisierung ist eine solche Abstraktion. Ein Beispiel dafür ist **Kubernetes**, eine Container-Orchestrierungsplattform, die bei gängigen Cloud-Anbietern lauffähig ist.

Vendor Lock-in

In der Wirtschaftswissenschaft wird der Begriff „Vendor Lock-in" als ein Zustand beschrieben, bei dem ein Kunde sich aufgrund einer technischen Umsetzung an einen speziellen Anbieter bindet. Der Wechsel zu einem neuen Anbieter würde hohe Entwicklungskosten mit sich bringen, was die Bereitschaft zu einem Anbieterwechsel einschränkt. Solange die Preisanstiege des Anbieters im Verhältnis der Entwicklungskosten für einen Anbieterwechsel gering sind, ist der Kunde gebunden (Lock-in).

Kontrovers ist, ob ein Vendor Lock-in durch die parallele Verwendung mehrerer Cloud-Anbieter vermieden werden kann. Jedoch erscheint es sinnvoller, sich auf einen Anbieter einzulassen und die Ressourcen dafür zu verwenden, diesen Dienst optimal und effizient zu nutzen. Durch die bereits erwähnte Abstraktion ist dann ein Umzug nicht ausgeschlossen.

Beispiel: Amazon Web Service



Exemplarisch am Amazon Web Service (AWS) sollen einige Dienste beschrieben und nach dem Service Model des NIST geordnet werden. Im Jahre 2023 stellt der Amazon Web Service über 200 Dienste zur Verfügung. Die drei Service Models bauen aufeinander auf und die unterste Ebene stellt die **Infrastructure as a Service (IaaS)** dar. Hierbei geht es darum, Rechenleistung, Speicher, ein Netzwerk oder andere grundlegende Ressourcen zur Verfügung zu stellen.

Einer der fundamentalsten Dienste ist der **Elastic Compute Cloud 2 (EC2)**. Dieser Dienst erlaubt es, einen virtuellen Computer mit Betriebssystem und gewünschter Rechenleistung zu erstellen. Um jetzt die Last zwischen verschiedenen Servern zu verteilen, wird über den Dienst „Load Balancing" eine Verteilung der Anfragen übernommen. Ob noch ein weiterer Server nötig ist oder vielleicht sogar einer abgeschaltet werden kann, weil gar nicht mehr so viele Anfragen für die zur Verfügung gestellte Anwendung ankommen, kann über den Dienst „Cloud Watch" ermittelt werden. Dieser Dienst sammelt Metriken über die Ausnutzung und Anfrage an die Serverinstanzen. Damit das Erstellen oder Beenden von Serverinstanzen nicht händisch, sondern automatisiert auf Grenzwerte der Metriken erfolgen kann, steht der Dienst „Auto Scale" bereit.

In die Kategorie IaaS fallen z. B. auch Speicherdienste. Der **Simple Storage Service (S3)** ist einer der ursprünglichsten dateibasierten Speicherdienste von Amazon. „Glacier" ist ein Archivspeicher. Die Zugriffszeiten auf ihn sind länger, dafür ist er kostengünstiger als beispielsweise „Block Storage", ein Dienst für Blockspeicher mit schnellem Zugriff, der aber eine eigene Verwaltung benötigt. „Elastic File System" ist im Gegensatz zu „Block Storage" ein schneller, dateibasierter Speicher, der eine Verwaltung mitbringt und dadurch mehr Geld kostet.

An dem Beispiel wird klar, warum es so viele Dienste gibt. Mehrere Dienste liefern ähnliche Funktionen, mit unterschiedlichen Aspekten, die dann entsprechend anders bepreist werden. Aufbauend auf IaaS ist die nächsthöhere Ebene im NIST-Service-Model der **Platform as a Service (PaaS)**. AWS bietet hierzu beispielhaft den Dienst „Lightsail". Mit ihm werden „containerisierte" Anwendungen zur Verfügung gestellt, z. B. WordPress, LAMP, GitLab, Django oder Plesk Hosting Stack on Ubuntu.

Abschließend sollen noch der „Container Service" und der „Kuberentes Service" erwähnt werden, die beide in die Kategorie der zuvor erwähnten Abstrahierung fallen und einen Weg darstellen, einen Vendor Lock-in zu vermeiden.

Kompetenzcheck

1. Geben Sie mindestens drei Plattformarten an.
2. Erklären Sie den Unterschied zwischen RISC- und CISC-Prozessoren.

3. Nennen Sie mindestens drei Betriebssystemplattformen.
4. Erklären Sie die beiden Begriffe „MDM“ und „BYOD“.
5. Geben Sie die Bestandteile einer IDE an.
6. Erklären Sie den Begriff „Resource Pooling“ im Zusammenhang mit Cloud-Systemen.
7. Beschreiben Sie, was mit dem Begriff „Vendor Lock-in“ in Bezug auf Cloud-Anbieter gemeint ist.
8. Bearbeiten Sie die Aufgabe 4 der Lernsituation 1 im Arbeitsbuch.

A4

1.2 Serverdienste und Plattformen gemäß Kundenanforderungen auswählen

S Sie sollen die Anforderungen an Serverdienste und Plattformen formulieren.

Die Kunden der JIKU IT-Solutions GmbH haben neben den grundlegenden Anforderungen an den Betrieb von Systemen auch anwendungs- und dienstspezifische Anforderungen an den Betrieb der IT-Infrastruktur. So sollen beispielsweise die Antwortzeiten der Systeme möglichst kurz und das Schutzziel der Verfügbarkeit idealerweise ununterbrochen gewährleistet sein. Diese Anforderungen können nur durch geeignete Auswahl der Basistechnologien erfüllt werden. Solche Technologien im späteren Betrieb zu ändern, kann große Aufwände nach sich ziehen. Im folgenden Kapitel werden zum einen grundlegende Anforderungen an den Betrieb der IT-Infrastruktur und zum anderen Varianten der Basistechnologien beschrieben.

1.2.1 Physische Server planen

S Sie werden für die Planung physischer Server die Administrierbarkeit, die Wirtschaftlichkeit und andere Aspekte berücksichtigen.

(1) Verfügbarkeit

Das Schutzziel der Verfügbarkeit (Availability) besagt, dass berechtigte Benutzer beim Zugriff auf Informationen und Systeme nicht behindert werden sollen. Ein Server oder dessen Dienste und Daten müssen erreichbar sein (siehe auch Jahrgangsband 1, Lernfeld 4, Kapitel 4.1.3). Diese Erreichbarkeit wird beim Betrieb von Serversystemen durch verschiedene Techniken gewährleistet. Damit ein System auch bei Spannungseinbrüchen oder -schwankungen erreichbar bleibt, werden unterbrechungsfreie Stromversorgungen (USV) eingesetzt (siehe auch Jahrgangsband 1, Lernfeld 3, Kapitel 3.9.1). Das Netzteil von Serversystemen wird oftmals redundant ausgelegt. Somit bleibt das System verfügbar, auch wenn ein Netzteil ausgefallen ist.

Redundante Stromversorgung
Verfügt ein System über mehr als ein Netzteil, so spricht man von einer redundanten Stromversorgung. Ein solches System muss nicht immer ein Server sein, auch Koppelelemente wie Switches oder Router können über eine redundante Stromversorgung verfügen. Jedes der mehrfach vorkommenden Netzteile kann das System alleine mit Strom versorgen, sodass trotz eines Ausfalls eines Netzteils weiterhin ein stabiler Betrieb des Systems und damit die Verfügbarkeit gesichert ist.

Im Normalfall wird eine redundante Stromversorgung eines Servers durch eine Dopplung des Netzteils erreicht. Die beiden Netzteile werden durch ein separates Gehäuse miteinander verbunden. In diesem Gehäuse ist die Steuerung der Netzteile (Backplane) untergebracht. Die Netzteile in Serversystemen wer-

den oftmals überwacht, z. B. durch den Baseboard Management Contoller (BMC), damit ein Ausfall bemerkt wird und das defekte Netzteil im laufenden Betrieb getauscht werden kann. Der Austausch findet ohne das Herunterfahren des Systems statt und wird **Hot Swapping** genannt.

Neben der Stromversorgung kann auch das eigentliche Serversystem redundant ausgelegt werden. Dabei gibt es verschiedene Ansätze, die redundanten Systeme zu verwenden.

Cluster
Ein Cluster ist ein Zusammenschluss von vernetzten Systemen. Für einen Benutzer erscheint dieser Zusammenschluss allerdings nur als ein System. Ein Cluster wird betrieben, um eine hohe Verfügbarkeit (High Availability) zu erreichen. Außerdem kann er dazu verwendet werden, um Programme mit einem hohen Ressourcenbedarf verteilt auf mehreren Systemen arbeiten zu lassen (High Performance Computing).

Ziel eines Hochverfügbarkeitsclusters ist es, die Ausfallsicherheit eines Dienstes zu gewährleisten. Ein solches Cluster verfügt über mindestens zwei Einzelsysteme (auch Knoten genannt). Es werden Active/Active-Cluster und Active/Passive-Cluster unterschieden.

Bei einem **Active/Active-Cluster** läuft der Dienst auf allen Systemen aktiv. Beim Ausfall eines Knotens übernehmen die anderen Knoten dessen Arbeit. Die verteilte Software muss diesen Clustertyp unterstützen oder entsprechend anpassbar sein.

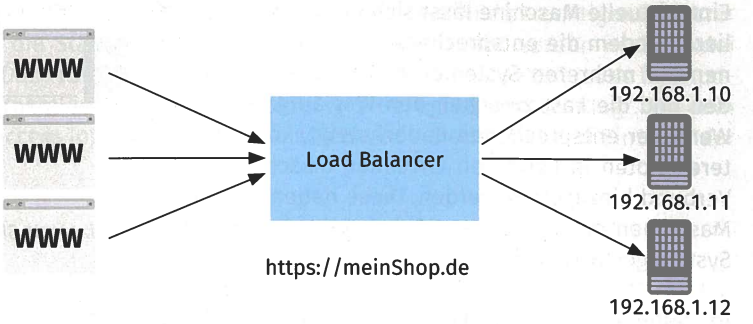
Bei einem **Active/Passive-Cluster**, auch Failover genannt, gibt es ein aktives und ein passives System. Wenn das aktive System ausfällt, übernimmt das passive dessen Aufgaben. Wenn das aktive System funktioniert, hat das passive keine Aufgabe.

Beim Teilen von Ressourcen, z. B. bei einem Active/Active-Cluster, wird der Grad der Autonomie eines Knotens beschrieben. Bei einer **Shared-nothing-Architektur** ist jeder Knoten komplett autonom und kann Aufgaben selbstständig durchführen. Dazu hat der Knoten beispielsweise einen eigenen Speicher, eine eigene Prozessorleistung und ein eigenes Betriebssystem. Bei dieser Architektur kann durch das Hinzufügen weiterer Knoten die Leistung des Gesamtsystems angehoben werden. Dadurch wird das System sehr gut skalierbar. Bei einer **Shared-all-Architektur** (als Gegenstück zur Shared-nothing-Architektur) werden sämtliche Ressourcen geteilt. Ein einzelner Knoten kann nicht mehr autonom arbeiten.

Load Balancer
Wenn ein Server, also ein Dienst, als Gesamtsystem mehrfach vorhanden ist, z. B. in einem Cluster, muss der Verkehr für diesen Dienst auf die einzelnen Systeme verteilt werden. Ein Load Balancer kann software- oder hardwarebasiert ausgeführt sein. Die Auswahl, an welchen Server eine Anfrage weitergeleitet wird, kann auf verschiedenen Wegen geschehen.

Beispiele für verwendete Algorithmen	
Verfahren	**Erklärung**
der Reihe nach (Round Robin)	Aus der Liste der vorhandenen Server wird einer der Reihenfolge nach ausgewählt.
wenigste Verbindungen (Least Connections)	Der Server mit den wenigsten Verbindungen wird ausgewählt.
schnellste Antwortzeit (Least Response Time)	Der Server mit kürzester Antwortzeit wird ausgewählt.
schlüsselbasiert (Hash Based)	Aus Parametern des anfragenden Clients wird ein Schlüssel gebildet; Anfragen dieses Clients werden immer zum gleichen Server gesendet.
zufallsbasiert (Random Based)	Aus der Liste der verfügbaren Server wird ein System per Zufall ausgewählt.

Load Balancer können auf verschiedenen Schichten des OSI-Modells arbeiten. Als Beispiel soll die OSI-Schicht 4 (TCP/UDP) und OSI-Schicht 7 (Anwendungsschicht, z. B. HTTP) beschrieben werden. Beim Load Balancing auf OSI-Schicht 4 werden Anfragen basierend auf IP-Adresse und Port an den Load Balancer gesendet. Dieser leitet die Anfrage an einen der redundanten Server weiter. Die höheren Protokolle werden nicht beachtet. Beim Load Balancing auf OSI-Schicht 7 basieren die Entscheidungen auf der Anwendungsschicht. Hier wird das Protokoll der Anwendungsschicht, z. B. HTTP, im Load Balancer terminiert. Basierend auf dem Inhalt des Protokolls wird die Anfrage an einen Server weitergeleitet. Je höher die Auswertung in der Hierarchie des OSI-Modells stattfindet, desto passender kann eine Lösung für komplexere Dienste implementiert werden. Ein gutes Beispiel ist eine Webseite, auf der eine Einwahl mit Sitzung stattfindet. Hier ist es sinnvoll, die ganze Sitzung an den gleichen Server weiterzuleiten, damit keine Synchronisation des Sitzungsstatus über verschiedene Server stattfinden muss.

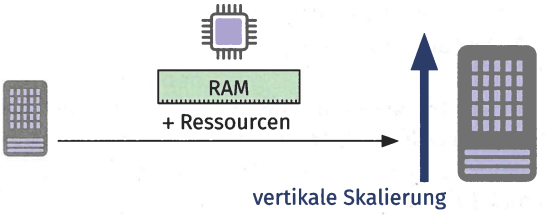


Arbeitsweise eines Load Balancer

(2) Skalierbarkeit

Unter Skalierbarkeit versteht man die Möglichkeit, ein System an geänderte Anforderungen anzupassen. In Bezug auf IT-Systeme beschreibt Skalierbarkeit meist die Vergrößerung oder Verkleinerung von Rechenleistung, um ein System an die Anfragen anzupassen. Im Normalfall muss ein System vor allem die Möglichkeit besitzen, vergrößert, also hochskaliert zu werden (vgl. Jahrgangsband 2, Lernfeld 9, Kap. 4.1.1, Abschnitt (3)).

Eine Skalierung wird in vertikale und horizontale Skalierung unterschieden. Bei **vertikaler Skalierung** werden dem Serversystem Ressourcen, z. B. CPU-Kerne, Arbeitsspeicher, Netzwerkbandbreite, hinzugefügt. Das System wächst vertikal. Für die meisten Softwaresysteme, z. B. Webserver oder Datenbanksysteme, ist die vertikale Skalierung kein Problem, da sie keine Anpassung der Software erfordert. Eine einfache vertikale Skalierung wäre es beispielsweise, einen Server mit mehr RAM auszustatten. Diese Skalierungsart ist meist kostengünstig und einfach durchführbar. Allerdings kann ein physisches System oft nicht grenzenlos erweitert werden. Durch Beschränkungen, z. B. durch das Mainboard, ist eine Aufrüstung nur beschränkt möglich.



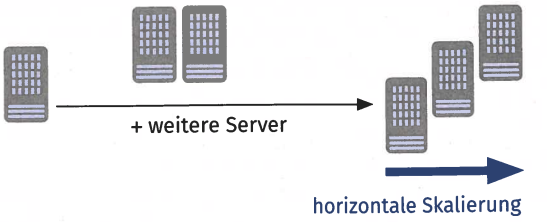
Vertikale Skalierung

Ein VM-basierter Dienst kann in den meisten Fällen gut skaliert werden. Der entsprechenden virtuellen Maschine können mehr Ressourcen wie Arbeitsspeicher und CPU-Leistung zugeordnet werden.

Im Serverbereich werden zur einfachen vertikalen Skalierbarkeit auch sogenannte Blade-Serversysteme eingesetzt. Diese physischen Server haben ein gemeinsames Gehäuse. In dieses Gehäuse, auch Blade-Center genannt, werden die eigentlichen Blade-Server gesteckt. Sie teilen sich die Stromversorgung, die Kühlung und die Netzwerkanbindung des Gehäuses. Jeder Blade-Server verfügt über eigene Prozessoren, eigenen Arbeitsspeicher und Speicherplatz. Ein Blade-System kann bis zur seiner maximalen Aufnahmekapazität erweitert werden.

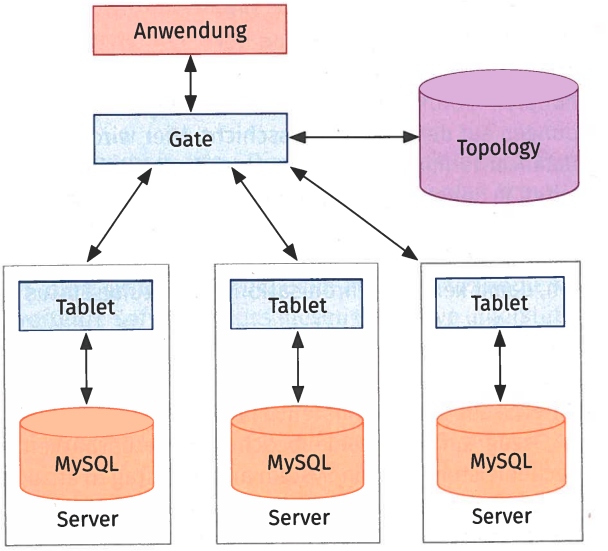
Bei einer **horizontalen Skalierung** werden dem Serversystem, auf dem ein Dienst gehostet wird, weitere Systeme (Knoten) hinzugefügt. Eine horizontale Skalierung hat den Nachteil, dass alle eingebundenen Systeme Strom verbrauchen, Abwärme erzeugen und lizenziert werden müssen. Außerdem benötigt jedes zusätzliche System Platz.

Eine virtuelle Maschine lässt sich gut horizontal skalieren, indem die entsprechenden virtuellen Maschinen auf mehreren Systemen parallel gestartet werden und die Last zwischen den VMs aufgeteilt wird. Wenn der entsprechende Bedarf steigt, können weitere Knoten in Form von virtuellen Maschinen dem Verbund hinzugefügt werden. Diese neuen virtuellen Maschinen müssen nicht auf demselben physischen System gestartet werden.



Horizontale Skalierung

Im Jahrgangsband 2, Lernfeld 8, Kapitel 3.7 wird in Bezug auf Datenbanken die vertikale Skalierung für relationale Datenbanken und die horizontale für NoSQL-Datenbanken beschrieben. Um relationale Datenbanken auch horizontal zu skalieren, ist mehr Aufwand nötig. Ein Beispiel für eine horizontale Skalierung einer relationalen Datenbank ist das Vitess-Projekt (siehe unter https://vitess.io). Mithilfe von Vitess greift eine Anwendung über ein **Gate** auf die Datenbank zu. Das Gate akzeptiert SQL-Anfragen und repräsentiert das Datenbank-Cluster als eine monolithische Datenbank. Die Verbindung zur eigentlichen Datenbankinstanz erfolgt über ein **Tablet**. Eine **Topology**-Datenbank enthält die Daten darüber, welche Instanzen vorhanden sind und für Anfragen zur Verfügung stehen.



Skalierung bei Datenbanken

(3) Administrierbarkeit

Ein System ist gut administrierbar, wenn bestimmte Kriterien erfüllt sind. Das System muss **konfigurierbar** sein. Dabei sollte es für **berechtigte** Personen möglich sein, alle Parameter des Systems oder des Dienstes einzustellen. Damit eine Konfiguration sinnvoll und sicher durchgeführt werden kann, ist eine gute **Dokumentation** der betriebenen Dienste und Systeme wesentlich. Dabei ist es wichtig, dass ein Dienst oder System mit **unterschiedlichen Berechtigungen** umgehen kann. Es sollte beispielsweise zwischen Benutzern und Personen mit erweiterten Rechten unterscheiden.

Zur Administrierbarkeit eines Systems gehört es auch, dass das System gut zu pflegen und zu überwachen ist. Ein gut zu administrierendes System benötigt **regelmäßige Sicherheitsupdates**. Dazu muss die Software des Systems weiterhin durch Entwickler gepflegt werden. Es ist sinnvoll, dass eine **Überwachung** des Systems oder des Dienstes möglich ist. Die Möglichkeit der Überwachung sollte grundlegend mit log-Dateien umsetzbar sein, damit eine schnelle und zielgerichtete Fehleranalyse durchgeführt werden kann. Weitergehend sollte der Dienst oder das System mittels „Monitoring" auf seine Verfügbarkeit überwacht werden können.

Beispiel: Webserver „lighttpd"

Die Software „lighttpd" steht unter einer Open-Source-Lizenz. Der Webserver läuft unter Unix-Derivaten und kann bspw. unter verschiedenen Linux-Distributionen über den Paketmanager der Distribution installiert werden, z. B. unter Debian mit dem Befehl „sudo apt install lighttpd".

Die Standardkonfiguration des Webservers findet über die Datei „/etc/lighttpd/lighttpd.conf" statt. Eine andere Datei kann beim Start des Webservers angegeben werden. In der Konfigurationsdatei können alle Einstellungen detailliert vorgenommen werden. Die Dokumentation zur Konfiguration findet sich auf der Webseite des Entwicklers.

Die Benutzerberechtigungen werden durch das Betriebssystem übernommen. Nur berechtigte Benutzer können den Webserver starten oder die Standardkonfiguration verändern. Fehler werden in der Datei „/var/log/lighttpd/error.log" notiert. Das Protokollieren des Zugangs wird über den Befehl „sudo lighttpd-enable-mod accesslog" aktiviert. Protokolliert wird in der Datei „/var/log/lighttpd/access.log". Mit dem Befehl „sudo tail /var/log/lighttpd/access.log -n 3" können beispielsweise die letzten drei Einträge der log-Datei angezeigt werden.

(4) Wirtschaftlichkeit

Wirtschaftlichkeit

Allgemein ist die Wirtschaftlichkeit ein Maß für die Effizienz. Erträge werden durch Aufwendungen geteilt. Im Falle von Servern sind die Aufwendungen die Anschaffungs- und Zusatzkosten sowie die Folgekosten. Zu den Zusatzkosten gehören z. B. Installationskosten, Schulungskosten und Kosten für den Einarbeitungsaufwand. Zu den Folgekosten zählen z. B. die Kosten für die Energie, die ein System benötigt.

Der Ertrag eines Servers kann in manchen Fällen nicht eindeutig bestimmt werden, da der erbrachte Dienst des Servers oftmals nicht monetär festgelegt werden kann. Bei der Entscheidung für ein Serversystem werden oftmals die verschiedenen Realisierungsmöglichkeiten und Nutzungsmodelle betrachtet. Ein Dienst kann auf einer dedizierten Hardware oder auf einer virtuellen Maschine innerhalb des Unternehmens installiert sein. Eine Installation vor Ort wird **On-Premise** genannt. Alternativ kann ein Dienst aber auch in der Cloud als Software as a Service (SaaS) gemietet werden. Jedes dieser Modelle bietet Vor- und Nachteile, die bei der Betrachtung der Wirtschaftlichkeit beachtet werden müssen (siehe auch Jahrgangsband 1, Lernfeld 1, Kapitel 1.3.3 und Lernfeld 2, Kapitel 2.3.3).

Wenn ein Dienst vor Ort betrieben wird, hat der Betreiber volle Kontrolle über das System und vor allem die Daten, die auf dem System bearbeitet werden. Gerade bezüglich der DSGVO ist es wichtig, dass nachvollziehbar ist, wo und wie die Daten verarbeitet und gespeichert werden.

Vorteile von On-Premise- und SaaS/Cloud-Lösungen	
On-Premise	**SaaS/Cloud**
• volle Kontrolle über den Server • Hoheit über die Daten • einfachere Integration in die vorhandene Umgebung • keine laufenden Mietkosten • unabhängig von einer Internetverbindung	• weniger Wartungsaufwand • Sicherheitsupdates werden durch den Anbieter installiert • keine Anschaffungskosten • einfache Skalierung • von überall erreichbar

Bei einer wirtschaftlichen Betrachtung müssen die Vor- und Nachteile beachtet und gewichtet werden. Als Beispiel wird die Anschaffung eines Datenbank-Servers für nicht unternehmenskritische Daten betrachtet.

Vergleich von On-Premise und SaaS/Cloud		
Kosten	**On-Premise**	**SaaS/Cloud**
Anschaffungskosten	vorhanden (ggf. mehrere tausend Euro)	nicht vorhanden
Zusatzkosten	vorhanden (ggf. tausend Euro)	nicht vorhanden
Folgekosten	vorhanden (z. B. Strom, Lizenzkosten, Personalkosten)	monatliche Mietkosten
Wartungskosten	vorhanden (z. B. Updates)	nicht vorhanden

(5) Sicherheit

Sicherheit

Ein Serversystem muss, besonders wenn ein Zugang zum öffentlichen Netzwerk besteht, gesondert abgesichert werden. Dabei wird nicht nur der Netzwerkzugang beispielsweise durch eine Firewall geschützt, sondern der Server selbst wird „gehärtet".

Die Härtung eines Systems wird durch das BSI als „die Entfernung aller Softwarebestandteile und Funktionen, die zur Erfüllung der vorgesehenen Aufgabe durch das Programm nicht zwingend notwendig sind" (Bundesamt für Sicherheit in der Informationstechnik: Leitfaden Informationssicherheit. IT-Grundschutz kompakt) definiert. Das bedeutet, dass alle nicht für das System nötigen Funktionen deaktiviert werden (siehe auch Jahrgangsband 1, Lernfeld 4, Kapitel 4.1.3).

Zur Härtung eines Servers sollten
- alle nicht benötigten Dienste deaktiviert,
- jede nicht benötigte Anwendung deinstalliert,
- alle (Sicherheits-)Updates installiert,
- alle nicht benötigten Benutzerkonten deaktiviert,
- alle Standardpasswörter geändert und
- ein verbindliche Zugangskontrolle konfiguriert werden.

Ein System muss aber nicht nur über das Betriebssystem gehärtet werden, sondern auch die Hardware eines Serversystems muss, z. B. durch eine Zugangskontrolle zum Serverraum, geschützt werden. Das BSI bietet mit dem Grundschutzhandbuch und dessen Bausteinen und Konfigurationsempfehlungen eine Hilfestellung für die administrierende Person, um allgemein Server und bestimmte Serverarten, z. B. Webserver, systematisch zu härten.

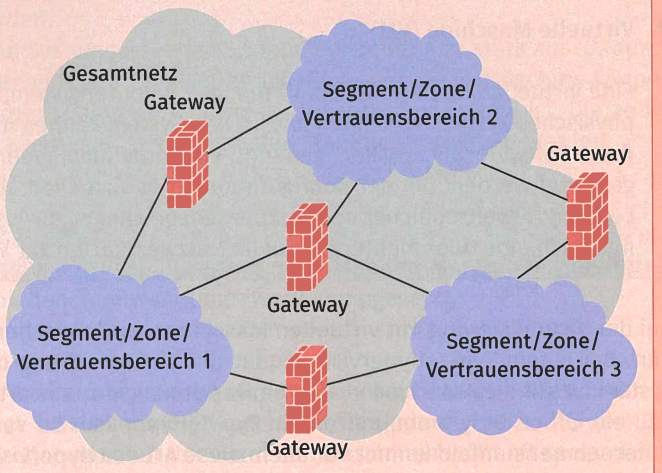
Wichtige Empfehlungen zur Härtung eines Server

Perimetersicherheit
Ein Perimeter ist eine Grenze. Ziel der Perimetersicherheit ist es, nur an ausgewählten Punkten, den Netzübergängen, einen kontrollierten Zugang in ein Netzwerk zuzulassen. Das BSI fordert nicht nur den Schutz an der Netzwerkgrenze, sondern auch den Schutz von Infrastruktur, z. B. den Zugang zu Gebäuden mit einer Umzäunung, Personenkontrolle und Alarmanlagen o. Ä. Dazu gibt das BSI Hilfestellung im Grundschutzbaustein „INF.2: Rechenzentrum sowie Serverraum".
Zur Durchsetzung von Perimetersicherheit werden Firewalls eingesetzt. Eine Firewall erlaubt oder verbietet eingehenden und ausgehenden Datenverkehr aufgrund von Regeln. Durch die Perimetersicherheit können Angriffe auf das Netzwerk unterbunden werden. Wenn Angreifer allerdings die Firewall, also die Grenze, überwinden, können sie im Netzwerk aktiv werden. Fällt die Firewall als einziger Zugangspunkt zum Netzwerk aus, ist auch jeglicher Verkehr in externe Netzwerke unterbunden. Dann können beispielsweise E-Mails nicht empfangen oder externe Webanwendungen nicht verwendet werden. Die Firewall ist ein „Single Point of Failure", der z. B. durch gezielte Angriffe stillgelegt werden kann (Denial of Service).
Perimetersicherheit als einzige Schutzmaßnahme ist in heutigen Computernetzwerken kein ausreichender Schutz mehr. Techniken wie beispielsweise Bring Your Own Device (BYOD), bei der Mitarbeiter ihre eigenen Geräte für die Arbeit im Firmennetzwerk verwenden können, oder auch verschlüsselte Verbindungen, die die Firewall aufgrund der Verschlüsselung nicht kontrollieren kann, unterlaufen das Konzept der Perimetersicherheit (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.2 und 4.4.2).

Zero Trust

Ein anderes Prinzip in der IT-Sicherheit ist das Zero-Trust-Prinzip. Es kann als Gegenentwurf oder als Erweiterung zur Perimetersicherheit angesehen werden. Die Standardisierungsbehörde NIST (National Institute of Standards and Technology) definiert Zero Trust als Konzept, bei dem sinnvolle Zugriffsentscheidungen getroffen werden. Bei einem Zugriff werden nur die Rechte gewährt, die für den jeweiligen Zugriff nötig sind. Dazu muss die Zugriffsregelung möglichst detailliert erfolgen und die Berechtigungen müssen granular erteilt werden können. Jeder Zugriff auf das System muss durch definierte Richtlinien festgelegt werden. Diese Richtlinien werden bei jedem Zugriff erneut geprüft. Der Zugriff auf ein System darf nur nach einer erfolgreichen Authentifizierung, z. B. mittels 2FA, erfolgen. Ein Zugriff muss verschlüsselt sein. Außerdem müssen alle Zugriffe überwacht und protokolliert werden (vgl. Jahrgangsband 2, Lernfeld 9, Kap. 4.4.2, Abschnitt (4)).

- **Identitätsorientierter Ansatz:** Die Richtlinien für die Überprüfung und den Zugriff auf ein System basieren auf Identitäten. Eine Identität kann ein Benutzer, aber auch ein anderes System oder ein anderer Dienst sein.
- **Netzwerkorientierter Ansatz:** Bei diesem Ansatz wird das Gesamtnetz in kleine Bereiche unterteilt. Die einzelnen Bereiche werden durch Gateways getrennt. Dadurch gibt es verschiedene Vertrauensbereiche (Zonen) im Netzwerk. Dieser Ansatz entspricht grundlegend einem abgestuften Perimetersicherheitssystem und hat den Nachteil, dass keine weitere Überprüfung stattfindet, wenn ein Angreifer den Zugang zu einem Bereich bekommen hat. Die Abbildung zeigt eine Einteilung des Gesamtnetzes in drei Vertrauensbereiche. Die Bereiche sind mittels Gateways, in diesem Fall Firewalls, verbunden. An diesen Punkten findet die Überprüfung der Berechtigung anhand von Richtlinien (oder Regeln) statt.



Zonenprinzip mit drei Zonen

- **Cloudbasierter Ansatz:** Dieser Ansatz beschreibt eine Mischung zwischen cloudbasierter Zugriffsverwaltung und Schutz von On-Premise-Systemen. Die Zugriffsverwaltung und die Identitäten werden in der Cloud geschützt. Die internen Systeme werden durch Firewall-Systeme geschützt.

Kompetenzcheck

1. Beschreiben Sie, wie der Einsatz von Clustern die Verfügbarkeit eines Systems erhöhen kann.
2. Geben Sie jeweils ein Beispiel vertikaler und horizontaler Skalierung an.
3. Geben Sie mindestens zwei Kriterien für eine gute Administrierbarkeit an.
4. Nennen Sie mindestens zwei Kostenarten, die bei der Wirtschaftlichkeit betrachtet werden.
5. Erklären Sie in eigenen Worten den Begriff der Perimetersicherheit.
6. Bearbeiten Sie die Aufgabe 1 der Lernsituation 2 im Arbeitsbuch.

A1

1.2.2 Virtuelle Server planen

S **Sie sollen einen virtuellen Serverdienst in Betrieb nehmen. Dazu ist es nötig, zunächst die Planung des Serverdienstes vorzunehmen.**

(1) Definition und Einsatz virtueller Maschinen

Die Verwendung von Virtualisierung gehört heutzutage fast immer zum Betrieb eines Computernetzes. Dazu können einzelne Server oder ganze Serverlandschaften über die für Endanwender zur Verfügung stehenden Desktopsysteme virtualisiert werden (siehe auch Jahrgangsband 1, Lernfeld 2, Kap. 2.4.1, 2.7.3 und Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.3).

Virtuelle Maschine (VM)

Eine virtuelle Maschine ist ein virtuelles Computersystem, das dieselben Funktionen bietet wie ein physisches System. Auf einem physischen System können mit einer Software (in diesem Kontext wird diese Software „Hypervisor" genannt) eine oder auch mehrere virtuelle Maschinen konfiguriert und gestartet werden. Bei der Konfiguration einer virtuellen Maschine wird dieser beispielsweise CPU-Leistung, Arbeitsspeicher und Festplattenspeicher zugewiesen. Außerdem können virtuellen Maschinen auch eine oder mehrere virtuelle Netzwerkkarten zur Verfügung gestellt werden.

Bei der Virtualisierung mit virtuellen Maschinen wird zwischen Typ-1- und Typ-2-Hypervisor unterschieden. Bei einem Typ-1-Hypervisor findet die Virtualisierung direkt auf der Hardware statt. Ein solches System ist auf die Arbeit mit virtuellen Maschinen spezialisiert und es ist nicht möglich, andere Software, z. B. ein Office-Programm, auf einem Typ-1-Hypervisor zu verwenden. In produktiven Umgebungen im Unternehmensumfeld kommt vor allem diese Art des Hypervisors zum Einsatz, da sie im Regelfall performanter ist und der Umgang mit vielen virtuellen Maschinen durch entsprechende Managementwerkzeuge deutlich einfacher ist. Ein Typ-2-Hypervisor ist ein Programm, das auf einem Betriebssystem gestartet wird. Innerhalb dieses Programms können eine oder mehrere virtuelle Maschinen gestartet werden. Neben dem Hypervisor können auf dem Betriebssystem noch weitere Programme gestartet werden.

(2) Vorteile des Einsatzes virtueller Maschinen

Die Verwendung von virtuellen Maschinen im Gegensatz zu physischen Servern bietet eine große Anzahl an Vorteilen.

- Ein physisches System benötigt Stellfläche. Wenn hingegen mehrere virtuelle Maschinen auf einem System laufen, spart das Platz ein. Das ist gerade im Rechenzentrum bei sehr vielen Systemen ein entscheidender Vorteil.
- Jedes physisches System benötigt Energie. Laufen mehrere virtuelle Maschinen auf einem System, wird ein deutliches Maß an Energie gespart. Auch das ist gerade im Rechenzentrum wichtig.
- Jedes System erzeugt Abwärme. Auch hier gilt wieder, wenn mehrere virtuelle Maschinen auf einem System laufen, wird weniger Abwärme erzeugt. Die Kühlung der Systeme und die damit verbundene Abfuhr der Wärme ist Kernthema beim Betreiben von Serverräumen oder eines Rechenzentrums.
- Virtuelle Maschinen sind leicht kopierbar. Eine virtuelle Maschine kann als Vorlage (Template) angelegt werden und mit diesem vervielfältigen, sodass auf diese Art eine Vielzahl gleichartiger virtueller Maschinen in kurzer Zeit erstellt werden können.
- Eine virtuelle Maschine lässt sich leicht von einem Hypervisor zu einem anderen Hypervisor verschieben. Dadurch kann im Bedarfsfall eine Migration auf einen neuen physischen Server erfolgen.
- Eine virtuelle Maschine besteht aus wenigen Dateien, z. B. einer Konfigurationsdatei und einer Datei, die den Datenspeicher der virtuellen Maschine repräsentiert. Durch das Sichern dieser Dateien wird eine Sicherung der virtuellen Maschine inklusive des Betriebssystems und der auf der virtuellen Maschine enthaltenen Nutzdaten durchgeführt.

- Neben dem herkömmlichen Backup ist auch ein Snapshot einer virtuellen Maschine möglich. Bei einem Snapshot wird ein Abbild der virtuellen Maschine zur Laufzeit erstellt (Momentaufnahme). Alle geöffneten Programme und geladenen Daten lassen sich auf diese Weise schnell wiederherstellen.
- Eine virtuelle Maschine verfügt über eine gute Skalierbarkeit, da einer einzelnen virtuellen Maschine im Hypervisor mehr Ressourcen wie CPU-Leistung oder Arbeitsspeicher zur Verfügung gestellt werden können (vertikale Skalierung).
- Ein Dienst kann über mehrere virtuelle Maschinen verteilt werden. Dadurch erhöht sich dessen Verfügbarkeit und eine Lastverteilung über die virtuellen Maschinen ist möglich (horizontale Skalierbarkeit).
- Innerhalb einer virtuellen Maschine kann Software isoliert betrieben werden. Dadurch ist es beispielsweise einfach möglich, eine Testumgebung aufzubauen. Einzelne virtuelle Maschinen können dazu auch ein virtuelles Netzwerk bilden.
- Mit einer virtuellen Maschine kann ältere, mit neuen Betriebssystemen nicht mehr kompatible, Software ressourcenschonend betrieben werden.
- Ein einzelner Dienst auf einem physischen System lastet das System meistens nicht stark aus. Dadurch werden Ressourcen des Systems nicht verwendet und liegen brach. Einer virtuellen Maschine können zielgerichtet Ressourcen zugewiesen werden, damit eine optimale Ausnutzung gegeben ist. Die anderen Ressourcen des physischen Systems können anderen virtuellen Maschinen zugeteilt werden.

(3) Varianten der Virtualisierung

Wie oben beschrieben gibt es zwei Varianten der Virtualisierung mit virtuellen Maschinen. Die unterschiedlichen Hypervisor werden bei verschiedenen Anwendungszwecken eingesetzt.

Typ-1-Hypervisor
Ein Typ-1-Hypervisor hat ausschließlich den Anwendungszweck, eine große Anzahl virtueller Maschinen zu hosten. Dieser Typ von Hypervisor arbeitet nah an der Hardware und ist dadurch performanter und effizienter als ein Typ-2-Hypervisor. Für das Management der virtuellen Maschinen steht oft spezialisierte Software zur Verfügung. Durch die Hardwarenähe kann das Virtualisierungs-Flag (bei Intel „VTx" bzw. bei AMD „AMD-V") der physischen CPU für die Virtualisierung an die virtuelle Maschine durchgereicht werden. Dadurch ist es möglich, innerhalb einer virtuellen Maschine wiederum eine weitere zu starten. Virtuelle Maschinen können also einfacher ineinander verschachtelt werden.

Ein Typ-1-Hypervisor kann die laufenden virtuellen Maschinen überwachen und ggf. Ressourcen wie CPU-Leistung einzelnen virtuellen Maschinen automatisch zuweisen oder von diesen entfernen. Die Zuweisung oder das Abziehen von Ressourcen wird aufgrund von Regeln durchgeführt. Dieser Vorgang wird dynamisches Ressourcenmanagement genannt.

Typ-2-Hypervisor
Ein Typ-2-Hypervisor (Hosted Hypervisor) arbeitet auf einem Betriebssystem als normales Anwendungsprogramm. Es ist also möglich, neben dem Hypervisor noch weitere Programme zu starten. Das macht das Gesamtsystem anfälliger für Abstürze, denn auch ein anderes Programm neben dem Hypervisor kann zu einem Systemabsturz führen.

Diese Art von Hypervisor ist meist einfach zu installieren und zu konfigurieren. Dabei ist keine besondere Hardware für die Virtualisierung nötig. Das Betriebssystem stellt bei diesem Hypervisor die Systemressourcen. Virtuelle Maschinen mit einem Typ-2-Hypervisor lassen sich außerdem gut zum Ausprobieren von Software verwenden. Beispielsweise kann damit ein neues Betriebssystem kennengelernt werden. Auch für Softwareinsellösungen, wenn eine Software nur noch an wenigen Stellen im Betrieb eingesetzt wird und nicht mehr mit dem verwendeten Betriebssystems zusammenarbeitet, kann ein Typ-2-Hypervisor eingesetzt werden.

(4) Migration eines bestehenden physischen Systems in eine virtuelle Maschine

Soll ein Server von einem physischen System auf eine virtuelle Maschine migriert werden, ist es oft sinnvoll, das System komplett neu zu installieren. Durch die Neuinstallation werden überflüssige Daten nicht übernommen. Außerdem kann die Konfiguration im Zuge der Neuinstallation grundlegend überprüft werden.

Kommt eine Neuinstallation des Servers nicht infrage, kann der physische Server in eine neue virtuelle Maschine überführt werden. Wichtig ist es, vor der Migration ein Backup des zu überführenden Systems zu erstellen.

Eine virtuelle Maschine besteht im Hypervisor auf dem Wirtssystem aus wenigen Dateien. Meist liegen eine Konfigurationsdatei und eine Datei, die das Laufwerk der virtuellen Maschine repräsentiert, vor.

Bei der Softwarelösung „VMware" sind in der Minimalkonfiguration eine vmx-Datei und zwei vmdk-Dateien vorhanden. Die vmx-Datei enthält die Konfiguration der virtuellen Maschine. Zur Konfiguration gehört beispielsweise, wie viel Arbeitsspeicher und CPU-Leistung der virtuellen Maschine zur Verfügung stehen. Die vmdk-Dateien repräsentieren die virtuellen Datenträger und enthalten alle Daten der virtuellen Maschine, z. B. das Betriebssystem. Die „-flat.vmdk"-Datei beinhaltet dabei die virtuellen Datenträger. Die andere Datei enthält die Beschreibung der Datenträger.

Bei der Virtualiserungssoftware „VirtualBox" tragen die Dateien die Endung „vbox" und „vdi". Die vbox-Datei beinhaltet die Konfiguration der virtuellen Maschine. Die Konfiguration liegt XML-codiert vor. Das hat den Vorteil, dass die Konfigurationsdaten auch für Menschen lesbar sind.

Beispiel: Auszug aus der Konfigurationsdatei einer VM von VirtualBox

```
<Network>
 <Adapter slot="0" enabled="true" MACAddress="080027628973" cable="true" type="Am79C973"
 <InternalNetwork name="intnet-LF 10b"/>
 </Adapter>
</Network>
```

Durch diese Konfiguration wird der virtuellen Maschine eine Netzwerkkarte mit dem AMD-Chipsatz Am79C973 zur Verfügung gestellt. Diese Netzwerkkarte ist aktiv (enabled="true") und angeschlossen (cable="true") und hat die MAC-Adresse 08:00:27:62:89:73. Die Karte hat nur eine Verbindung in ein virtuelles Netzwerk mit dem Namen „intnet-LF 10b". Die vdi-Datei beinhaltet den virtuellen Datenträger.

Ziel einer Migration ist es, ein Abbild der physischen Speichermedien in die entsprechende Datei der virtuellen Maschine zu kopieren. Dazu stehen je nach Virtualisierungssoftware spezielle Werkzeuge bereit. Beispielsweise steht die Software „VMware vCenter Converter" zur Verfügung. Nach der Installation der Software lassen sich die Partitionen des physischen Systems auf einen externen Datenträger als vmdk-Datei abspeichern.

Unter Linux ist es mit Boardmitteln möglich, eine Partition in eine Image-Datei zu kopieren. Diese Image-Datei kann dann in eine Virtualisierungssoftware eingebunden werden. Mit dem Befehl „dd if=/dev/sda1 of=/path/to/images/vm.img" (dd = disk dump) wird die Partition „sda" in das Image „vm.img" kopiert. Diese Kopie wird bit- oder byte-weise durchgeführt und ist unabhängig vom eingesetzten Betriebssystem.

(5) Varianten zum Sichern einer virtuellen Maschine

Ein Vorteil von virtuellen Maschinen sind die verschiedenen Möglichkeiten der Sicherung.

Backup-Agenten und agentenloses Backup

Wie bei physischen Maschinen ist es möglich, einen Backup-Agenten auf einer virtuellen Maschine zu installieren. Durch den Einsatz dieser Technik kann ein bereits für physische Systeme angeschafftes System weiter verwendet werden. Somit können das erworbene Know-how weiter genutzt und die Kosten für eine Neuanschaffung vermieden werden. Durch den Agenten lässt sich eine vollständige Sicherung des Systems vornehmen oder nur ein bestimmter Datenbestand sichern, z. B. die Nutzdaten. Wenn nur mit wenigen virtuellen Maschinen gearbeitet wird, ist eine Backup-Lösung mit Agenten sinnvoll.

Bei einem agentenlosen Backup wird auf der virtuellen Maschine keine Backup-Software installiert. Die Sicherungssoftware wird auf der Ebene des Hypervisors ausgeführt. Häufig werden bei virtuellen Maschinen sogenannte Snapshots gespeichert. Ein Snapshot ist ein Abbild des Zustands und der Daten einer virtuellen Maschine zu einem bestimmten Zeitpunkt. Die Snapshots werden auf dem Backup-Server oder dem Backup-Medium abgelegt. Durch den Einsatz eines agentenlosen Backups lassen sich die Kosten für die Lizenzen einsparen. Außerdem kann losgelöst vom Betriebssystem der virtuellen Maschine gearbeitet werden. Wenn nur die Nutzdaten eines Systems gesichert werden müssen, ist das Sichern von Snapshots nicht effizient.

Kompetenzcheck

1. Geben Sie an, welche Aussagen richtig sind.
   a) Virtuelle Maschinen benötigen mehr Platz im Rechenzentrum.
   b) Durch den Einsatz von virtuellen Maschinen kann der Energiebedarf reduziert werden.
   c) Virtuelle Maschinen können nur mit Aufwand zwischen Hypervisoren verschoben werden.
   d) Virtuelle Maschinen können durch Backups oder Snapshots gesichert werden.
   e) Physische Server lassen sich leichter vertikal skalieren als virtuelle Maschinen.
   f) VirtualBox ist ein Beispiel für einen Typ-1-Hypervisor.
   g) VirtualBox legt für die Beschreibung einer virtuellen Maschine eine XML-Datei an.
   h) Backups von virtuellen Maschinen können mit oder ohne Agenten erfolgen.

2. Bearbeiten Sie die Aufgaben 2 und 3 der Lernsituation 2 im Arbeitsbuch. A2 A3

1.2.3 Container planen

S **Sie sollen sich mit dem Konzept der Containerisierung auseinandersetzen und sich einen Überblick über marktübliche Containersysteme verschaffen.**

(1) Definition von Container

Um containerisierte Virtualisierung besser zu verstehen, ist es sinnvoll, die Funktion Change Root („chroot") in einem Linux-Betriebssystem zu betrachten, die eine Grundlage in der Containerisierung bildet. Im Gegensatz zu Windows, bei dem die Verzeichnisstruktur in der Regel mit Laufwerksbuchstaben gekennzeichnet wird und bei dem nur wenige Ordner auf der obersten Ebene vom Betriebssystem vorgegeben werden, haben Unix-/Linux-basierte Systeme einen standardisierten Verzeichnisbaum, der von der Linux Foundation als „Filesystem Hierarchy Standard" spezifiziert wird (https://refspecs.linuxfoundation.org/fhs).

Pfadseparator in dieser Verzeichnisstruktur ist der Schrägstrich (/), im Gegensatz zum umgekehrten Schrägstrich (\) im Windows-Verzeichnisbaum. Der oberste Schrägstrich wird als „Root" des Verzeichnisbaums bezeichnet. Dies sollte nicht mit dem Administratorkonto unter Linux verwechselt werden, welches auch „root" genannt wird. Im folgenden Beispiel werden einige standardisierte Verzeichnisse unter „Root" erklärt:

```
/
|
+ bin
+ dev
+ etc
+ home
+ lib
+ usr
+ var
```

Verzeichnisstruktur des Linux-Dateisystems	
Verzeichnis	**Beschreibung**
/bin	Hier werden essenzielle Binärdateien abgelegt, die im ersten Bootmodus des Systems (Einzelnutzermodus) schon verfügbar sein müssen. Dazu gehören Dateien, um das System weiter zu starten (Mehrbenutzermodus etc.) oder auch zu reparieren.
/dev	Alle Hardwarekomponenten eines Unix-/Linux-Systems werden über sogenannte Device-Dateien abgebildet, die unter diesem Verzeichnis erstellt werden. Ein Zugriff auf eine Partition einer Festplatte kann beispielsweise über „/dev/sda/sda1" stattfinden.
/etc	Hier liegen alle Konfigurationsdateien des Betriebssystems und aller installierter Software als textbasierte Dateien. Das Betriebssystem legt die Dateien oft auf der obersten Ebene ab. Installierte Software erzeugt Unterverzeichnisse und legt die Konfiguration dort ab. Benutzernamen und Passwörter sind beispielsweise in der Datei „passwd" gespeichert. Ein installierter Apache-Webserver erzeugt z. B. ein Unterverzeichnis „/etc/apache2" und legt dort seine Konfiguration ab.
/home	Dies ist das Heimverzeichnis für angelegte Benutzer, die das System nutzen. Ein Benutzer mit dem Benutzernamen „westermann" erhält unter „/home/westermann" ein Verzeichnis mit exklusiven Zugriffsrechten, unter dem er seine persönlichen Daten ablegen kann. Installierte Programme wie „LibreOffice" speichern Dokumente des Nutzers in dem Bereich.
/lib	Hier liegen die essenziellen Bibliotheken, die für den Start des Systems und der Ausführung von Binärdateien aus „/bin" und „/sbin" nötig sind.
/usr	Das ist ein Unterbereich, der nur lesbare Dateien enthält, die über mehrere Hostsysteme geteilt werden können. Hier gibt es eine eigene Unterstruktur mit Verzeichnissen „/usr/bin", „/usr/lib" etc.
/var	Variable Dateien, also sich verändernde Dateien, z. B. log-Dateien, werden hier abgelegt. Die Struktur ist ähnlich der unter „/etc", also die log-Dateien für den Apache-Webserver findet man unter „/var/log/apache2" und die log-Dateien vom System direkt unter „/var/log/", z. B. die Datei „syslog", in der das System Log-Einträge vornimmt.

Der Verzeichnisbaum ist einmalig in einem Unix-/Linux-System. Das bedeutet, zusätzlicher Speicher durch neue Festplatten oder SSDs muss einem Ordner im bestehenden Verzeichnisbaum zugeordnet werden. Es kann kein paralleler Baum wie unter Windows mit neuem Laufwerksbuchstaben erstellt werden. Eine typische Konfiguration ist, eine SSD unter dem **Root-Verzeichnis** einzubinden. Der Vorgang wird „mounten" genannt und so heißt auch das Kommando (mount), mit dem die eingebundenen Speichersysteme angezeigt oder verändert werden können. Wird weiterer Speicher benötigt, kann beipsielsweise das Verzeichnis „/home" auf ein anderes Speichermedium „gemounted" werden. Das Root-Verzeichnis und alle Unterverzeichnisse außer „/home" sind dann beispielsweise auf der SSD-1 und alle Unterverzeichnisse unter „/home" auf der SSD-2 abgelegt.

Die Information über den einheitlichen Verzeichnisbaum und über das Einbinden von weiterem Speicher in diesen zeigt, dass hierüber die Zugriffsrechte auf den Verzeichnisbaum in isolierten Schichten für darin laufende Prozesse geschaffen werden können. Genau hier setzt das Kommando „chroot" an. Die

folgende Abbildung zeigt in dem Verzeichnisbaum die hinzugefügten Ordner „/chroot/jail_1“ und „/chroot/jail_2“:

```
/
|
+ bin
+ chroot
| + jail_1
| | + bin
| | + lib
| + jail_2
| | + bin
| | + lib
+ dev
+ etc
+ lib
+ sbin
+ usr
```

In den Unterverzeichnissen sind im Beispiel nur „bin“ und „lib“ aufgeführt. In der Regel ist es aber ein völlig eigener Verzeichnisbaum. Führt man jetzt das Kommando „chroot /chroot/jail_1/“ aus, sieht der Prozess, aus dem „chroot“ ausgeführt wurde, nur noch das Verzeichnis ab „/chroot/jail_1“ abwärts als sein Dateisystem. Ein Zugriff auf „/etc“ oder „/dev“ ist nicht mehr möglich. Dadurch können Prozesse auf dem System laufen, die eine isolierte Umgebung erhalten. Diese Umgebung kann mit anderen Bibliotheken versehen werden und dadurch kann ein anderes System installiert und zur Ausführung gebracht werden. Zur Abgrenzung der Verzeichnisstruktur wurden noch Namensräume hinzugefügt, die eine weitere Isolierung eines laufenden Prozesses, z. B. durch Control Groups (cgroups), ermöglichen.

Linux Containers (LXC) und Linux Container Daemon (LXD)

Aus den Namensräumen entwickelte sich das Linux Container Projekt LXC, das Bestandteil des Linux Kernels seit Version 2.6.29 ist, der 2009 veröffentlicht wurde. Ab Kernelversion 3.12 (veröffentlicht 2013) können Kernel-Namensräume neben dem Prozess auch für Netzwerk- und Mount-Punkte verwendet werden. Für LXC-Version 1.0 konnten auf diese Weise erhebliche Sicherheitsprobleme behoben werden. LXC ist eine Lösung, um eine virtuelle Umgebung für Software zur Verfügung zu stellen, in der eine Anwendung isoliert vom Gast-Betriebssystem (Host-OS) ausgeführt werden kann.
Eine Weiterentwicklung von LXC wurde durch die Ubuntu-Firma Canonical vorgenommen und ergab LXD (Linux Container Daemon). LXD stellt zusätzliche Tools und Konzepte zur Verfügung, die eine einfachere Nutzung der LXC-basierten Container erlauben. Die Einführung sogenannter Container-Images ist ein Beispiel dafür (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.3).

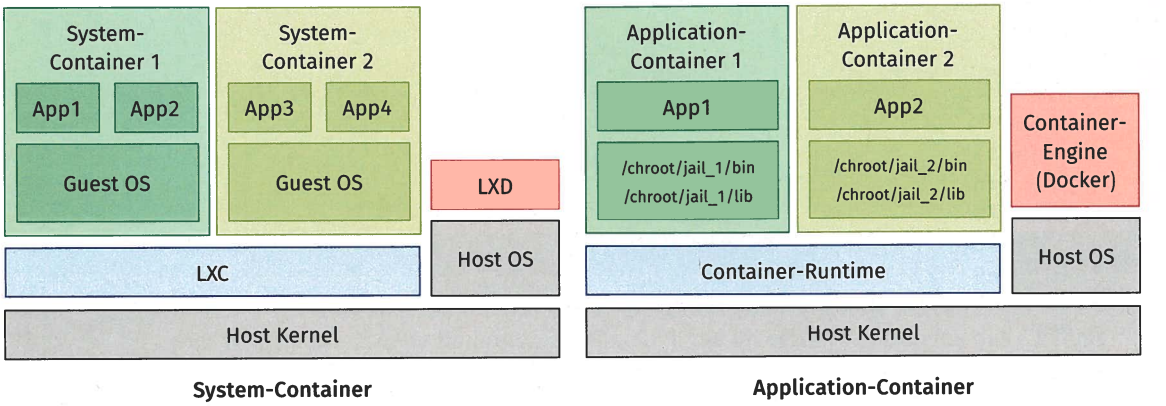
(2) System-Container vs. Application-Container

LXC benutzt sogenannte System-Container, während z. B. die Container-Implementierung „Docker“ Application-Container benutzt.

System-Container sind mit virtuellen Maschinen vergleichbar. Die Virtualisierung findet auf der Betriebssystemebene statt und erlaubt mehrere verschiedene linuxbasierte Virtual Environments (VE) isoliert auf einem Hostsystem zu betreiben. In solch einer VE können **ein oder mehrere Anwendungen** zur Ausführung kommen.

Demgegenüber stehen **Application-Container**, wie sie beispielsweise von „Docker“ eingesetzt werden. Auf dem Hostkernel läuft die Container-Runtime „runc“. Darauf läuft dann ein Anwendungs-Container, der die entsprechenden Binär- und Bibliotheksdateien enthält und in dem **eine Anwendung** ausgeführt

wird. Auf dem Betriebssystem des Systems wird eine Container-Engine wie „Docker“ ausgeführt, die für das Management der Virtualisierung zuständig ist.



System- und Application-Container im Vergleich

Abgrenzung zwischen Virtual Environment (VE) und Virtual Machine (VM)

In der Virtual Environment findet keine Hardwareemulation statt, die beispielsweise in einer VM durch den Hypervisor durchgeführt wird. Die VM enthält das komplette Betriebssystem inklusive eigenem Kernel, Festplatten-/SSD-Speicher, virtuelle Prozessoren und Netzwerkkarten. Die VE hingegen nutzt den Kernel des Hostsystems. Ein System-Container wird auf der Betriebssystemebene virtualisiert, während eine VM auf der Hardwareebene virtualisiert wird. Der geringere Ressourcenverbrauch einer VE hat aber auch ihren Preis, denn die Container sind enger mit dem Hostsystem verbunden, was sicherheitstechnisch mehr Angriffsmöglichkeiten bietet.

(3) Imperative und deklarative Beschreibung von Infrastruktur

Bei den meisten Softwareprodukten wird die Anwendung mit allen benötigten Abhängigkeiten in einer Installer-Datei ausgeliefert. Diese Installer-Datei ist für ein System mit einem bestimmten Betriebssystem gedacht. Entspricht das System nicht den Grundanforderungen, muss es erst durch Aktualisierungen auf den geforderten Stand gebracht werden. Diese Vorgehensweise nennt sich **imperative Konfiguration**. Der Nachteil ist, dass es häufig schwer ist, nach einer Installation das System wieder in seinen vorherigen Zustand zurückzuversetzen, wenn die Installation, aus welchen Gründen auch immer, wieder entfernt werden soll.

Demgegenüber steht die **deklarative Konfiguration**. Mit den Containern hält eine deklarative Beschreibung von Systemen Einzug. Mit ihr wird eine Beschreibung – meist in Form einer Textdatei – verfasst, in der dargestellt wird, wie das System aussehen soll. Ein Beispiel dafür sind Beschreibungen mit Ansible, Vagrant oder Docker-Container, mit denen erläutert wird, was in dem System vorhanden sein soll. Dabei gehen die Beschreibungen konkret auf die zu verwendenden Softwareversionen ein. Wenn mit Vagrant beispielsweise eine virtuelle Maschine erstellt werden soll, wird darin festgelegt, dass sie z. B. Ubuntu mit einer konkreten Version enthalten soll (siehe auch Lernfeld 10b, Kapitel 1.5.2 und Lernfeld 12b, Kapitel 5.4).

Vorteile durch Einsatz von Containern
In der Softwareentwicklung werden Funktionen, die von mehreren Programmen genutzt werden können, in Bibliotheken ausgelagert. Die ursprüngliche Idee war, dass die Bibliothek nur einmal auf einem System installiert wird und dann mehrere Programme diese nutzen. Mit der Weiterentwicklung der Bibliotheken entstanden aber verschiedene Versionen, die auch unterschiedliche Funktionen beinhalteten. Das führte dazu, dass mit der Softwareinstallation auch meistens die abhängigen Bibliotheken mit installiert werden, was zu einer umfangreicheren Installation führt.

Container bieten hier die Möglichkeit, ein System nur mit den nötigen Abhängigkeiten zu installieren. Dadurch verschwinden auch Probleme, die beispielsweise mit der Installation verschiedener Softwareversionen auf einem System auftreten. Mit der deklarativen Beschreibung wird festgelegt, wie das System aussehen soll. Somit wird bei jedem Start des Containers ein definierter Ausgangszustand des Systems erreicht. Softwareentwicklung, Testung und das Ausrollen erfolgt mit dieser Beschreibung und ermöglicht zu jedem Start, immer einen gleichen Zustand des Systems zu erhalten.

(4) Docker

Ziel von Docker ist es, für Anwendungsentwickler die Nutzung von Containern zu vereinfachen. Dazu stellt Docker eine Schnittstelle zur Verfügung, um Container einfach erstellen, starten und ausrollen zu können. Auch neu entwickelte Software lässt sich damit testen. Das System basiert auf dem Client-Server-Modell. Der Kommandozeilen-Client steuert den Server, auf dem die Container erstellt und ausgeführt werden. Dabei können sich Client und Server auf einem Computer befinden oder über das Netzwerk verteilt betrieben werden.

Kern eines Containers ist das sogenannte **Image**, ein TAR-Archiv, in dem alle Dateien enthalten sind, die auf dem Kernel zur Ausführung kommen. Images haben einen **Schichtenaufbau**. Man kann mit einem leeren Container anfangen. Zur Erinnerung: Der Container wird auf dem Kernel ausgeführt, hat also die Kernelressourcen zur Verfügung. In der Regel soll aber ein Container ein Minimum der Ressourcen des Linux-Systems nutzen. In der Container-Nutzung wird häufig die Linux-Distribution „Alpine" verwendet, da sich hiermit ein sehr kleines Minimalsystem bauen lässt. Wenn ein eigenes Image erstellt wird, sucht man sich ein bereits bestehendes Image als Grundlage aus und erweitert dieses. Dem zugrunde gelegten Image sollte man vertrauen, denn über diesen Weg können Funktionen ins System gelangen, die nicht gewünscht sind. Die Images werden standardmäßig aus dem Repository „Docker Hub" bezogen. Es können aber auch eigene Repositories verwendet werden.

Die allgemeine Vorgehensweise ist somit:
- „create image"
- „build image"
- „push image"
- „pull image"
- „run image"

Create-Prozess (Create Image)
Ein sehr einfaches Image kann nur aus zwei Zeilen bestehen. In einem leeren Ordner wird eine Datei **„Dockerfile"** erstellt.

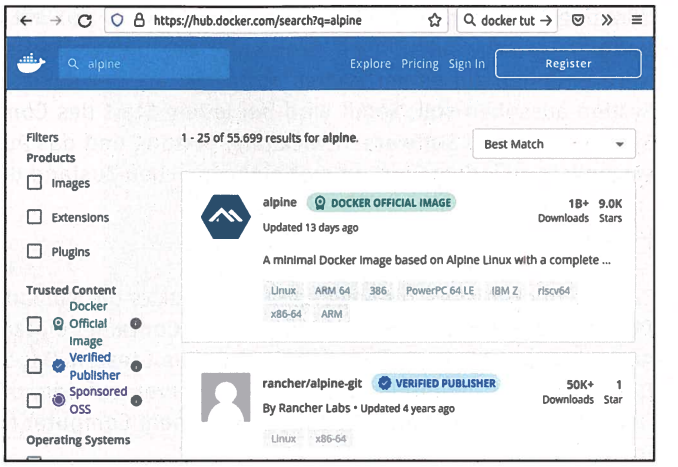
> ! Auf Linux-Systemen wird zwischen Groß- und Kleinschreibung von Dateinamen unterschieden, auf Windows-Systemen nicht.

Beispiel: Das Folgende ist der Inhalt des Dockerfiles, die deklarative Beschreibung des später auszuführenden Containers.

```
1 FROM alpine
2 CMD ["echo", "hello world!"]
```

Zeile	Erklärung
1	Mit der FROM-Anweisung wird angegeben, auf welchem Image das hier beschriebene aufbaut. Dabei greift Docker auf das Docker Hub zu, eine Webseite mit vorgefertigten Images.
2	Hier wird auf der Kommandozeile ein Befehl ausgegeben.

Die nebenstehende Abbildung zeigt das Suchergebnis auf der Webseite des Docker Hub (hub.docker.com). Erkennbar ist, dass durch Zertifikate versucht wird, qualitativ sichere Images von anderen Images zu unterscheiden. Auf Docker Hub kann sich jeder registrieren und eigene Images veröffentlichen.



Weboberfläche Docker Hub

Build-Prozess (Build Image)

Nach der Beschreibung des Images folgt der Build-Prozess. Über das Kommando **„docker build“** wird er angestoßen.

```
 1 docker build .
 2 Sending build context to Docker daemon 2.048kB
 3 Step 1/2 : FROM alpine
 4 latest: Pulling from library/alpine
 5 530afca65e2e: Pull complete
 6 Digest: sha256:7580ece7963bfa863801466c0a488f11c86f85d9988051a9f9c68cb27f6b7872
 7 Status: Downloaded newer image for alpine:latest
 8  ---> d7d3d98c851f
 9 Step 2/2 : CMD ["echo", "hello world!"]
10  ---> Running in a70970653c56
11 Removing intermediate container a70970653c56
12  ---> cb2bb7afa3b2
13 Successfully built cb2bb7afa3b2
```

Zu sehen ist, dass der Prozess in zwei Stufen abläuft. In der ersten Stufe wird geschaut, ob in der lokalen Registry, in der Images abgelegt werden, das Image „alpine“ schon vorhanden ist. Da es nicht vorhanden war, wird es in Zeile 4 von Docker Hub heruntergeladen. Images werden mit einer eindeutigen Nummer gekennzeichnet, das Alpine-Image hat die Kennung „d7d3d98c851f“. In Zeile 10 wird jetzt der zweite Schritt durchgeführt. Dazu wird ein temporärer Container mit der ID „a70970653c56“ erzeugt. Er wird verwendet, um das endgültige Image mit der ID „cb2bb7afa3b2“ zu bauen. Schaut man sich nun das lokale Repository mit „docker image list“ an, sieht man die beiden folgenden Images:

```
docker image list
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
<none>              <none>    cb2bb7afa3b2   51 seconds ago   5.53MB
alpine              latest    d7d3d98c851f   2 weeks ago      5.53MB
```

Zu sehen ist das erstellte Image „cb2bb7afa3b2“ und das heruntergeladene Alpine-Image „d7d3d98c851f“. Bemerkenswert ist, dass das Image nur 5,53 MB groß ist.

Schritte „Push Image“ und „Pull Image“

Mit den beiden Schritte „Push Image“ und „Pull Image“ ist es möglich, das Image z. B. auf Docker Hub oder einem anderen entfernten Repository hoch- bzw. herunterzuladen und dann zu starten.

Run-Prozess (Run Image)

```
docker run --name MyImage cb2bb7afa3b2
hello world!
```

Für den Aufruf wird ein Name vergeben und der Container wird ausgeführt. Im Anschluss wird auf der Kommandozeile der Echo-Befehl ausführt, wodurch der Text „hello world!“ ausgegeben wird.

Open Container Initiative (OCI)

Um eine Austauschbarkeit von Container-Images zu erreichen, wurde durch die Open Container Initiative (OCI) ein standardisiertes Format für die Image-Dateien beschrieben. Damit ist ein Austausch unter verschiedenen Container-Systemen möglich.

Ein Nachteil der einfachen Implementierung ist die Skalierbarkeit. Sollen mehrere Container auf verschiedenen Systemen ausgeführt werden, ist das zwar prinzipiell mit der Kommandozeile möglich, aber umständlich. Um Docker im Cluster zu verwenden, gibt es die Anwendung **„Docker Swarm“**, die sich jedoch nicht durchgesetzt hat. Weitverbreitet ist stattdessen die Software **„Kubernetes“**, eine Container-Orchestrierungsplattform.

Sicherungsmöglichkeiten für Container

Mit dem Paradigmenwechsel von imperativer zu deklarativer Konfiguration von Containern und der Tatsache, dass Container nicht ihren Status speichern, ergibt sich hinsichtlich der Datensicherung eine andere Herangehensweise als für virtuelle Maschinen.

Der Status eines Containers setzt sich zusammen aus:
- Grundimage, auf dem der Container beruht, z. B. Alpine-Linux mit bestimmter Version,
- deklarativer Beschreibung des Containers und
- enthaltener Anwendung mit Daten.

Um den Status zu speichern, sind also diese drei Information nötig. Die zu speichernden Daten sind anwendungsspezifisch und können in einer Datenbank oder einem Dateispeicher abgelegt sein. Hierzu müssen anwendungsspezifische Sicherungsmethoden implentiert werden.

Kompetenzcheck

1. Erklären Sie in eigenen Worten den Vorgang „mounten“ in Zusammenhang mit einem Linux-Dateisystem.
2. Beschreiben Sie, welche Änderung sich für den aufrufenden Prozess durch den Befehl „chroot/chroot/jail“ ergibt.
3. Erklären Sie den Unterschied zwischen einem Anwendungs- und einem System-Container.
4. Erklären Sie die imperative und die deklarative Konfiguration.
5. Beschreiben Sie, was mit Schichtenaufbau eines Container-Images gemeint ist.
6. Beschreiben Sie, wie die Sicherung eines Containers sich von der einer virtuellen Maschine unterscheidet.
7. Bearbeiten Sie die Aufgabe 4 der Lernsituation 2 im Arbeitsbuch. A4

1.3 Serverdienste und Plattformen planen

S **Sie planen den Einsatz von Serverdiensten und Plattformen gemäß den Kundenanforderungen.**

Die Installation eines neuen Servers beim Kunden gehört zum Aufgabenbereich, den die Mitarbeiter der JIKU IT-Solutions für ihre Kunden abdecken. Vor der Inbetriebnahme eines Servers müssen allgemeine und technische Anforderungen entsprechend den Anforderungen der Kunden geplant werden. Zu den Anforderungen gehören die Verteilung der IP-Adressbereiche auf DHCP-Server genauso wie die Übertragungsraten, die z. B. für ein Videokonferenzsystem bereitstehen müssen. Auch die Umsetzung von Sicherheit und Verfügbarkeit der Systeme darf bei der Planung nicht außer Acht gelassen werden. In diesem Kapitel wird exemplarisch die Planung für verschiedene Dienste und Anforderungen vorgestellt.

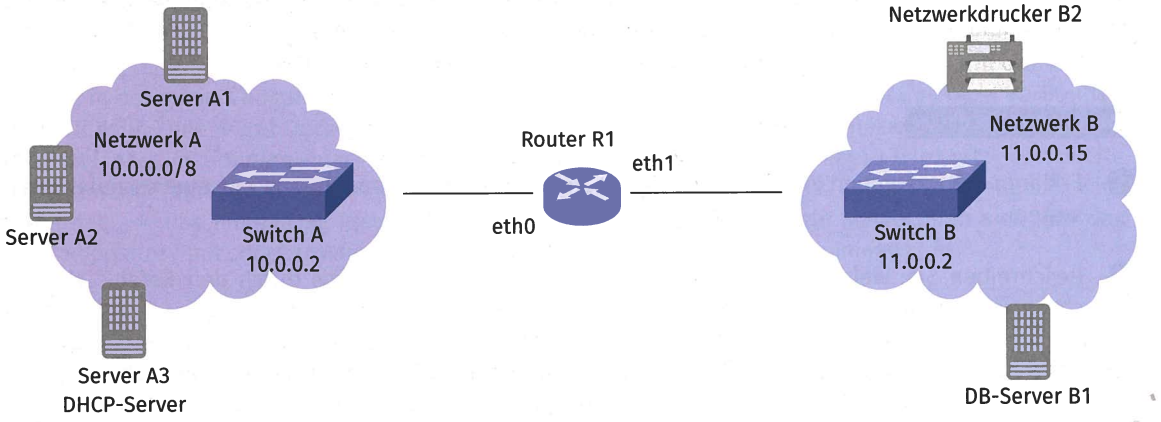
1.3.1 Serverdienste planen

S **Sie sollen verschiedene Serverdienste wie DHCP, Web- und Echtzeitdienste planen.**

(1) DHCP planen

In den meisten IP-Netzwerken ist ein DHCP-Server (Dynamic Host Control Protocol) installiert. Er ist für die Verteilung von IPv4-Adressen, Subnetzmasken, Adressen des Standard-Gateway und des DNS-Servers zuständig, kann aber auch zum Verteilen von IPv6-Adressen und vieler weiterer Parameter verwendet werden (vgl. Jahrgangsband 2, Lernfeld 9, Kap. 4.3.1).

Vor der Installation, der Konfiguration und dem Einsatz eines DHCP-Servers ist es sinnvoll, einige Konfigurationsparameter festzulegen. Zunächst muss natürlich der IP-Adressbereich bestimmt werden, für den der DHCP-Server zuständig ist. Dieser Adressbereich wird auch DHCP-Pool genannt.



Beispielnetzwerk mit DHCP-Server

Nicht alle Geräte im Netzwerk benötigen eine Adresszuweisung durch den DHCP-Server. Bei Koppelelementen und Servern werden die IP-Adressen in der Regel statisch vergeben. Außerdem gibt es Geräte, z. B. Netzwerkdrucker, die keine sich verändernde IP-Adresse beziehen sollen (also eine Adresse mit einer eingeschränkten Nutzungsdauer oder Leasetime), sondern immer die gleiche IP-Adresse vom DHCP-Server erhalten. Dies muss aber dokumentiert sein. Dazu ist es sinnvoll, einen IP-Adressvergabeplan zu erstellen, der alle Arten der Zuweisung enthält. In diesem Plan sollten alle verwendeten (Sub-)Netzadressen, Bereiche und die Ausnahmen beschrieben werden.

Beispiel:

IP-Adressvergabeplan	
Verwendete IP-Adressbereiche	
Netzwerk A	10.0.0.0/8
Netzwerk B	11.0.0.0/8
IP-Bereiche zur Vergabe	
Netzwerk A	10.0.0.100 - 10.0.0.200
Netzwerk B	11.0.0.100 - 11.0.0.200
manuell vergebene statische IP-Adressen	
Server A1 (MAC-Adresse 58:6c:25:2e:35:ce)	manuell vergeben, statisch: 10.0.0.22/8
Server A2 (MAC-Adresse 58:6c:25:66:67:a1)	manuell vergeben, statisch: 10.0.0.22/8
Server A3 (MAC-Adresse 58:6c:25:77:43:d9)/DHCP-Server	manuell vergeben, statisch: 10.0.0.10/8
Switch Netzwerk A	manuell vergeben, statisch: 10.0.0.2/8
Router Port eth0 (Netzwerk A)	manuell vergeben, statisch: 10.0.0.2/8
Server B1 (MAC-Adresse c7:12:d2:43:9f:00)/DB	manuell vergeben, statisch: 11.0.0.50/8
Switch Netzwerk B	manuell vergeben, statisch: 11.0.0.15/8
Router Port eth1 (Netzwerk B)/DHCP-Relay-Agent	manuell vergeben, statisch: 11.0.0.2/8
Fest zugewiesene IP-Adressen (DHCP)	
Netzwerkdrucker B2 (MAC-Adresse 2c:44:fd:75:12:dd)	statisch zugewiesen: 10.0.0.22/8

Durch einen IP-Adressvergabeplan ist dokumentiert, welche IP-Adressen und IP-Adressbereiche im Netzwerk verwendet werden. Vor der Installation müssen die weiteren Parameter bekannt sein. Meist werden zumindest die IP-Adressen des Standard-Gateways und des DNS-Servers verteilt.

Nachdem die grundlegenden Parameter vorliegen, muss definiert werden, welche Strategie bei einem Ausfall des DHCP-Servers verwendet wird. Es kann beispielsweise sein, dass der Einsatz zweier DHCP-Server sinnvoll ist und eine Aufteilung der Bereiche zwischen den Servern konfiguriert wird. Der Einsatz der DHCP-Relays muss ebenfalls festgelegt werden, im abgebildeten Netzwerk (S. 66) ist das die Schnittstelle „eth1" des Routers.

Nachdem die Struktur der Adressvergabe definiert ist, können noch weitere DHCP-Parameter festgelegt werden. Ein Parameter ist beispielsweise die Dauer einer Lease. Eine lange Lease-Dauer führt zu weniger DHCP-Verkehr und weniger Server-Auslastung. Systeme können die zugewiesene IP-Adresse lange verwenden. Bei einer kurzen Lease-Dauer hingegen werden die Adressen häufiger gewechselt. Dadurch ist die Adressvergabe aktueller und Konfigurationsänderungen können schnell bekannt gegeben werden.

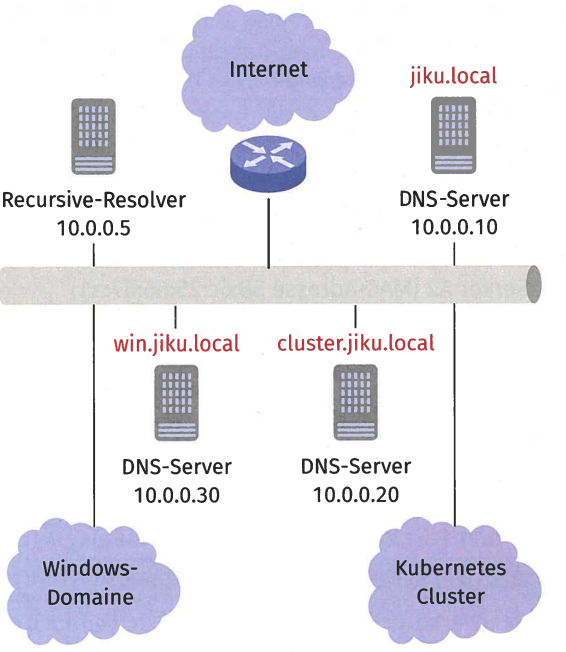
Vor dem Einsatz eines DHCP-Server sollte auch die Absicherung des Netzwerks gegenüber sogenannten Rogue-DHCP-Servern geplant werden. Ein Rogue-DHCP-Server ist ein feindlicher Server, der DHCP-Anfragen von Clients absichtlich mit gefälschten Informationen beantwortet, um beispielsweise Datenverkehr mitzuschneiden. Auf Switchen kann DHCP-Snooping aktiviert werden. Diese Funktion sorgt dafür, dass nur über zugelassene Switch-Ports DHCP-Anfragen beantwortet werden können (also eine DHCP-Offer-Nachricht gesendet werden kann).

Auch beim Backup-Konzept darf der DHCP-Server nicht ausgeschlossen werden. Es ist sinnvoll, die Lease-Daten in regelmäßigen Abständen zu sichern, um die Verfügbarkeit der Adressvergabe sicherzustellen.

(2) DNS planen

Für die Planung der Namensauflösung (vgl. Jahrgangsband 2, Lernfeld 9, Kap. 4.3.1) wird das folgende Szenario verwendet:

Die JIKU IT-Solutions GmbH hat für die Arbeitsplatzrechner und zugehörigen Server eine Windows-Domäne, die unten links in der nebenstehenden Abbildung gezeigt ist, eingerichtet. Die Namensauflösung in der Windows-Domäne wird vom DNS-Server des Domain Controllers übernommen, der die IP-Adresse „10.0.0.30" besitzt. Dieser ist der autoritative DNS-Server für die Unterdomäne „win.jiku.local". Die JIKU IT-Solutions hat ein Kubernetes-Cluster, das in der Unterdomäne „cluster.jiku.local" liegt. Dafür zuständig ist der autoritative DNS-Server mit der IP-Adresse „10.0.0.20". Der für „jiku.local" autoritative Namensserver „10.0.0.10" delegiert Anfragen über die Unterdomänen an die jeweilig autoritativen Server. In dem Szenario gibt es außerdem einen **Recursive-Resolver**, der Abfragen, die nicht von den drei Servern beantwortet werden können, in der DNS-Hierarchie des Internets durchführt.



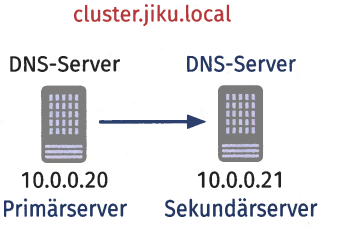
Ebenen der digitalen Vernetzung

Delegation

Ziel der Delegation ist es, Unterdomänen auf eigene autoritative DNS-Server zu verlagern. Dadurch kann die Administration aufgeteilt werden und zusätzlich die Last der DNS-Abfragen verteilt werden.

Ausfallsicherheit

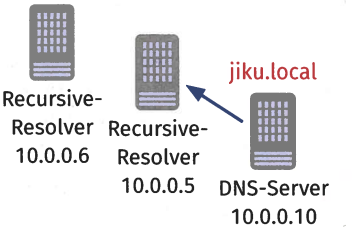
Zu jedem in der oberen Abbildung dargestellten Server, der als primärer Server dient, gibt es einen weiteren sekundären Server (exemplarisch rechts abgebildet). In der Konfiguration werden die beiden zusammengeschaltet und greifen auf die gleiche Zonendatei zu. Der sekundäre Server synchronisiert seine Konfiguration im festgelegten Intervall. Die nebenstehende Abbildung zeigt die beiden Server für die Unterdomäne „cluster.jiku.local".



Ausfallsicherheit für DNS-Server

Recursive-Resolver

Zentraler DNS-Server im Netzwerk der JIKU IT-Solutions ist der DNS-Server „jiku.local", der jedoch keine Abfragen selber durchführt, sondern diese nur weiterleitet. Erreichen ihn Abfragen für die Unterdomänen „win.jiku.local" oder „cluster.jiku.local", leitet er diese durch die konfigurierte Delegation an die entsprechenden DNS-Server weiter. Erhält er Anfragen für FQDN aus anderen Top-Level-Domänen, leitet er diese an den **Recursive-Resolver** weiter, der dann die Abfrage in der DNS-Hierarchie des Internets durchführt. Dazu ist die IP-Adresse „10.0.0.5" als **Forwarder** im DNS-Server „10.0.0.10" eingetragen. Zur Ausfallsicherheit werden auch wieder zwei Resolver betrieben und beide entsprechend konfiguriert.

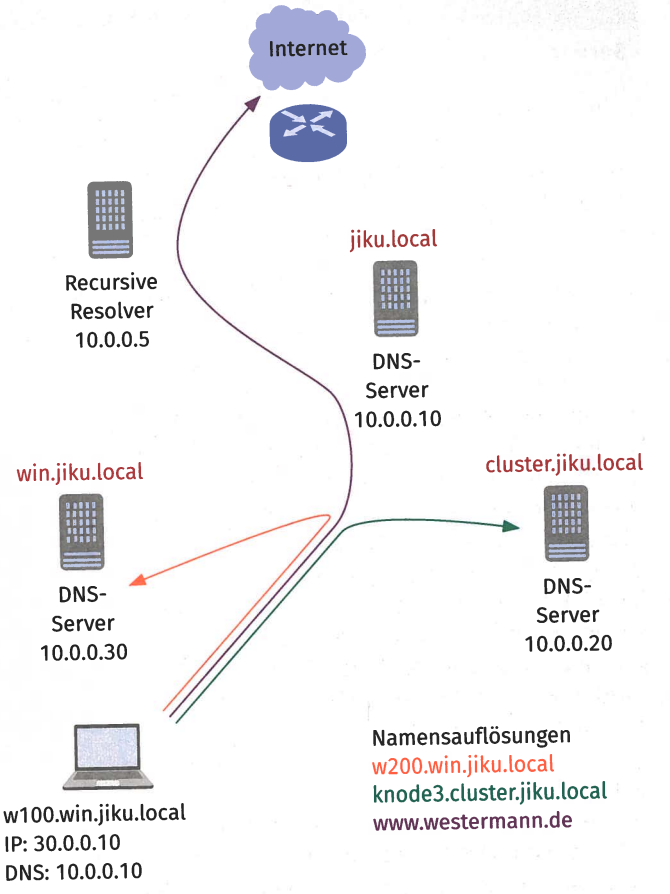



```
forwarders {
    10.0.0.5;
    10.0.0.6;
};
```

Recursive-Resolver-Anfrage

DNS-Anfrageszenarien aus Sicht eines Windows-Clients

Die nebenstehende Abbildung zeigt die Anfrage von drei FQDN und welche Server davon betroffen sind. Die Anfrage wird von einem Windows-Client in der Windows-Domäne gestartet. Für den Client ist der DNS-Server mit der IP-Adresse „10.0.0.10" konfiguriert. Dieser ist auch der autoritative DNS-Server für die Domäne „jiku.local".



DNS-Anfrage-Hierarchie aus Client-Sicht

Beschreibung der abgebildeten FQDN-Anfragen	
w200.win.jiku.local	Für die Unterdomäne „win.jiku.local" ist eine Delegation eingetragen, also leitet der Server die Anfrage an diesen weiter. Die Rückantwort kommt über ihn zurück und er sendet sie zu dem anfragenden Client „w100.win.jiku.local".
knode3.cluster.jiku.local	Für die Unterdomäne „cluster.jiku.local" ist eine Delegation eingetragen, also leitet der Server die Anfrage an diesen weiter. Die Rückantwort kommt über ihn zurück und er sendet sie zu dem anfragenden Client „w100.win.jiku.local".
www.westermann.de	Für die Toplevel-Domäne „.de" ist der DNS-Server nicht autoritativ, daher nutzt er seine Forwarder-Konfiguration und leitet die Anfragen an den Recursive-Resolver weiter. Dieser führt die Anfrage in der DNS-Hierarchie des Internets durch und sendet die Antwort zurück an den DNS-Server, der sie an den Client weiterleitet.

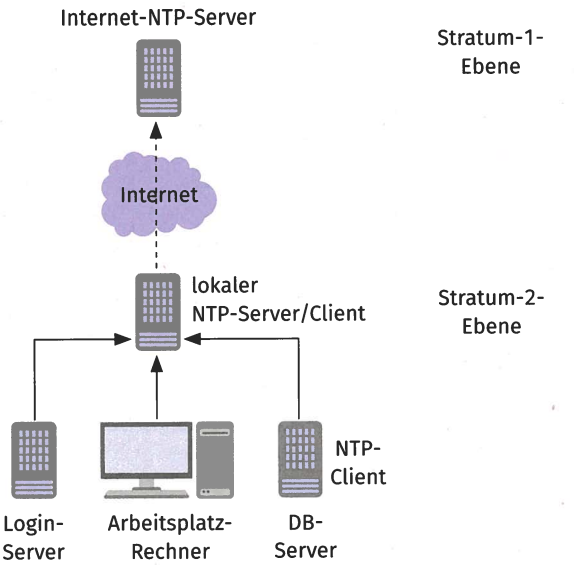
(3) NTP planen

Bei der Planung von IT-Systemen muss immer eine Bewertung der geplanten Systeme hinsichtlich einer genauen Zeitmessung erfolgen. Folgende Auswahl von Servertypen benötigen exakte Zeitmessungen.

Serverdienste mit Bedarf für genaue Zeitmessung	
Server	**Begründung**
Datenbank	zur Synchronisation von aktualisierten Datensätze in DB-Clustern
Single-Sign-On	Anmeldevorgänge oder zeitbasierte Zugriffsrechte
log-Dateien	zur Erfassung von Ereignissen
Kryptografische Protokolle	zur Berechnung von Chiffren, Prüfsummen oder Zertifikaten
Backup	zur präzisen Steuerung von Backups bzw. Restores
Cache/Proxy	zur Ermittlung der aktuellsten Daten für die Zwischenspeicherung

Die internen Computeruhren sind für solche Systeme nicht genau genug bzw. laufen in der Regel nicht synchron, da sie zu unterschiedlichen Zeitpunkten und meist ungenau konfiguriert werden.

Durch den Einsatz einer NTP-Infrastruktur kann auf jedem System die Uhrzeit präzise konfiguriert werden. Dabei muss ein NTP-Server sowohl als Server (Anfragen aus lokalem Netz) als auch als Client (zum Internet-NTP-Server) eingeplant werden. Entsprechende Firewall-Regeln und Routing-Einstellungen sind vorzusehen. Handelt es sich um sehr zeitkritische Systeme, muss ggf. ein zweiter NTP-Server als Redundanz eingeplant werden. Dies kann über sogenannte **verkettete Server** realisiert werden.

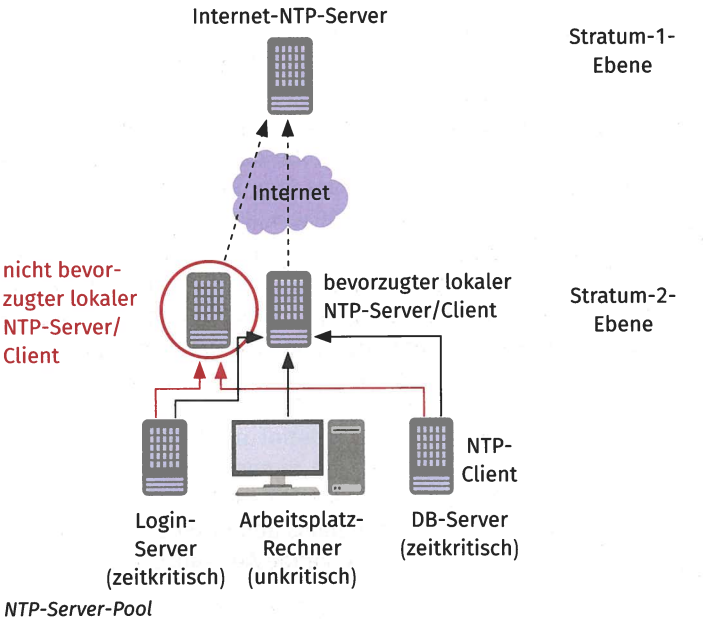


Einfache NTP-Lösung mit einem lokalen NTP-Server

Die beiden NTP-Server sollten auf unterschiedlicher Hardware eingerichtet werden. Beide Server z. B. auf dem identischen Virtualisierungssystem zu betreiben, würde die gewünschte Redundanz untergraben.

Eine weitere Absicherung des Zeitsystems stellt den Betrieb eines eigenen Stratum-1-Servers dar, der im Normalbetrieb als Client die Zeit über einen Internet-Server bezieht. Fällt diese Verbindung aus, bezieht der Server die Zeit von einer direkt angeschlossenen Uhr, die z. B. eine GPS-betriebene Uhr sein kann.

Bei der Planung muss ebenfalls beachtet werden, ob nur bestimmte Server zur Synchronisation angefragt werden sollen oder ein Pool genutzt werden soll. Ein Pool besteht dabei aus einer Gruppe von Servern, sodass die Ausfallwahrscheinlichkeit reduziert wird.



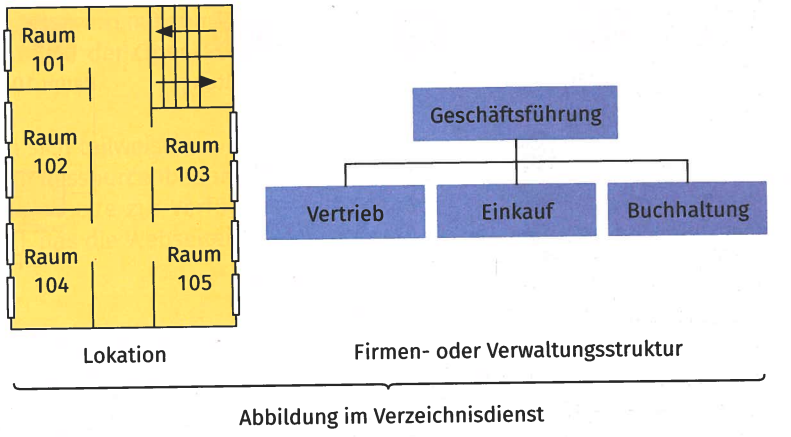
NTP-Server-Pool

(4) Windows-Domäne basierend auf logischer und physikalischer Netzwerkstruktur planen

Für die Planung einer Windows-Domäne ist es sinnvoll, einige Grundsätze zu beachten. Ein Gruppenrichtlinienobjekt enthält alle Gruppenrichtlinien und es ist prinzipiell möglich, alle nötigen Konfigurationen in nur einem Objekt durchzuführen. Zur Übersichtlichkeit sind aber folgende Richtlinien sinnvoll:

- Erstellen einer **OU-Struktur** (OU = Organizational Unit) in der Windows-Domäne, die getrennt ist nach Computer- und Benutzerkonten, da die Gruppenrichtlinien in diese zwei Kategorien unterteilt sind. Dadurch findet immer eine klare Trennung zwischen Richtlinien für Computer und Richtlinien für Benutzer statt.
- Für die **Benutzerstruktur** der OU empfiehlt sich die Organisationsstruktur der Firma.
- Für die **Computerstruktur** der OU empfiehlt sich, die Standort-, Gebäude- und Raumstruktur der Firma.
- Gruppenrichtlinienobjekte erhalten eine aussagekräftige Benennung und in der GPO werden Einstellungen für einen bestimmten Bereich vorgenommen.

Beispiel: Die Firma DOM ist in die Abteilungen Vertrieb, Einkauf und Buchhaltung eingeteilt. An dem Standort belegt sie fünf Räume (101–105), in denen für die Mitarbeiter Computer und Drucker zur Verfügung stehen. Für die Planung einer Windows-Domäne sollen die oben genannten Planungsregeln angewendet werden.

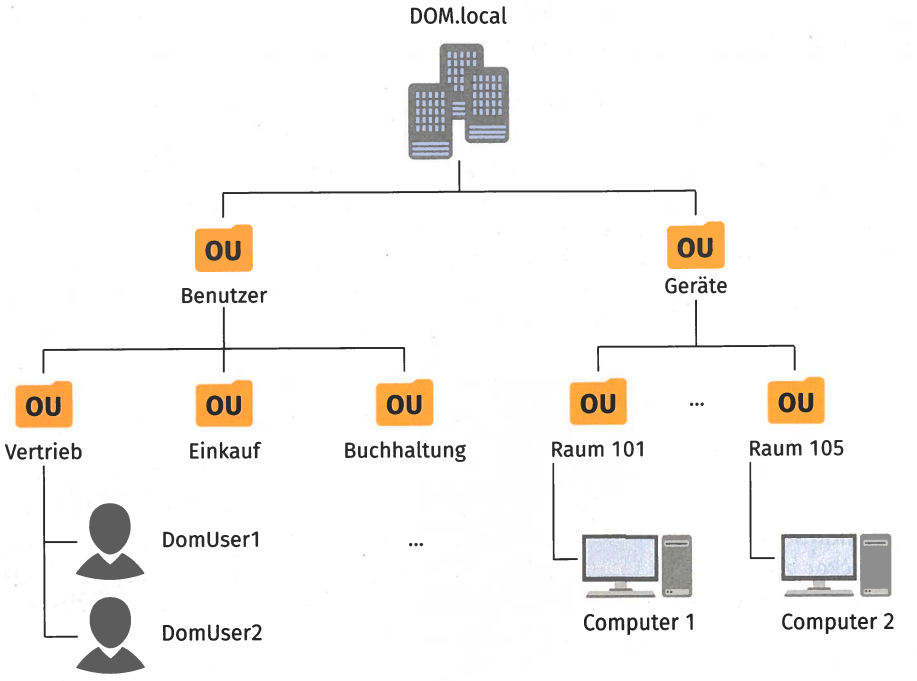


- Festlegung der Windows-Domäne: Hier bietet sich einfach der Firmenname „DOM" an.
- Identifizierung der Objekte:
  - Benutzer1, Benutzer2, ... BenutzerN
  - Computer1, Computer2, ... ComputerN
  - Drucker1, Drucker2, ... DruckerN
- Identifizierung der Organisationseinheit:
  - Vertrieb, Einkauf, Buchhaltung
  - Raum101, Raum102, ... Raum105

Schließlich werden die erstellten Objekte in der Domäne entsprechend den Planungsregeln angeordnet.

- Übernahme vorhandener Strukturen: Die Benutzer werden gemäß der Firmenstruktur in die OU mit Namen der Abteilungen gruppiert.
- Nutzung lokaler Ressourcen: Die Computer und Drucker werden gemäß der Raumaufteilung an dem Standort in eine OU mit entsprechenden Raumnamen gruppiert.
- Nutzung eingeschränkter Administratoren: Hier entscheidet sich, wie viele OUs wirklich nötig sind. Innerhalb einer OU können Benutzer spezielle administrative Aufgaben über die Objekte der OU erhalten. Jedes Mal, wenn so eine Delegation erwünscht ist, muss eine OU eingefügt werden.

Es ergibt sich schließlich eine entsprechende Domänenstruktur:



Domänenstruktur mit OU

Jetzt ist es möglich, beispielsweise Gruppenrichtlinienobjekte für abteilungsabhängige Einstellungen vorzunehmen. Benutzer, die in der entsprechenden OU einsortiert sind, erhalten die vorgesehenen Einstellungen. Neue Mitarbeiter werden in die OU der Abteilung einsortiert und erhalten dadurch ebenfalls die entsprechenden Einstellungen.

(5) Webserver planen

Webserver gehören heute zu den Standardservern in IT-Umgebungen (vgl. Jahrgangsband 1, Lernfeld 3, Kap. 3.2.3). Sie werden zur Bereitstellung von Webseiten und Webdiensten oder auch zur Administration von Geräten, vom Drucker bis zum Router, verwendet. Die Webserver-Planung kann in zwei Abschnitte geteilt werden: Zum einem müssen die technischen Voraussetzungen bestimmt werden, zum anderen müssen bei der Planung auch allgemeine Anforderungen festgelegt werden.

Die sichere Planung des Einsatzes eines Webservers wird vom BSI durch einen eigenen Baustein (Baustein APP.3.2 Webserver) unterstützt. Außerdem sollten die vom BSI im Baustein „SYS. 1.1 Allgemeiner Server" beschriebenen Anforderungen berücksichtigt werden. Dazu gehören beispielsweise ein geeigneter Aufstellungsort mit entsprechender Zugangskontrolle, eine geregelte Benutzerauthentisierung am Server, die Protokollierung aller Systemereignisse, der Betrieb einer Notstromversorgung oder das Backup-System.

! Ein Server sollte nie als Benutzerarbeitsplatz verwendet werden!

Im ersten Schritt muss geplant werden, für welchen Anwendungszweck der Webserver eingesetzt werden soll. Diese trivial klingende Festlegung auf den Anwendungszweck bedingt im weiteren Verlauf bestimmte technische Anforderungen. Es besteht z. B. ein Unterschied bei der Wahl und Konfiguration zwischen einem Webserver für die öffentliche Bereitstellung eines Webservices für einen Kundenkreis von mehreren tausend Anwendern und einem für die interne Bereitstellung des wöchentlichen Menüplans der Kantine.

Für eine fachgerechte Dokumentation sollten auch die bereitgestellten Inhalte und der Benutzerkreis beschrieben werden. Wichtig ist ebenfalls die Festlegung von Verantwortlichkeiten und Ansprechpartnern für den Webserver. Im Fall eines Fehlers oder eines Sicherheitsproblems kann durch eine klare Definition der verantwortlichen Stelle schnell reagiert werden.

Vor der Entscheidung für eine bestimmte Serversoftware steht die Entscheidung, ob der Server „On-Premise" arbeiten oder von einem externen Hoster gemietet werden soll. Wenn die Entscheidung auf einen externen Dienstleister fällt, muss der Grad der Dienstleistung bestimmt werden. Möglich ist beispielsweise das Anmieten eines virtuellen Servers (PaaS), die Miete eines laufenden Webservers oder die Miete eines Content-Management-Systems (SaaS). Bei der Entscheidung für ein externes Angebot muss vertraglich geregelt sein, welche Schutzmaßnahmen getroffen werden, wer Zugriff auf den Datenbestand hat, welche Maßnahmen im Fehlerfall ergriffen werden und welche Dienste, neben der Bereitstellung des Webservers, noch durch den Dienstleister erbracht werden.

Es stehen eine große Anzahl von Webservern mit zum Teil spezialisierten Einsatzgebieten zur Verfügung. Beliebte Webserver sind beispielsweise der Open-Source-Webserver „nginx" und der „Apache HTTP-Server".

Technische Anforderungen ergeben sich teilweise aus dem geplanten Anwendungszweck. Eine hohe Besucheranzahl bedingt einen höheren Ressourcenbedarf an CPU-Leistung, Arbeitsspeicher und Datenrate. Ein Webserver, der eine statische Webseite zur Verfügung stellt, benötigt weniger Ressourcen als ein Content-Management-System (CMS), das die Webseiten dynamisch zur Laufzeit aus Datenbankeinträgen erstellt.

Weitere technische Anforderungen sind die Festlegung auf eine (oder mehrere) IP-Adressen und die Überlegung, durch welchen OSI-Schicht-4-Port das System erreichbar sein soll. Ein öffentlich erreichbarer Dienst sollte idealerweise den Well-Known-Port (WKP) des Dienstes verwenden. Bei HTTPS wird der Port 443 verwendet (bei der unverschlüsselten Variante HTTP kommt der Port 80 zum Einsatz.). Es kann aber auch ein anderer Port genutzt werden, z. B. aus dem Bereich der „Registered Ports". Einen solchen Port wählt man, wenn der Well-Known-Port schon durch einen anderen HTTPS-Server belegt ist. Ein häufig anzutreffendes Beispiel ist die administrative Oberfläche zum Konfigurieren eines gemieteten Webservers. Während der Webserver über die Eingabe der URL im Browser (z. B. https://www.beispiel.de) auch ohne die Angabe eines Ports erreicht wird, benötigt das Laden der administrativen Oberfläche die explizite Angabe des entsprechenden Ports (z. B. https://www.beispiel.de:8443).

Bei der Planung der Installation eines Webservers sollte auch der Verwendungszweck der Zertifikate für Authentifizierung und Verschlüsselung beachtet werden. Ein öffentlicher Webserver benötigt ein öffentlich gültiges Zertifikat, das durch einen unbekannten Benutzer bzw. dessen Browser als gültig validiert werden kann. Das Zertifikat muss dazu von einer öffentlichen Zertifizierungsstelle signiert sein. Wird der Server nur im internen Netzwerk verwendet, ist es ausreichend, wenn ein Zertifikat einer internen Zertifizierungsstelle verwendet wird. Die internen Systeme müssen die Zertifizierungsstelle als vertrauenswürdig einstufen, indem das Stammzertifikat der Zertifizierungsstelle in den Zertifikatsspeicher der internen Systeme aufgenommen wird.

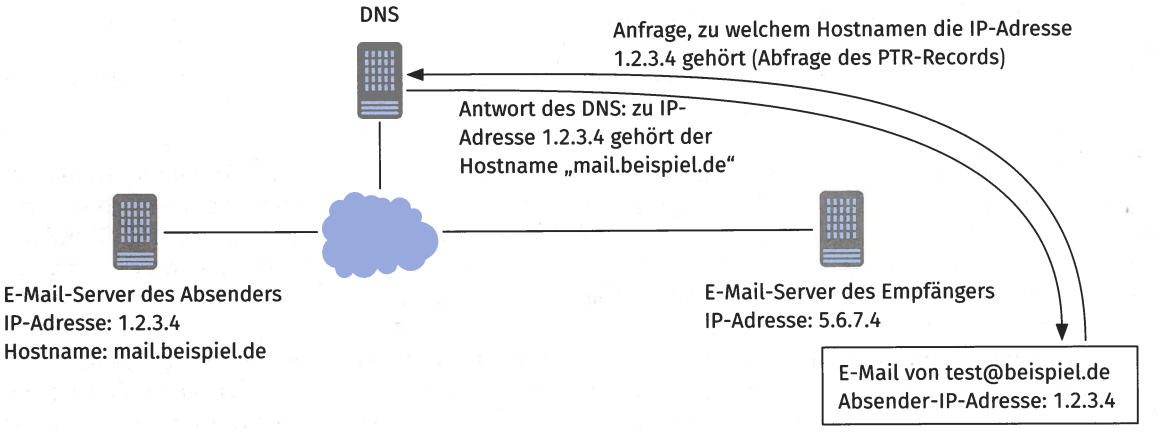
(6) E-Mail-Server planen

Wie Webserver gehören auch E-Mail-Server heute zu den Standardservern in IT-Umgebungen, da die Kommunikation mittels E-Mail einen großen Anteil der Unternehmenskommunikation ausmacht. Verschiedene Systeme können nicht nur E-Mails versenden, sondern entwickeln sich zu Kooperationsplattformen, mit denen z. B. die Integration eines gemeinsam geführten Kalenders mit Terminabsprachen möglich ist.

Analog zur Planung einer Webserver-Implementierung kann auch die Planung eines E-Mail-Servers in zwei Abschnitte geteilt werden. Einerseits müssen die technischen Voraussetzungen festgelegt werden, andererseits sind die allgemeinen Anforderungen zu beachten.

Der Betrieb eines eigenen E-Mail-Servers bietet verschiedene Vorteile. Administratoren haben bei einem selbst administrierten E-Mail-Server die Kontrolle über alle Systemeinstellungen. E-Mail-Postfächer können nach Belieben angelegt werden und Parameter, wie die Größe eines Dateianhangs, können unternehmensspezifisch vorgegeben werden. Außerdem kann die Abweisung von Spam oder die Entfernung unerwünschter Anhänge beliebig konfiguriert werden. Der Betrieb eines eigenen E-Mail-Servers erfordert es aber auch, die Verfügbarkeit des Server selbst zu garantieren. Ein Server bei einem Provider ist gegen Ausfälle meist gut abgesichert. Außerdem haben E-Mail-Provider eine große Datenbasis für die Entwicklung von Spam-Filtern.

Ein weiterer Nachteil beim Betrieb eines selbst betriebenen E-Mail-Servers ist die Anerkennung und die Annahme von E-Mails durch andere E-Mail-Provider. Oftmals werden E-Mails von kleinen, unbekannten E-Mail-Servern als Spam geblockt. Damit das nicht passiert, ist es sinnvoll, dass der E-Mail-Server unter einer *festen IP-Adresse* arbeitet. Außerdem benötigt der Server einen Hostnamen und entsprechende Einträge im öffentlichen DNS. Dazu gehört auch ein **PTR-Record** (Pointer Record). Diese Record-Art ordnet einer IP-Adresse einen Hostnamen zu. Wenn ein E-Mail-Server eine E-Mail empfängt, kennt er die Absender-IP-Adresse der empfangenen IP-Pakete. Mit dem PTR-Record kann der empfangende E-Mail-Server überprüfen, ob der Domain-Teil der Absender-E-Mail-Adresse auch zum versendenden E-Mail-Server gehört.



Überprüfung des absendenden E-Mail-Servers

In der Abbildung beispielsweise erhält der empfangende E-Mail-Server vom E-Mail-Server mit der IP-Adresse 1.2.3.4 eine E-Mail mit dem Absender „test@beispiel.de“. Der Server stellt nun an das DNS-Systen einen PTR-Request. Damit fragt der empfangende E-Mail-Server nach, welcher Hostname zur IP-Adresse „1.2.3.4“ gehört. Das DNS-System liefert den Hostnamen „mail.beispiel.de“ zurück. Wenn die Second-Level-Domain des zurückgelieferten FQDN, in diesem Fall „beispiel.de“, zum Domain-Teil der Absender-E-Mail-Adresse passt, wird die E-Mail akzeptiert.

Neben dem PTR-Record sollte auch ein **SPF-Record** (Sender Policy Framework) vorliegen. Mit dem Eintrag wird sichergestellt, dass ein E-Mail-Server die Berechtigung hat, eine E-Mail mit einem bestimmten Domain-Anteil zu versenden. Zusätzlich sollte auch das DomainKeys-Identified-Mail-Verfahren (DKIM) unterstützt werden (siehe Kapitel 1.1.1).

! Das BSI stellt einen Baustein (APP5.3) zur Verfügung, der den sicheren Betrieb eines E-Mail-Servers unterstützt. Der Server muss verschlüsselten Verkehr bereitstellen und darf nicht als Open-Relay betrieben werden. Dazu müssen sich Benutzer vor dem Versenden einer E-Mail beim Server authentifizieren. Der Baustein schreibt zusätzlich Datensicherung und Archivierung sowie einen Spam- und Virenschutz vor. Wie bei allen Servern sollte eine Sicherheitsrichtlinie für E-Mails definiert werden. In dieser Richtlinie werden beispielsweise Sicherheitstechniken und Verantwortlichkeiten festgelegt.

Es existiert eine Vielzahl von E-Mail-Servern verschiedener Entwickler mit unterschiedlichen Schwerpunkten. Alternativ zum On-Premise betriebenen E-Mail-Server kann ein E-Mail-Server auch in der Cloud als SaaS gemietet werden. Vor der Entscheidung für eine bestimmte Software ist es sinnvoll, weitere technische Anforderungen festzulegen.

Beispiel:

Checkliste	
Protokolle zum Abrufen von E-Mails	*z. B. IMAPS oder POP3S*
Protokoll zum Versenden von E-Mails	*SMTP*
TLS zur Verschlüsselung und Authentifizierung	*ja/nein*
Funktionsmailboxen/Mailboxen für Rollen	*ja/nein*
Filtermöglicheiten	*ja/nein*
Einrichten von Filtern und Regeln	*ja/nein*
Unterstützung von DKIM	*ja/nein*

(7) Videokonferenzsysteme und WebRTC planen

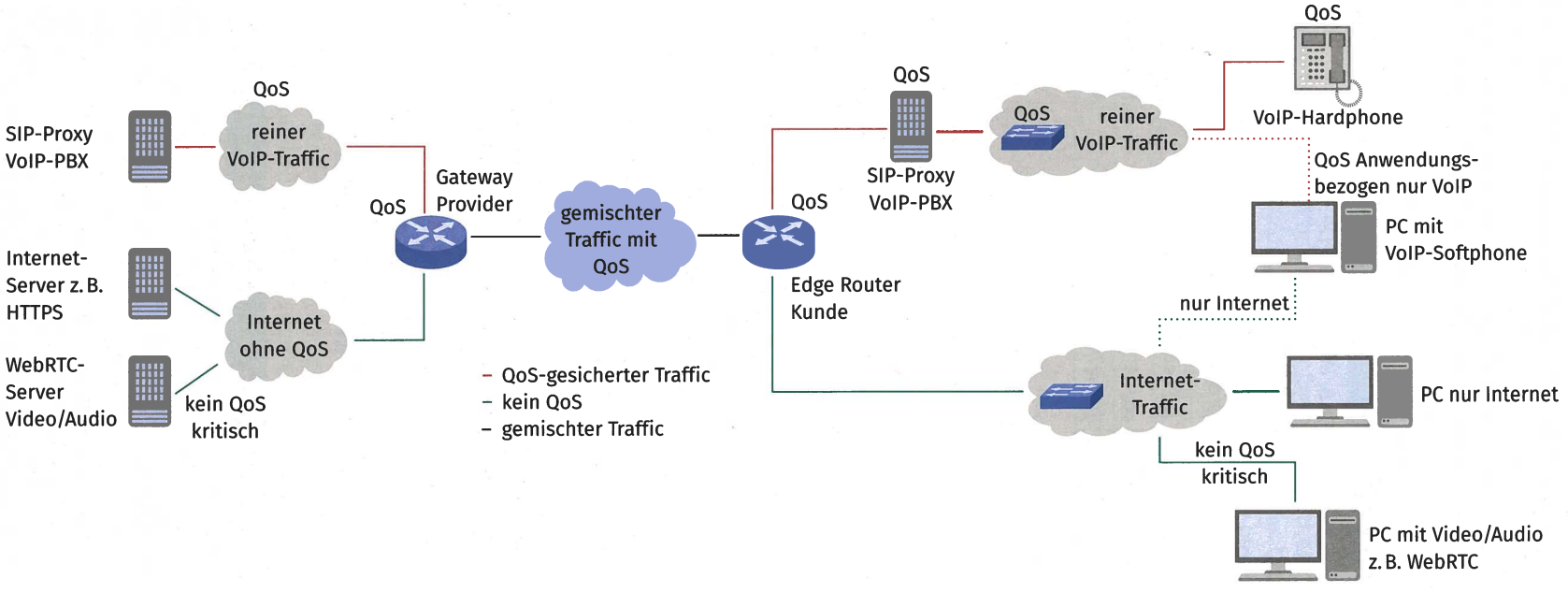
Bei der Planung eines Echtzeitdienstes sind besonders hohe Anforderungen zu berücksichtigen. Die Informationen müssen bei Sprechern und Empfängern immer zeitgleich vorliegen. Selbst wenige Millisekunden Unterschied führen zu Irritationen der Beteiligten. Man nennt diese Art der Übertragung isochrone Übertragung (isos, griech.: „gleich"; chronos, griech.: „Zeit"). Im alten Telekommunikationsnetz wurde dies durch einen zentralen Takt sichergestellt. In modernen IP-basierten Netzen existiert ein derartiger Mechanismus nicht und kann auch nicht nachgebildet werden. Eine weitere negative Beeinflussung können Engpässe innerhalb der IP-Netze darstellen. Ohne weitere Maßnahmen darf ein Router jedes beliebige Paket verwerfen, wenn die Übertragungsraten nicht ausreichen.

Um diese hohen Anforderungen zu erfüllen, wird meist Quality of Service (QoS) eingesetzt. Dabei ist es wichtig, dass die gesamte Übertragungsstrecke betrachtet wird. Ein einziger Engpass kann die gesamte Strecke beeinträchtigen. Einfacher Internet-Datenverkehr muss nicht priorisiert werden. Diese Art der Übertragung wird häufig als „best effort" bezeichnet.

Bei Rechnern, die über WebRTC kommunizieren, wird es hingegen schwieriger, den Traffic zu identifizieren. Meist wird hier HTTP als Basis verwendet, womit die Unterscheidung von normalem Internet-Datenverkehr erschwert wird. Auch die Providerseite muss betrachtet werden. Der Bereich WebRTC ist allerdings auch hier meist unpriorisiert.

Im lokalen Netz wird die Priorisierung meist per VLAN vorgenommen. Es ist demnach bei der Planung auf entsprechende VLAN-Unterstützung mit Priorisierungsmöglichkeit an den Netzelementen zu achten. Teilweise werden diese Features auch als VoIP-Priorisierung oder es werden nur allgemein QoS in den Spezifikationen aufgeführt.

Zu berücksichtigende Aspekte bei WebRTC-Planung	
Firewall-Konfiguration	Vermeidung von mehrfachem NAT → Probleme durch zwei unabhängige Datenverbindungen vermindern; Signalisierung (HTTP) und Echtzeitdaten (RTP) gemeinsam betrachten
Netzauslastung	Übertragungsraten kalkulieren; gleichzeitige Nutzer ermitteln
Audio-/Video-Hardware	Menge der benötigten Endgeräte planen



Beispiel mit QoS- und „best effort"-Verbindungen

Bei WebRTC-Anwendungen wie „BigBlueButton" (BBB) kommen neben den Videoübertragungen der Webcams auch die Screenshare-Datenraten (Bildschirmfreigabe) hinzu, sodass leicht mit mehreren Mbit/s gerechnet werden kann. Die folgende Beispielrechnung mit insgesamt 15 Nutzern soll eine Orientierung geben. Es wird jeweils die minimale und maximale Übertragungsrate kalkuliert. In der Praxis werden diese Werte nicht erreicht, da meist ein Mix der unterschiedlichen Übertragungsraten zum Einsatz kommt. Zu Planungszwecken ist eine solche Betrachtung allerdings hilfreich.

Annahmen pro Stream	Minimal	Maximal
Audio	40 kbit/s	60 kbit/s
Webcam	50 kbit/s	800 kbit/s
Screenshare	4 Mbit/s (ca. 1024 · 768)	8 Mbit/s (ca. 1920 · 1080)

Es wird davon ausgegangen, dass sich zehn Teilnehmer direkt und fünf weitere Teilnehmer über einen TURN-Server (Traversal Using Relays around NAT ein Hilfsprotokoll für NAT) in einer BBB-Sitzung zusammenfinden. Insgesamt sieben Teilnehmer nutzen ihre Webcams.

BBB-Kalkulation	Anzahl
Teilnehmer direkt	10
mit Webcam	4
mit Screenshare	1
Teilnehmer über TURN	5
mit Webcam	3
mit Screenshare	0
Teilnehmer gesamt	**15 (10 + 5)**
mit Webcam	7 (4 + 3)
mit Screenshare	1 (1 + 0)

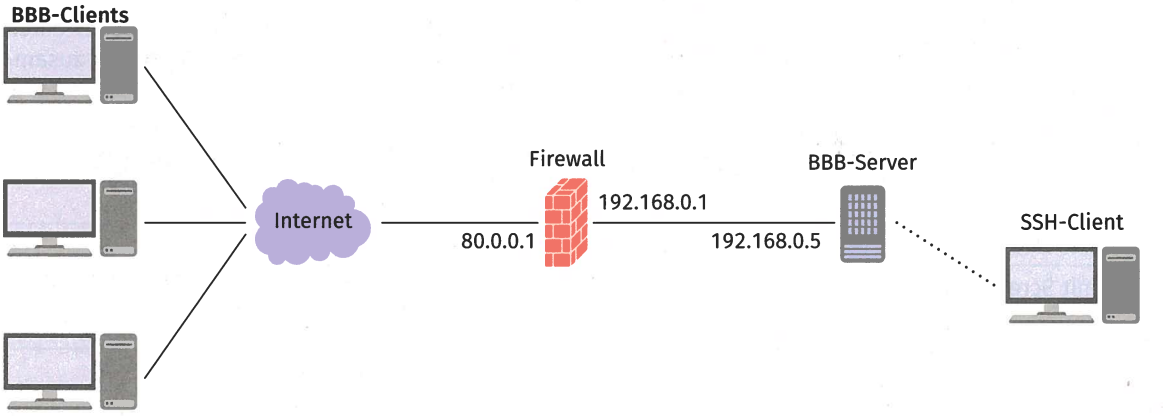
Berechnung der benötigten Übertragungsrate des BBB-Servers		
Eingehender Traffic Server	**Minimal**	**Maximal**
Audio	600 kbit/s (15 · 40 kbit/s)	900 kbit/s (15 · 60 kbit/s)
Video	350 kbit/s (7 · 50 kbit/s)	5600 kbit/s (7 · 800 kbit/s)
Screenshare	4000 kbit/s (1 · 4000 kbit/s)	8000 kbit/s (1 · 8000 kbit/s)
Eingehender Traffic gesamt	4,95 Mbit/s	11,5 Mbit/s
Ausgehender Traffic Server	**Minimal**	**Maximal**
Audio	600 kbit/s (15 · 40 kbit/s)	900 kbit/s (15 · 60 kbit/s)
Video	4900 kbit/s (14 · 50 kbit/s)*	78400 kbit/s (14 · 800 kbit/s)*
Screenshare	60000 kbit/s (15 · 4000 kbit/s)	120000 kbit/s (15 · 8000 kbit/s)
Ausgehender Traffic gesamt	65,5 Mbit/s	199,3 Mbit/s

**Das eigene Video wird nicht vom Server zurücktransportiert.*

Bei der Firewall-Konfiguration eines BBB-Servers müssen einige Ports freigegeben werden. Hierzu gehören die HTTP/HTTPS-Ports sowie die für die Medienstreamübertragung benötigten UDP-Ports zwischen 16384 und 32768.

Firewall-Freigaben	
Port	**Bemerkung**
80/443 TCP	HTTP/HTTPS wird für die gesamte Anwendung auf dem Nginx-Webserver benötigt.
16384 bis 32768 UDP	RTP-Ports für die Audio-/Video-/Screenshare-Daten
22 TCP (optional)	SSH wird zur remote-Konfiguration benötigt; wird der Server direkt konfiguriert, kann dieser Port entfallen.

Bei der Planung der Firewall-Regeln ist es wichtig zu wissen, auf welchen IP-Adressen der BBB-Server betrieben werden soll. Auf einem Server mit mehr als einer IP-Adresse wählt das Installationsskript unter Umständen eine nicht gewünschte Adresse aus. Um den Überblick zu behalten, ist ein Netzplan hilfreich.



Beispielhafter Netzwerkplan für den Zugriff auf einen WebRTC-Server über eine NAT-Firewall

In der Abbildung führt die Firewall neben den Filterfunktionen auch eine NAT-Funktion (von 80.0.0.1 auf 192.168.0.5) durch. Dies muss sowohl bei den HTTP/S-Ports wie auch den RTP-Ports berücksichtigt werden. Sofern der SSH-Client innerhalb des lokalen Netzes liegt, muss hier keine Firewall-Regel vorgesehen werden.

(8) VoIP planen

Für Voice over IP (VoIP) sind hinsichtlich QoS dieselben Anforderung zu erfüllen, da auch hier Personen isochron miteinander verbunden werden. Neben Personen wurden über die alten Telekommunikationsnetze auch Anwendungen wie Magnetkartentelefone (Terminals zum bargeldlosen Bezahlen mit Kredit- bzw. EC-Karten) oder analoge Faxgeräte verbunden. Daher sind derartige Verbindungen über ein VoIP-Netz nur mit zusätzlichen Hilfsmechanismen möglich. Weichen die Paketlaufzeiten innerhalb eines Datenstroms zu stark voneinander ab, so kann die entstehende Verzögerungsschwankung (Jitter) die Synchronizität der Signale zerstören und es kommt zum Verbindungsabbruch. Moderne Magnetkartentelefone arbeiten aus diesem Grund direkt mit Datenpaketen und VPN-Verbindungen, weil hier ohnehin nur Datensätze ausgetauscht werden. Bei analogen Faxgeräten ist dies nur mithilfe von Protokollen wie z. B. T.38 (Fax over IP) möglich.

Bei VoIP muss daher ebenfalls QoS eingeplant werden. Bei reinen VoIP-Geräten ist die Priorisierung einfach, da alle Datenpakete zu priorisieren sind. Hier kann die Priorisierung direkt am Switch-Port erfolgen. Bei Endgeräten wie einem PC mit VoIP-Software, die sowohl Internet-Traffic als auch zu priorisierenden Traffic generieren, wird dies erschwert. Die Priorisierung muss hier anwendungsbezogen erfolgen. Sofern es sich beispielsweise um SIP/RTP-Traffic handelt, kann dieser klar identifiziert und mit geeigneter Hardware priorisiert werden. Die Priorisierung im Providernetz wird bei den meisten Providern durch den Betrieb eigener VoIP-Netze, die entsprechend priorisiert sind, gewährleistet. Die Priorisierung im lokalen Netz wird bei VoIP häufig durch Einsatz von VLAN gewährleistet.

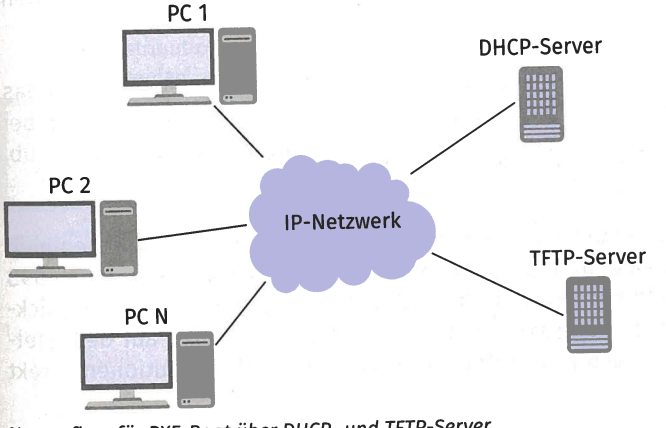
Zu berücksichtigende Aspekte bei VoIP/WebRTC-Planung	
Verkabelung	VoIP-Endgeräte wie Telefone mit PoE (Power over Ethernet) einsetzen zur Reduktion der Verkabelungsaufwände; VoIP-Endgeräte über eigenes Netzkabel anschließen, sofern möglich
Firewall-Konfiguration	Vermeidung von mehrfachem NAT → Probleme durch zwei unabhängige Datenverbindungen vermindern; Signalisierung (SIP) und Echtzeitdaten (RTP) gemeinsam betrachten
Netzauslastung	Übertragungsraten kalkulieren; gleichzeitige Nutzer ermitteln
VoIP-Hardware	Menge der benötigten Endgeräte planen
Alte Telefonie-Hardware	alte Telefone oder Anlagen dahingehend überprüfen, ob diese weiterbetrieben werden müssen

Eine VoIP-Verbindung kann mit ca. 60 bis 80 kbit/s kalkuliert werden. Erst bei einer größeren Anzahl gleichzeitiger Gespräche (ca. 50) kommen bei modernen Anschlüssen beachtenswerte Übertragungsraten von ca. 4 Mbit/s zustande. Entscheidend wird die Übertragungsrate auf dem Internet-Anschluss im Bereich „Upload". Dieser ist insbesondere bei DSL-basierenden Anschlüssen meist geringer als der Download.

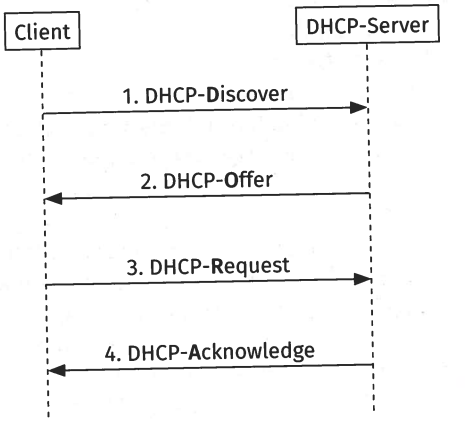
Bei der Firewall-Konfiguration eines VoIP-Servers müssen einige Ports freigegeben werden. Hierzu gehören die SIP-Ports (5060/5061) sowie die für die Medienstreamübertragung benötigten UDP-Ports zwischen 20000 und 50000. Bei der Planung der Firewall-Regeln ist es wichtig zu wissen, auf welchen IP-Adressen der VoIP-Server betrieben werden soll (siehe auch Lernfeld 10b, Kapitel 1.3.1).

(9) PXE-Server planen

Mit **Preboot Execution Environment (PXE)** kann ein Computer das Betriebssystem über das Netzwerk laden. Die Voraussetzung für PXE ist, dass der Client über eine Netzwerkkarte verfügt, die das Booten über das Netzwerk unterstützt. Zusätzlich muss im BIOS/UEFI das „Booten über Netzwerk" (oder ein ähnlicher Punkt) ausgewählt werden können. Die Funktion des PXE ist in der Netzwerkkarte integriert.



Netzaufbau für PXE-Boot über DHCP- und TFTP-Server



DORA-Prinzip bei DHCP

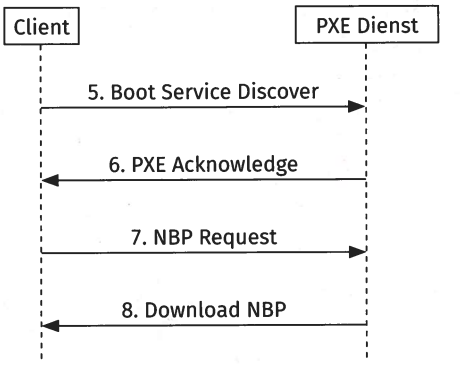
Für PXE ist ein DHCP-Server und ein TFTP-Server (Trivial File Transfer Protocol; einfacher als FTP) nötig. Der DHCP-Server weist dem Client zunächst eine IP-Adresse zu. Der Ablauf entspricht dem DORA-Prinzip des DHCP. Im Wireshark-Mitschnitt kann man erkennen, dass der DHCP-Server in der DHCP-Acknowledge-Nachricht den Namen des NBP (Network Bootstrap Program) mitteilt. Ein Servername ist nicht nötig, da der TFTP-Dienst in diesem Beispiel unter der gleichen IP-Adresse erreichbar ist.

	Source	Destination	Protocol	Length	Info
1	0.0.0.0	255.255.255.255	DHCP	590	DHCP Discover
3	10.0.0.2	255.255.255.255	DHCP	342	DHCP Offer
5	0.0.0.0	255.255.255.255	DHCP	590	DHCP Request
6	10.0.0.2	255.255.255.255	DHCP	342	DHCP ACK

Server host name not given
Boot file name: pxelinux.0

Wireshark-Mitschnitt eines DHCP-Requests

Nachdem der Client eine eigene IP-Adresse vom DHCP-Server zugewiesen bekommen hat, sucht er im Netzwerk nach einem PXE-Server. Dazu sendet er eine „Boot Service Discover"-Nachricht, diese ähnelt der DHCP-Discover-Nachricht (siehe Nachricht 6). Der PXE-Server antwortet mit Verweis auf das NBP. Das ist ein Bootloader, mit dem ein Image geladen werden kann. Bei PXE können verschiedene Images zur Verfügung stehen. Der Client fordert das angefragte Image an (Nachricht 7) und der Server sendet dem Client das entsprechende Image (Nachricht 8).



PXE-Boot

Alternativ zur PXE kann das einfachere „BOOTstrap Protocol" (BOOTP) eingesetzt werden. Mit diesem Protokoll kann einem System eine IP-Adresse zugewiesen werden. Zusätzlich wird dem bootenden System der Name und die Position einer Netzwerk-Boot-Datei mitgeteilt. Im Gegensatz zur PXE kann nur ein Boot-Image zur Verfügung gestellt werden.

(10) Software-Aktualisierung planen

In der Informationstechnik durchläuft fast jede Hardware und Software einen Produktzyklus, der nach der Einführung Aktualisierungen notwendig macht. Dabei handelt es sich um Aktualisierungen zur Funktionserweiterung oder -reduktion. Oft sind es aber sicherheitsrelevante Aktualisierungen, die auf jeden Fall zeitnah durchgeführt werden sollten.

Aktualisierungen sollten allerdings nicht vollautomatisch durchgeführt werden, da jede davon auch das Risiko von Inkompatibilitäten mit bestehenden Systemen mit sich bringt. Daher sollte insbesondere bei Funktionsänderungen vor jeder Aktualisierung ein ausführlicher Test stattfinden (siehe auch Lernfeld 10b, Kapitel 1.4.2).

Eines der ältesten Update-Systeme stellt das Paketverwaltungssystem von Linux dar. Genau genommen sind alle aktuellen App Stores von Apple, Google und Microsoft proprietäre Kopien dieser bereits 1993 eingeführten Idee. Die Motivation lag zu Beginn vor allem in der Automatisierung von Softwareentwicklung. Später wurde die Software nicht mehr in Form von Source-Code ausgeliefert und auf dem Zielsystem in ausführbaren Code übersetzt. Es wurden von den unterschiedlichen Distributionen direkt einsatzfähige Pakete bereitgestellt.

Name des Paketmanagers	Beschreibung
rpm	RedHat Package Manager; wird seit 1994 eingesetzt; einer der ältesten Paketmanager und sehr weit verbreitet; Einsatz in „RHEL/CentOS/Fedora"
dpkg	Debian Package Manager; seit 1994 im Einsatz; gehört ebenfalls zu den ältesten Paketmanagern und ist in vielen debian-basierten Distributionen vertreten; Einsatz bei debian/Ubuntu

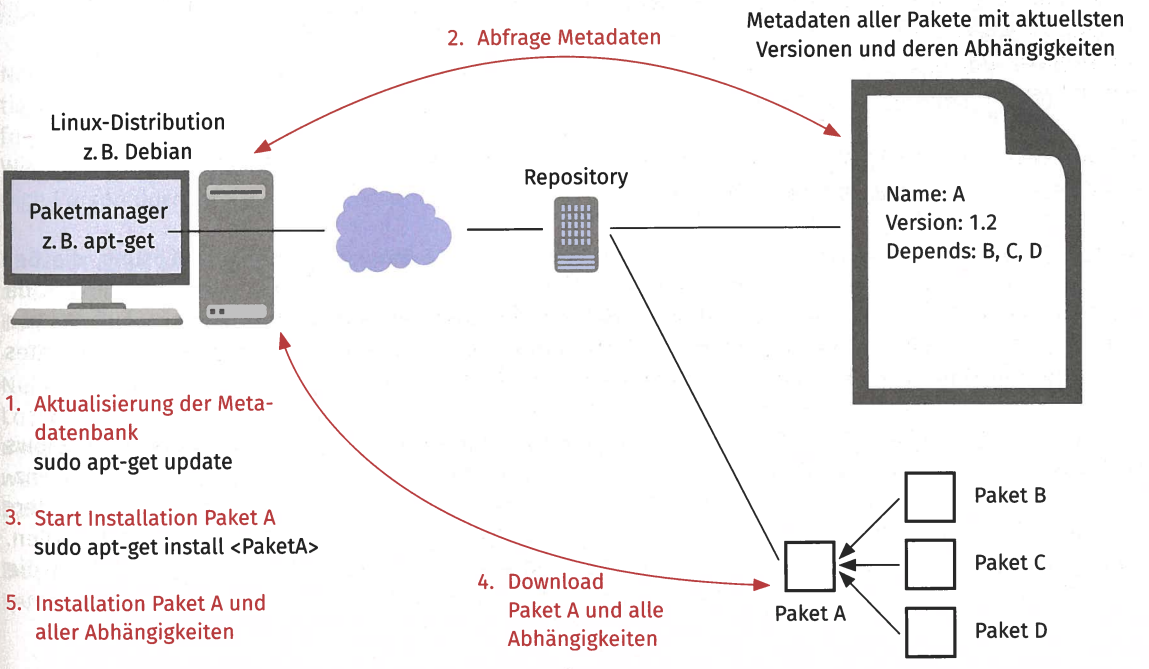
apt-get / apt	Advanced Package Tool; seit 1998 in debian-basierten Distributionen zu finden; wurde vor allem durch Ubuntu bekannt; „apt" ist die neuere Variante, die die Funktionen von „apt-get" und „apt-cache" zusammenführt.
yast	Yet another Setup Tool; seit 1996 wird „yast" von der deutschen Distribution „SuSE Linux" und „openSUSE" eingesetzt.
yum	Yellowdog Updater, Modified; seit 2002 wird „yum" zur Verwaltung von RPM-Paketen eingesetzt.

Mit dem Befehl „apt-cache show <PAKETNAME>" kann man alle Informationen und Abhängigkeiten (Metadaten) zu einem bestimmten Paket anzeigen lassen.

```
apt-cache show openssh-client
# oder
apt show openssh-client

# Ausgabe in Auszügen
Package: openssh-client
Architecture: amd64
Version: 1:8.2p1-4ubuntu0.5
Depends: adduser (>= 3.10), dpkg (>= 1.7.0), passwd, libc6 (>= 2.26),
libedit2 (>= 2.11-20080614-0), libfido2-1 (>= 1.3.0), libgssapi-krb5-2
(>= 1.17), libselinux1 (>= 1.32), libssl1.1 (>= 1.1.1), zlib1 g (>= 1:1.1.4)
...
Description-de: Secure Shell (SSH) Client, für den sicheren Zugang zu
entfernten Rechnern ...
```

Durch eine lokal verwaltete Datenbank, die alle installierten Pakete enthält, können nicht mehr benötigte Pakete identifiziert werden.



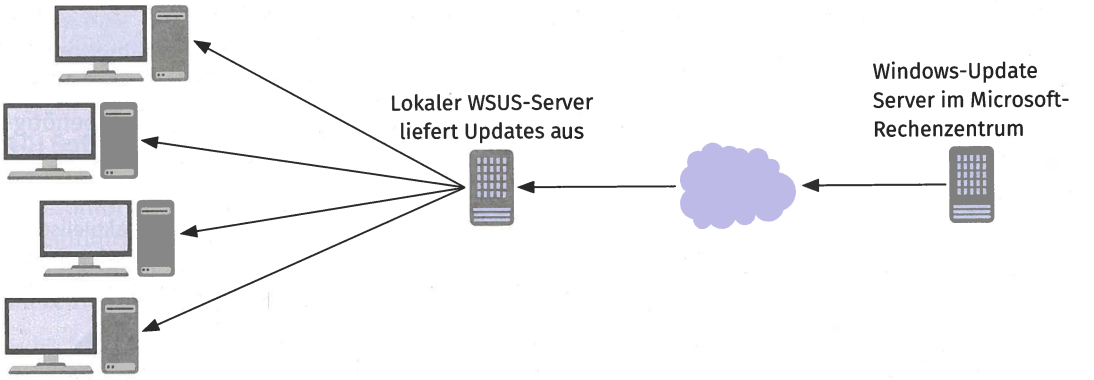
Ablauf bei der Nutzung einer Paketverwaltung

Wird beispielsweise ein Paket A deinstalliert, ermittelt das Paketmanagement alle Abhängigkeiten (Depends) zu diesem Paket. Zu jedem dieser Depends wird dann ermittelt, ob noch andere installierte Pakete dieses Depend benötigen. Ist Paket A das letzte Paket für dieses Depends gewesen, so kann das Depend deinstalliert werden, da es keine Verwendung mehr hat. Durch sogenannte Meta-Pakete wird dabei sichergestellt, dass alle wichtigen Pakete des Kernsystems unberührt bleiben.

Das Paketmangement wird in den meisten Distributionen so vorkonfiguriert, dass die Metadaten des Repositorys in regelmäßigen Abständen auf neue Versionen geprüft werden. Diese werden den Benutzern dann zur Installation angeboten.

Microsoft hat diesen Mechanismus mit Einführung von Windows 10 in ähnlicher Form umgesetzt. Dabei können Benutzer lediglich die Installation verzögern, aber nicht grundsätzlich verhindern. Zusätzlich bietet Microsoft noch einen App Store an, über den auch Fremdsoftware installiert und aktuell gehalten werden kann. Google hat ein ähnliches System für das Betriebssystem „Android" und seine Apps etabliert. Google geht hier den Weg, dass nur Kernfunktionen und sicherheitsrelevante Updates für das Betriebssystem zentral angeboten werden. Die restlichen Funktionen des Betriebssystems müssen vom Hersteller des Geräts aktuell gehalten werden. Größter Nachteil dabei besteht in den zum Teil stark abweichenden Versionsständen der aktiven Hardware. Apple pflegt alle Geräte mit neuesten Betriebssystemversionen, deren Installation ähnlich wie bei Microsoft erzwungen wird.

In großen Unternehmen mit vielen Geräten ist es sinnvoll, die Aktualisierungen gesteuert einzuspielen und dies nicht den Benutzern zu überlassen. Hierzu werden Update-Manager eingesetzt.



Prinzipieller Aufbau des Windows Server Update Service

Microsoft liefert mit dem WSUS-Server (Windows Server Update Service) eine eigene Lösung, die Betriebssystemaktualisierungen lokal installiert. Dies führt zu einer massiven Reduktion der Downloads, da die Pakete von den Clients lokal vom WSUS-Server bezogen werden, statt über das Internet direkt von Microsoft. Die Grundfunktionalität eines WSUS-Servers umfasst zunächst nur Betriebssystem-Updates und unternehmenseigene Software wie MS Office.

Darüber hinaus sind auch andere Anwendungen aktuell zu halten. Hierfür können im Bereich MS Windows kommerzielle Produkte eingesetzt werden. Bei der Auswahl dieser Produkte sollte auch auf die Lizenzverwaltungs- und Inventarisierungsfunktionen geachtet werden. Der Aufwand, die Lizenzen von hunderten Rechnern zu verwalten, ist nicht zu unterschätzen und sollte daher systemisch unterstützt werden. Die Inventarisierung der eingesetzten Anwendungen ist ebenfalls wichtig, um den Überblick über die genutzten Anwendungen zu behalten. Diese Informationen können zur Planung von Migrationen auf neue Anwendungen sehr hilfreich sein.

(11) Datenschutzkonformes Backup planen

In einem Unternehmen wird an vielen Stellen, z. B. bei dem Erstellen von Rechnungen, mit personenbezogenen Daten gearbeitet. Immer wenn diese Daten und die dazugehörigen Dokumente abgespeichert werden, muss die DSGVO (Datenschutzgrundverordnung) beim Erstellen eines Backup-Konzepts beachtet werden.

In Artikel 5f) der DSGVO über die „Grundsätze für Verarbeitung von personenbezogenen Daten" wird angegeben, dass Daten so verarbeitet werden müssen, dass sie auch vor unbeabsichtigtem Verlust, unbeabsichtigter Zerstörung oder Schädigung geschützt sind. Ein solcher Schutz wird durch ein Backup-Konzept unterstützt.

In Artikel 32 der DSGVO wird ein deutlicher Bezug auf Backups genommen. Nach Absatz 1 c) müssen personenbezogene Daten und der Zugang zu diesen Daten nach Beschädigung oder unbeabsichtigter Löschung schnell wiederherstellbar sein. Ein Backup ist eine technische und organisatorische Maßnahme (TOM), die diese Anforderung an den Zugang und die Unversehrtheit von personenbezogenen Daten erfüllt.

Backup-Konzept

Das Backup-Konzept muss festlegen, welche Daten von welchen Systemen mit dem Backup gesichert werden. Dieser Punkt ist essenziell, da Daten verloren gehen können, wenn nicht alle Systeme in das Backup eingebunden sind. Es muss festgelegt werden, wie oft ein Backup durchgeführt wird und welche Art von Backup (volles, differenzielles oder inkrementelles) eingesetzt wird. Außerdem muss bestimmt werden, an welcher Stelle die Backups aufbewahrt werden. Dabei ist zu beachten, ob ein schneller Zugriff auf das Backup nötig ist oder das Backup beispielsweise in einem Bankschließfach aufbewahrt werden kann. Ebenfalls muss der Zugriff auf die Backups definiert werden. Jeder, der auf ein Backup physischen Zugriff hat, kann u. U. auch auf die Daten des Backups zugreifen. Zusätzlich sollte das Backup-Konzept festlegen, auf welche Weise das Backup verschlüsselt ist. Abschließend sollte das Backup-Konzept eine Anleitung enthalten, wie die Daten gesichert bzw. zurückgespielt werden können.

Nach dem Erstellen des Backup-Konzepts ist es sinnvoll, die Backup-Software zu überprüfen. Es ist wichtig, dass nur berechtigte Personen die Software verwenden können. Damit wird verhindert, dass Unbefugte einen ungewollten Wiederherstellungsprozess starten. Sinnvollerweise werden alle Backup- und Wiederherstellungsprozesse überwacht und protokolliert. Bei Fehlern oder Warnungen muss die zuständige Person durch die Software informiert werden.

Im Normalfall wird ein Backup in regelmäßigen Abständen über einen bestimmten Datenbestand gesichert. Das Abändern eines solchen Prozesses, z. B. eine Änderung der Taktung oder der zu sichernden Daten, sollte durch den Benutzer separat bestätigt werden. Damit werden Konfigurationsfehler reduziert.

Neben der Vorgabe, ein Backup für personenbezogene Daten zu erstellen, ist es aber auch nötig, ein Löschkonzpt für personenbezogene Daten vorzuhalten. Artikel 17 DSGVO räumt betroffenen Personen ein umfangreiches Recht auf Löschung von personenbezogenen Daten ein. Das ist besonders dann wichtig, wenn die Aufbewahrungsfrist für zweckgebundene Daten abgelaufen ist und die personenbezogenen Daten gelöscht werden müssen. Diese Daten müssen auch aus den Backups entfernt werden.

Bei der Erstellung eines Löschkonzepts werden die Verantwortlichen durch die DIN 66398 unterstützt. Diese Norm stellt eine Leitlinie für die Inhalte und den Aufbau eines Konzepts zur Verfügung.

Löschkonzept

Ein Löschkonzept sollte Daten in sogenannte Löschklassen einteilen. Daten einer Löschklasse werden gleich behandelt. Außerdem sollte ein Löschkonzept Löschregeln enthalten. Eine Löschregel muss die Datenart und die Dauer der Datenhaltung beschreiben und somit die Frist festlegen, wann die Daten gelöscht werden müssen. Die Umsetzungsvorgaben beschreiben die eigentliche Löschmaßnahme.

(12) Revisionssichere Archivierung planen

Der Begriff des revisionssicheren Archivs leitet sich von den „Grundsätzen zur ordnungsmäßigen Führung und Aufbewahrung von Büchern, Aufzeichnungen und Unterlagen in elektronischer Form sowie zum Datenzugriff" (GoBD) ab. Die Beachtung dieser Grundsätze wird durch das Bundesministerium der Finanzen vorgeschrieben. Ursprünglich sind die GoBD für Buchungen und Rechnungen vorgesehen. Sie lassen sich aber auch auf die Archivierung von Daten anwenden, besonders wenn beispielsweise Daten aus dem Rechnungswesen dauerhaft gesichert werden sollen.

Allgemeine Anforderungen der GoBD	
Anforderung	**Erklärung**
Nachvollziehbarkeit und Nachprüfbarkeit	Alle zu archivierenden Daten müssen einem Besitzer zugeordnet werden können. Es muss protokolliert sein, wann ein Dokument erstellt und wann es von welcher Person verändert wurde.
Vollständigkeit	Daten, z. B. die Daten eines Vorgangs, müssen komplett archiviert werden. Dabei ist zu beachten, dass die Daten eines Vorgangs auch aus unterschiedlichen Quellen stammen können.
Richtigkeit	Die Originaldaten und die archivierten Daten müssen übereinstimmen. Während des Archivierens darf es zu keiner Änderung kommen. Die Richtigkeit kann durch die Verwendung von Prüfsummen sichergestellt werden.
Zeitgerechte Buchung und Aufzeichnungen	In Bezug auf eine Archivierung bedeutet das, dass die Dokumente nach dem Abschluss eines Vorgangs zeitnah archiviert werden.
Ordnung	Zusammengehörige Dokumente sollen sinnvollerweise auch in einer Struktur archiviert werden, die die Zusammengehörigkeit abbildet. Zusammenhangloses archivieren führt dazu, dass Daten ggf. nicht ordnungsmäß zugeordnet werden können.
Unveränderbarkeit	Wie oben bei der Anforderung der Nachvollziehbarkeit beschrieben, muss jede Veränderung der Daten protokolliert werden. Ein nachträgliches Ändern von archivierten Daten darf nicht möglich sein. Eine Änderung führt zu einer Kopie der Daten, die wiederum archiviert werden muss.

Das Sichern von Daten mittels Backup-Verfahren und die Archivierung von Daten sind zwei ähnliche Konzepte, die aber eine andere Zielsetzung verfolgen.

Unterschied zwischen Archiv und Backup

Archiv
Die Archivierung von Daten hat das Ziel, die Daten langfristig aufzubewahren. Für manche Daten, z. B. für Rechnungen, gibt es für Unternehmen gesetzliche Aufbewahrungsfristen. Diese Daten müssen also langfristig gesichert werden. Im Gegensatz zum Backup enthält ein Archiv keine Da-

ten, mit denen das Unternehmen noch arbeitet. Während bei der Wiederherstellung eines Backups auf eine kurze Wiederherstellungszeit geachtet wird, ist eine schnelle Lesegeschwindigkeit eines Archivs nur eine Nebenanforderung. Wichtiger ist es, dass Daten im Archiv gefunden werden können. Das wird durch eine Indizierung der Daten gewährleistet. Archivierte Daten müssen vor jeglicher Art von Veränderung geschützt werden. Die Integrität der Daten muss gewährleistet sein. Als Medium für eine Archivierung werden vor allem Datenbänder (Tapes) eingesetzt. Diese sind im Vergleich günstiger und haben eine lange Lebensdauer.

Backup
Das Ziel eines Backups ist es, Daten im Fall von unbeabsichtigter Löschung, Datenverlust oder Beschädigung schnellstmöglich wieder verfügbar zu machen. Dazu wird von den Daten in regelmäßigen Abständen eine Kopie erstellt. Als Backup-Medien kommen beispielsweise Festplatten oder Tapes zum Einsatz.

Kompetenzcheck

1. Beschreiben Sie in Stichworten die Vorteile, die sich durch das Erstellen eines IP-Vergabeplans ergeben.

2. Nennen Sie neben der IP-Adresse und der Subnetzmaske drei weitere Parameter, die der DHCP-Server an Clients verteilen kann.

3. Erläutern Sie zwei Anwendungsfälle für den Einsatz des NTP.

4. Erklären Sie, warum ein Server nicht als Benutzerarbeitsplatz verwendet werden sollte.

5. Erläutern Sie die Aufgabe eines PTR Record im DNS-System.

6. Gegeben sind die benötigten Datenraten für die Kommunikation während einer Videokonferenz. Es konferieren fünf Teilnehmer mit Webcam. Außerdem wird Screensharing betrieben. Geben Sie die benötigte ausgehende Datenrate des Servers in MByte/s an.

Annahmen pro Stream	Maximal
Audio	55 kbit/s
Webcam	750 kbit/s
Screenshare	8 Mbit/s (1920 · 1080)

7. Geben Sie zwei Informationen an, die der Client als Nächstes benötigt, nachdem er eine IP-Adresse zugewiesen bekommen hat.

8. Erklären Sie den Vorteil, den die Verwendung eines Paketmanagers bietet.

9. Geben Sie an, ob die folgenden Aussagen richtig sind. Begründen Sie Ihre Auswahl.
   a) Archivierte Daten sollen verändert werden können.
   b) Die Begriffe „Backup“ und „Archiv“ sind Synonyme.
   c) Eine Archivierung hat das Ziel, Daten langfristig aufzubewahren.
   d) Backup-Medien können durch neue Backups überschrieben werden.

10. Bearbeiten Sie die Aufgaben 1 bis 6 der Lernsituation 3 im Arbeitsbuch.

A1
A6

1.3.2 Container planen

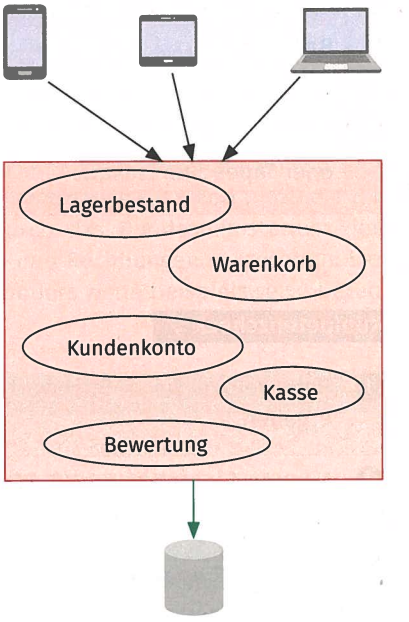
S **Sie sollen zunächst die Funktionsweise von Containern kennenlernen, um anschließend deren Einsatz planen zu können.**

(1) Microservices

Microservices sind ein Designkonzept, Softwareanwendungen in kleine abgekapselte Anwendungen aufzuteilen, die vernetzt zusammenarbeiten. Durch diese Aufteilung sind Microservices gut geeignet für die Implementierung von Containern in größeren Architekturen.

Um Microservices zu beschreiben, soll als Beispiel ein Onlineshop dienen, der sein Inventar durchsuchbar anzeigt, Kundenkonten verwaltet, Warenkörbe zur Verfügung stellt und für den Kaufabschluss den Bezahlvorgang an der Kasse ausführt. Für die angebotenen Waren können Kommentare und Bewertungen vergeben werden.

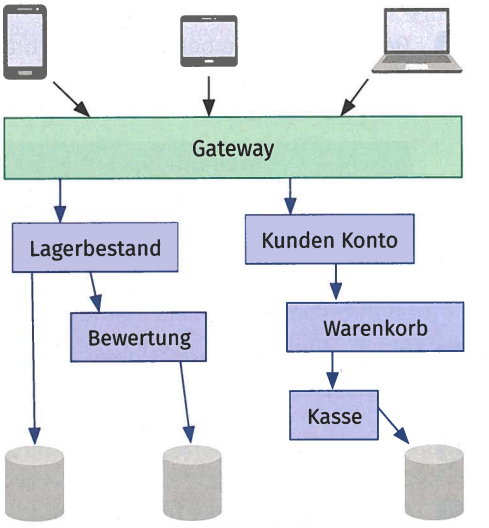
In einer herkömmlichen Implementierung, die als monolithisch bezeichnet wird, hat die Umsetzung eine gemeinsame Softwarebasis. Alle Funktionen sind dadurch softwaretechnisch miteinander verbunden, obwohl möglicherweise funktionell teilweise keine Verbindung besteht. Ein solcher Monolith kann durch Vergrößern der Ressourcen oder Hinzufügen von weiteren Instanzen, die über Load Balancer angesprochen werden, administriert werden. Schwieriger wird das für die Datenbank. Hier können Ressourcen zwar ergänzt werden, ein Load Balancing benötigt aber eine meist sehr komplexe Synchronisation der Datenbasis.



Beispiel für monolithische Microservices

Ein über Microservices aufgebautes System enthält eine funktionelle Aufteilung. Dadurch sind funktionelle Einheiten in einer eigenen Softwarebasis enthalten. Diese kleinen Einheiten werden über ein Netzwerk miteinander verbunden. Die Anfragen der Clients werden über eine einheitliche Schnittstelle durch das sogenannte Gateway entgegengenommen und von dort an die verarbeitenden Einheiten verteilt. Die kleinen Einheiten können sich dadurch im Hintergrund verändern, ohne dass eine Anpassung der Clients notwendig wird.

Durch die Aufteilung in kleine Einheiten kann die Datenspeicherung auf mehrere Datenbanken verteilt werden, so dass auch die Skalierung einfacher wird. Wenn die Funktion einer kleinen Einheit skaliert werden muss, muss auch die Einheit und die zugehörige Datenbank entsprechend skaliert werden. Durch die bessere Skalierbarkeit der Microservices ist eine entsprechende Implementierung gut für die Umsetzung mit Container-Architekturen geeignet.



Verteilung der Anfragen durch das Gateway auf die Container

(2) Orchestrierung: Aufbau, Grundlagen, Struktur

Mit den Microservices haben Anwendungsentwickler also ein Muster, um Software und Systeme gut skalierbar zu gestalten. Mit den Containern ist es Administratoren möglich, diese isoliert von Abhängigkeiten auf Servern zur Ausführung zu bringen. Ein Microservice-System kann aus einer Vielzahl von Contai-

nern bestehen. Diese lassen sich nicht mehr so einfach mit Skripten managen. Weiterhin können die Anforderungen an eine Cloud, bei der Kunden selbstständig Ressourcen hinzufügen oder wegnehmen können, Rechenleistung für mehrere Kunden zusammengefasst angeboten wird oder bei der Kapazitäten elastisch provisioniert oder freigegeben werden (siehe Kapitel 1.1.2), einen Überbau zur Orchestrierung der Container nötig machen. Diese Orchestrierung kann z. B. durch die Software „Kubernetes" umgesetzt werden.

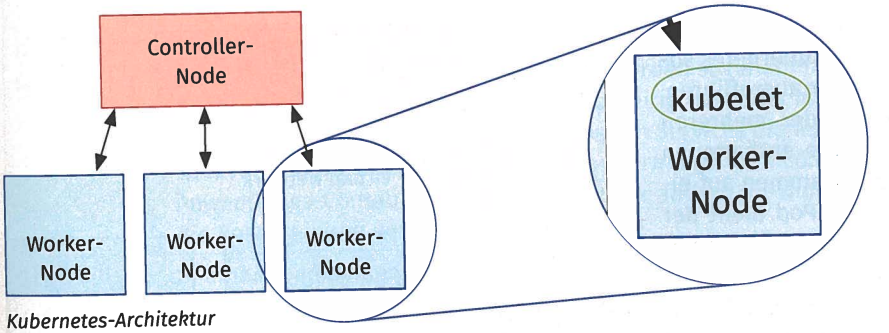
Die Funktionalitäten von Kubernetes können mit drei Begriffen zusammengefasst werden:

Hochverfügbarkeit	Sie ist rein statistisch mit einer Verfügbarkeit von 99,99 % oder besser beschrieben. In der Praxis wird dieser Wert durch einen Cluster von Servern erreicht. Fällt ein Server aus, werden die Container auf einem anderen zur Ausführung gebracht.
Skalierbarkeit	Wenn ein ausgeführter Container eine kritische Marke an Ressourcen erreicht, z. B. CPU-Last, Speichernutzung oder Netzwerklast, wird, je nach vorkonfigurierter Beschreibung, eine Skalierung durchgeführt. Im einfachsten Fall wird ein weiterer Container gestartet und beispielsweise über einen Load Balancing eingebunden.
Wiederherstellung im Fehlerfall	Fällt die Ausführung von Containern aus, werden diese automatisch neu gestartet und der Status der letzten Ausführung wiederhergestellt.

Kubernetes (griech. Steuermann) wurde 2014 als Open-Source-Projekt von Google veröffentlicht. Das Projekt ging 2015 in die von der Linux Foundation neu gegründete Cloud Native Computing Foundation (CNCF) über.

Kubernetes-Architektur

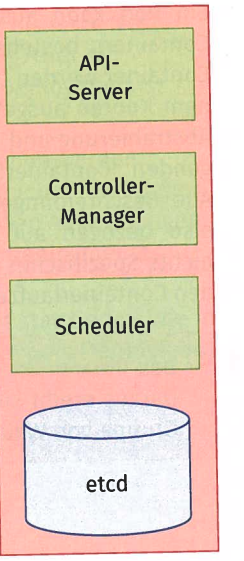
Die nachfolgende Abbildung zeigt die Architektur eines Kubernetes-Clusters. Grundlage des Systems sind Arbeitsknoten (**Nodes**), die zu einem Cluster zusammengefasst werden. Auf jedem Node kommt eine Containerplattform zur Ausführung, wobei Kubernetes unterschiedliche Containerplattformen unterstützt. Neben „Docker" können auch andere Plattformen wie z. B. „containerd" oder „rkt" eingesetzt werden.



Kubernetes-Architektur

Auf jedem Arbeitsknoten kommt ein **Kubelet**-Prozess zur Ausführung, der für die Überwachung und Steuerung des Knotens auf der einen Seite und der Kommunikation mit dem Controller auf der anderen Seite zuständig ist. Die Arbeitsknoten werden von mindestens einem **Controller** verwaltet. In einer Produktivumgebung wird dieser in der Regel ausfallsicher mit zwei oder mehreren Controllern umgesetzt.

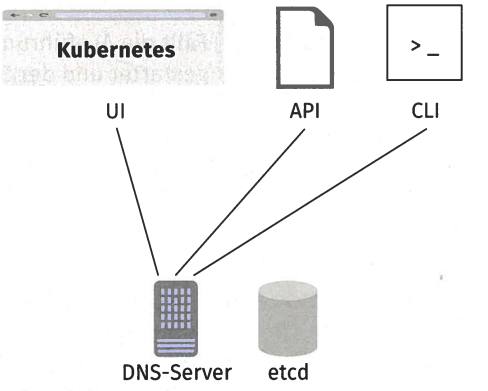
Der Controller enthält die Komponenten „API-Server", „Controller-Manager", „Scheduler" und „etcd".



Komponenten eines Kubernetes-Controllers

API-Server	Der API-Server ist die Schnittstelle zum Cluster und unterteilt sich in drei Zugänge: UI, API und CLI. Die UI (User-Interface) ist ein Webinterface mit dem Dashboard, über das Statusabfragen und die Steuerung des Clusters vorgenommen werden können. Die API (Application Programming Interface) ist ein Skript- oder Programmiersprachenzugang, über den Schritte automatisiert werden können. Das CLI (Command Line Interface), schließlich ist ein Kommandozeilentool (kubectl), über das auf das Cluster zugegriffen werden kann.
Controller-Manager	Der Controller-Manager ist die Oberaufsicht im Cluster. Er fragt die Knoten ab und überprüft die Verfügbarkeit der benötigten Container. Ausgefallene Container werden neu gestartet.
Scheduler	Der Scheduler überwacht die Auslastung der einzelnen Knoten. Er wählt für die Bereitstellung von neuen Containern den Knoten mit den passend verfügbaren Ressourcen aus.
etcd	Speicher, in dem der Status und die Konfiguration des Cluster gespeichert werden.

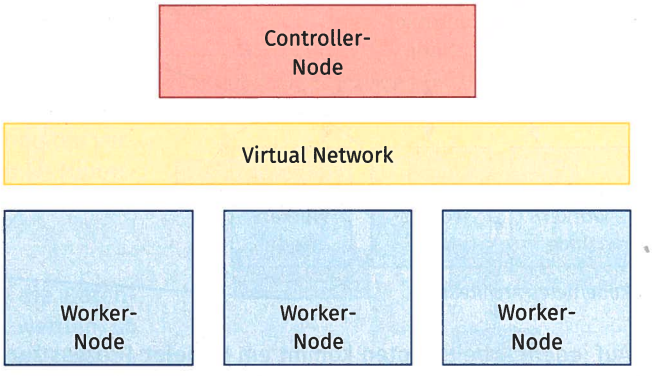
Schließlich bindet das virtuelle Netzwerk (**Virtual Network**) alle Systeme zusammen und lässt den Cluster wie ein monolithisches System erscheinen.



Zugänge zum API-Service bei Kubernetes

Kubernetes-Komponenten

Auf den Arbeitsknoten nutzt Kubernetes sogenannte **Pods**, um Container auszuführen. Ein Pod kann aus einem oder mehreren Containern bestehen und alle enthaltenen Container werden immer zusammen auf einem Knoten ausgeführt. Ein Pod dient der Abstrahierung und erlaubt, die darunterliegenden Containerlaufzeit auszutauschen. Alle Beschreibungen in Kubernetes finden also bezogen auf Pods statt und haben nichts Spezifisches mit der darunterliegenden Containerlaufzeit zu tun.



Aufteilung der Clusterumgebung in Nodes

! Das Docker-Logo (ein Wal, der Container transportiert) wurde in einem Design-Contest ermittelt. Daraus ergibt sich der Begriff **Pod** in Kubernetes, denn es ist die englische Bezeichnung für eine Gruppe von Walen, die im Deutschen auch als Schule bezeichnet wird.

Kubernetes hat weitere Komponenten, die in der folgenden Tabelle erläutert werden:

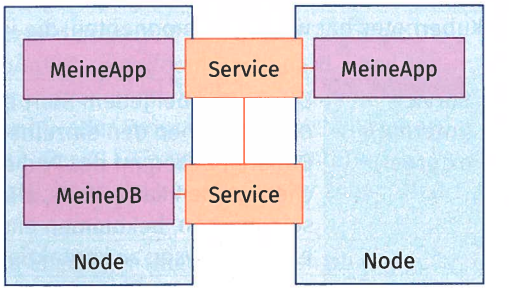
Service (internal/ external)	Da ein Pod bei jedem Start eine neue IP-Adresse erhält, ist es nötig, eine feste Verbindung zwischen den einzelnen Pods zu erreichen. Das wird über Services realisiert. Dadurch bekommt der Pod eine eindeutige IP-Adresse, über die er immer angesprochen werden kann. Pods, z. B. Datenbanken, die nur von anderen Containern erreichbar sein müssen, bekommen einen internen Service. Um den Pod von extern, z. B. von einem Browser, zu kontaktieren, gibt es einen externen Service.
Ingress	Um einem Pod über einen Domänennamen aufzurufen, gibt es den Dienst „Ingress". Dieser leitet die Anfragen, z. B. für einen Domänenname, an den entsprechenden Pod weiter.
ConfigMap	Zur Speicherung der Konfigurationsparameter können Parametervariablen verwendet werden. Diese Parameter sind in eigenen Textdateien gespeichert. Die Container arbeiten nur mit den Variablen. *Beispiel:* `DB_NAME='MeineDB'`. Ändert sich der Name, muss er nur in der ConfigMap geändert werden. Die Container erhalten durch Nutzung der Variablen „DB_NAME" automatisch den richtigen Wert.
Secret	Zur Speicherung von sensiblen Daten z. B. Benutzername oder Passwort, sollte nicht die ConfigMap verwendet werden, da dort alle Informationen im Klartext gespeichert sind. Hierfür gibt es Secrets, in der Daten in base64-Kodierung abgelegt werden. Die Kodierung macht die Informationen allerdings nicht sicher und Kubernetes bietet hierfür auch keine Lösung. Zur Sicherung eines Passworts existieren Tools von Drittanbietern, die eine Verschlüsselung ermöglichen. Die Containeranwendungen führt die Entschlüsselung dann entsprechend aus.
Volume	Ein Volume verbindet nicht-flüchtigen Speicher mit einem Container. Der Speicher ist nicht Teil von Kubernetes, kann aber beispielsweise als Festplatten- oder SSD-Speicher des Nodes oder über das Netzwerk zur Verfügung gestellt werden. Der Container speichert Daten über das Volume und kann so den Status über einen Neustart des Pods behalten. Kubernetes unterstützt keine Datenpersistenz. Diese muss durch die Anwendung sichergestellt werden.
Deployment	Deployment ist eine nächste Abstraktion oberhalb von Pods. Um eine hohe Verfügbarkeit von Anwendungen zu erreichen, werden sogenannte Replikationen einer Anwendung erstellt. Es laufen also gleichzeitig zwei oder mehrere Kopien des Anwendungs-Containers. Die zuvor beschriebenen Services dienen nicht nur als feste Netzwerkverbindung zwischen den Pods, sondern auch zum Load Balancing. Fällt ein Anwendungs-Container aus, übernimmt der andere die Aufgabe. Das Deployment ist die Beschreibung des Systems.
StatefulSet	In einem Deployment kann die Replikation einer Anwendung beschrieben werden, jedoch nicht die einer Datenbank, da eine Datenbank einen Status speichert und daher eine Synchronisation nötig macht. Die Beschreibung von statusbasierten Anwendungen oder Datenbanken wird über StatefulSet beschrieben und nicht über Deployments.

Die Abbildung rechts zeigt ein **Deployment**, in dem eine Replikation der Anwendung stattfindet. Wie in der vorherigen Tabelle unter **StatefulSet** beschrieben, kann eine Datenbank nicht in einem **Deployment** repliziert werden. Daher wird sie an dieser Stelle nicht repliziert.

Die links von der Abbildung angeordnete YAML-Datei beschreibt das Deployment.

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: MeineApp
5    labels:
6      app: MeineApp
7  spec:
8    replicas: 2
9    selector:
10     machtLabels:
11       app: MeineApp
12   template:
13     metadata:
14     labels:
15       app: MeineApp
16     spec:
17       containers:
18         - name: MeineApp
19           image: mein-image
20           env:
21             - name: MeineDB
22               value: $DB_NAME
23           ports:
24             - containerPort: 8080
```

Beispiel für ein Deployment



Beispiel für ein Deployment

Zeile	Erklärung
2	`kind: Deployment` – Deployment ist eine Vorlage, um Pods zu erstellen.
8	Es sollen zwei Replikationen der Anwendung erzeugt werden. Der Name der Anwendung „MeineApp“ wird in Zeile 11 festgelegt.
17	Hier wird der Container für diese Anwendung festgelegt und auf welchem Image er basiert.
20	Hier werden Variablen festgelegt.
23	Hier wird der Port festgelegt, über den der Container erreichbar ist.

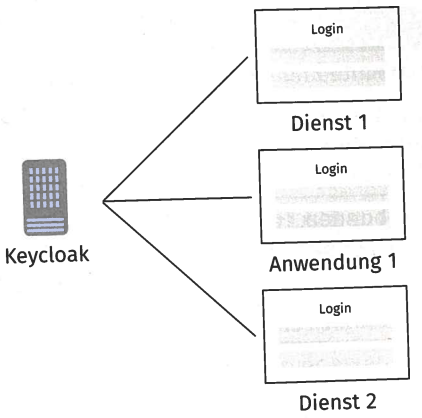
(3) Single-Sign-On mit Keycloak planen

Mit der Vernetzung heterogener Systeme kommt dem systemübergreifenden Identitäts- und Zugriffsmanagement (Identity and Access Management – IAM) eine besondere Rolle zu.

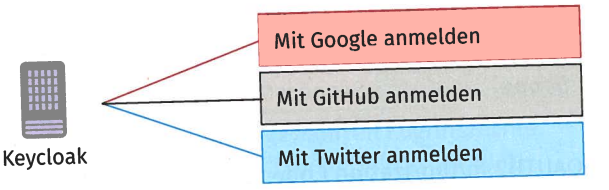
Single-Sign-On/-Off

Unter **Single-Sign-On** (SSO) wird die Funktionalität verstanden, sich mit Zugangsdaten anzumelden und mit dem angemeldeten Zustand auf verschiedene Systeme Zugriff zu haben. Eingeschlossen ist auch das **Single-Sign-Off**. Darunter wird eine Abmeldung von einem System verstanden, die zur Abmeldung von allen anderen zugehörigen Systemen führt.

Ein Open-Source-Projekt für Identitäts- und Zugriffsmanagement ist der **Keycloak-Server** (siehe unter www.keycloak.org). Er stellt verschiedene Funktionen bereit, die eine flexible Verbindung auch bestehender Systeme erlaubt. Zentrale Funktion ist ein **Single-Sign-On/-Off**. Dazu stellt der Server Funktionen zur Verfügung, die in entwickelte Dienste und Anwendungen einfach eingebunden werden können. Jeder Dienst und jede Anwendung leitet die Anmeldung an den Keycloak-Server um, wo die An- oder Abmeldung durchgeführt wird.



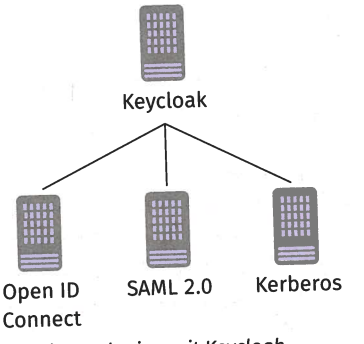
Single-Sign-On/-Off mit Keycloak



Verwendung von externen Logins als Single-Sign-On-Credentials

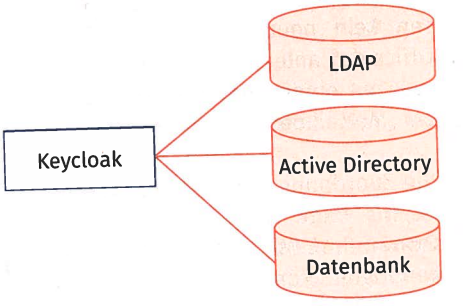
Statt Konten auf dem Keycloak-Server anzulegen, kann Nutzern angeboten werden, sich über ein externes Konto, z. B. bei Google, GitHub, Twitter etc., anzumelden. Neben dem Eingabefeld für Benutzername und Passwort (Credentials) erscheinen dann verlinkte Felder zu den entsprechenden Diensten.

Die Anbindung zu diesen Diensten findet über sogenannte **Identity Broker** statt. „OIDC" (Open ID Connect) ist eine Weiterentwicklung von „OAUTH2" (Open Authorization v2.0) und bietet die Anbindung zu gängigen Diensten wie Google, GitHub oder Twitter. Ältere Dienste, häufig auf Microsoft-Technologie basierend, werden über „SAML 2.0" (Security Assertion Markup Language) angebunden. Ein Broker des Keycloak-Servers wird mit dem **Identity Provider** des jeweiligen Dienstes verbunden. Wenn ein Nutzer sich anmeldet, wird er umgeleitet, meldet sich dort an und wird danach zurück zum Dienst oder der Anwendung geleitet.



Identity Brokering mit Keycloak

Über **User Federation** kann ein LDAP, Active Directory oder eine beliebige relationale Datenbank an den Keycloak-Server angebunden werden. Im Zusammenhang mit dem Keycloak-Server bedeutet das, dass einige oder alle Benutzerdaten von dem angebundenen Dienst in den Keycloak-Server importiert werden. Welche Daten importiert werden, hängt von dem angeschlossenen Dienst ab. Entsprechend kann auch die Überprüfung der Zugangsdaten über den Keycloak-Server oder den angeschlossenen Dienst stattfinden.



User Federation mit Keycloak

OAUTH2

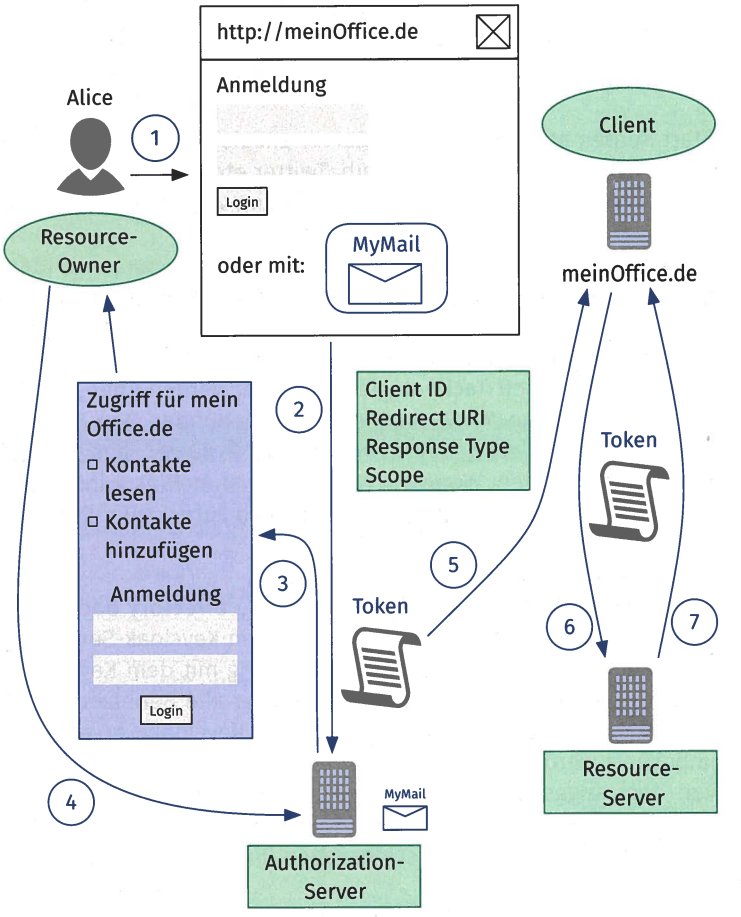
OAUTH2 ist ein Sicherheitsstandard (RFC 6749, RFC 8252, RFC 8996), mit dessen Hilfe ein Single-Sign-On implementiert werden kann. Grundlage von OAUTH2 ist, dass eine Anwendung die Berechtigung erhält, auf Daten einer anderen Anwendung zuzugreifen. Im Folgenden werden Begriffe aus OAUTH2 in der Tabelle beschrieben und in einem exemplarischen Ablauf (siehe Abbildung unten) genutzt.

Wichtige Begriffe zu OAUTH2	
Resource-Owner	Nutzer, dem die Ressourcen gehören und der diese bei einem anderen Dienst (Client) nutzen möchte.

Wichtige Begriffe zu OAUTH2	
Client	Anwendung, die der Resource-Owner nutzen möchte und die auf die Ressourcen des Resource-Owner zugreifen möchte.
Authorization-Server	Hier hat der Resource-Owner ein Benutzerkonto, über das der Zugriff auf seine Ressourcen autorisiert werden kann.
Resource-Server	Der Server ist mit dem Authorization-Server verbunden. Hier werden die eigentlichen Ressourcen gespeichert. In einigen Fällen können Authorization- und Resource-Server zusammengefasst werden.
Scope	Scope ist eine Untermenge der Ressourcen, auf die der Client zugreifen möchte.

OAUTH2 Authorization Code Flow
OAUTH2 beschreibt verschiedene sogenannte Flows. Der am häufigsten verwendete ist der **Authorization Code Flow,** der anhand der gezeigten Abbildung erklärt werden soll. Die grün hinterlegten Informationen sind wichtige Begriffe, die in der Tabelle oben beschrieben sind.

Im Beispiel wird die Nutzerin „Alice" betrachtet. „Alice" hat ein E-Mail-Konto bei „MyMail.de". „Alice" möchte neu den Dienst „mein Office.de" nutzen. Der Anbieter von „meinOffice.de" hat sich mit „MyMail.de" abgesprochen und bietet ein Single-Sign-On an. Die Server wurden entsprechend konfiguriert. Das bedeutet, dass bei der Anmeldung zu „meinOffice.de" ein Link zu „MyMail.de" erscheint. Nutzer müssen kein neues Konto bei „mein Office.de" anlegen, sondern können sich mit einem existierenden Konto bei „MyMail.de" anmelden.



OAUTH2 Authorization Code Flow

Die Zuordnung der OAUTH2-Begriffe ist wie folgt: „Alice" ist **Resource-Owner**. Ihr E-Mail-Konto und die zugehörigen Kontaktdaten sind die **Ressourcen**, die auf einem **Resource-Server** abgelegt werden. Die Benutzerverwaltung ist manchmal auf dem gleichen, oft auch auf einem extra zugehörigen Server abgelegt. Dieser Server übernimmt die Funktion des **Authorization-Server**. Der „meinOffice.de"-Dienst ist der **Client**.

Der Ablauf des Authorization Code Flows wird nach den in der Abbildung gezeigten Nummern folgendermaßen durchgeführt:

Ablauf des Authorization Code Flow	
1	„Alice“ greift auf den neuen Dienst „meinOffice.de“ zu.
2	Über den Link auf der Webseite wird sie zu ihrem E-Mail-Konto beim Dienst „MyMail.de“ umgeleitet. Genauer: Die Umleitung findet zu dem Authorization-Server statt. Mit der Umleitung werden die Parameter „Client ID“, „Redirect URI“, „Response Type“ und „Scope“ mitgeliefert.
3	„Alice“ bekommt eine Liste an Ressourcen angezeigt, die mit dem Scope übermittelt wurden, auf die „meinOffice.de“ zugreifen möchte.
4	„Alice“ wählt aus, auf welche Ressourcen sie den Zugriff erlauben möchte und meldet sich mit Benutzername und Passwort bei „MyMail.de“ an.
5	Der Authorization-Server erzeugt einen Token, in dem die von „Alice“ gewährten Zugriffsrechte enthalten sind. Dieser Token wird an die „Redirect URI“ gesendet, die in dem Fall zurück zum Client, dem „meinOffice.de“-Dienst, gesendet wird.
6	Der Client kann jetzt mit dem Token auf den Resource-Server zugreifen.
7	Der Resource-Server liefert die vom Client angefragten Ressourcen an diesen.

Kompetenzcheck

1. Erläutern Sie das Konzept von Microservices in eigenen Worten.
2. Nennen Sie die drei Begriffe, mit denen die Orchestrierung bei Kubernetes zusammengefasst werden kann.
3. Geben Sie mindestens zwei Komponenten eines Kubernetes-Systems an.
4. Erklären Sie die Funktionen „Single-Sign-On“ und „Single-Sign-Off“ in eigenen Worten.
5. Bearbeiten Sie die Aufgabe 7 der Lernsituation 3 im Arbeitsbuch. A7

1.4 Serverdienste implementieren, testen, überwachen und kritische Zustände beheben

S **Sie nehmen Serverdienste in Betrieb und überwachen diese. Sie ergreifen Maßnahmen, um kritische Zustände zeitnah zu erkennen und zu beheben.**

Nach der Planung der Inbetriebnahme eines neuen Serverdienstes gemäß den Kundenanforderungen, müssen die Mitarbeiter der JIKU IT-Solutions GmbH den neuen Dienst implementieren. Anhand der Planung wird der Dienst zunächst in einer Testumgebung auf virtuellen Maschinen implementiert. Erst wenn alle Konfigurationsparameter fehlerfrei integriert und getestet sind, darf eine Übertragung des Serverdienstes in die Produktivumgebung des Kunden erfolgen.

Die Installation eines Serverdienstes gestaltet sich oftmals einfach. Die Konfiguration, gerade wenn bestimmte Anforderungen beachtet werden müssen, kann komplex werden. Der konfigurierte Dienst muss anschließend überwacht werden, damit die Verfügbarkeit des Systems im Bedarfsfall wiederhergestellt werden kann. Die Planung, das Testen und die Integration von Diensten im Kundenauftrag gehören bei den Auszubildenden zum Fachinformatiker/zur Fachinformatikerin bei JIKU IT-Solutions zu einem beliebten Abschlussprojekt (Kapitel 5.4). In Kapitel 1.4 werden exemplarisch verbreitete Dienste implementiert. Dabei wird auch auf die Protokollierung und das Monitoring bestimmter Dienste und Systeme eingegangen.

1.4.1 Serverdienste implementieren

S **Nach der Planungsphase sollen Sie verschiedene Dienste implementieren.**

(1) DHCP implementieren

Bevor ein DHCP-Server in einer produktiven Umgebung eingesetzt wird, ist es sinnvoll, ihn in einer Testumgebung zu installieren und zu konfigurieren. Für eine Testumgebung bietet sich die Verwendung virtueller Maschinen an. Für die weiteren Beispiele wurde eine virtuelle Maschine mit dem Betriebssystem „Debian 11" verwendet. Das Betriebssystem sollte vor der Installation des DHCP-Servers auf den aktuellen Stand gebracht werden. Für das Beispiel wurde ein ISC-Kea-DHCPv4-Server verwendet.

Im Gegensatz zum älteren ISC-DHCP-Server kann der ISC-Kea-DHCPv4-Server (noch) nicht über das Standard-Repository des Betriebssystems installiert werden. Er kann aus den Quellen kompiliert werden, aber einfacher ist es, das Repository von ISC den Quellen des Betriebssystems hinzuzufügen. Dann kann der Server über die Paketverwaltung des Systems installiert werden. Das folgende Beispiel zeigt die Konfiguration eines Kea-DHCPv4-Servers. Konfiguriert werden die nachfolgenden Parameter:

Konfigurationsparameter	
Adresspool für die dynamische Zuordnung	192.168.178.10–192.168.178.50
Lease-Time	12 Stunden
Renew-Time	6 Stunden
Rebind-Time	9 Stunden
Adressen mit statischer Zuordnung	192.168.178.5, 192.168.178.6
Gateway	192.168.178.1
DNS-Server	192.168.178.1
NTP-Server	192.168.178.5
Schnittstelle des DHCP-Dienstes	eth0

</>
```
"Dhcp4": {

    "valid-lifetime": 43200,  // leases ist 12h
    "renew-timer": 21600,     // nach 6h findet das "renew" statt
    "rebind-timer": 32400,    // nach 9h findet das "rebind" statt

    "interfaces-config": {
         //Der DHCP-Server soll auf dem Interface eth0 den Dienst anbieten
        "interfaces": [ "eth0" ],
    },

    "subnet4": [{
        // automatische Zuordnung
        "subnet": "192.168.178.0/24",
        "pools": [{ "pool": "192.168.178.10 - 192.168.178.50" }],
        "user-context": {
            "comment": "Gäste"
        }
    }],
```

```
    "reservations": [
        // hier werden statische Zuordnungen zugewiesen
        {
          "hw-address": "1a:1b:1c:1d:1e:1f",
          "ip-address": "192.168.178.5"
        },
        {
          "hw-address": "2a:2b:2c:2d:2e:2f",
          "ip-address": "192.168.178.6"
        }
    ],

    "option-data": [
        // Weitere Optionen die konfiguriert werden
        {
          // Gateway
          "name": "routers",
          "data": "192.168.178.1"
        },
        {
          // DNS; in dem Fall im Router enthalten
          "name": "domain-name-servers",
          "data": "192.168.178.1"
        },
        {
          // NTP-Konfiguration
          "name": "ntp-servers",
          "code": 42,
          "csv-format": true,
          "data": "192.168.178.5"
        }
    ]
}
```

Die vom DHCP-Server vergebenen IP-Adressen werden in der Datei „/var/lib/kea/kea-leases4.csv“ hinterlegt.

```
[redacted] /var/lib/kea$ cat kea-leases4.csv
address,hwaddr,client_id,valid_lifetime,expire,subnet_id,fqdn_fwd,fqdn_rev,hostname,state,user_context
192.168.178.10,08:00:27:c0:a1:44,,43200,1660024341,1,0,0,osboxes,0,
```

Auszug aus der kea-leases4.csv-Datei

Die Abbildung zeigt, dass dem System mit der MAC-Adresse „08:00:27:c0:a1:44“ die IP-Adresse „192.168.178.10“ zugewiesen wurde. Der Wert für die Lease-Dauer beträgt wie in der Konfiguration oben beschrieben 43 200 Sekunden (entspricht 12 Stunden).

Standardmäßig protokolliert der DHCP-Server alle Meldungen in das Syslog des Betriebssystems. Damit die DHCP-spezifischen Informationen in eine separate log-Datei geschrieben werden können, muss das Überwachen der Daten in der Konfigurationsdatei angegeben werden. Im Kontext des DHCP-Server-Blocks muss ein „logger“-Block konfiguriert werden.

```
"logger":[{
    "name": "kead-dhcp4",
    "output_options":[
        {
          "output":"/var/log/kea.log"
        }
    ],
          "severity":"INFO"
}]
```

Der ISC-Kea-DHCPv4-Server differenziert in viele Logger, die jeweils unterschiedliche Aspekte protokollieren. Nach Bedarf können diese hinzugefügt werden. Das Schlüsselwort „severity“ gibt an, welche Mel-

dungen protokolliert werden. Mit dem Wert „INFO“ werden auch Warnung und Fehlermeldungen in der log-Datei gespeichert.
Abschließend kann es sinnvoll sein, einen Stabilitätstest durchzuführen. Der ISC-Kea-DHCPv4-Server stellt dazu die Software „perfdhcp“ zur Verfügung. Die Software kann über das Packet „isc-kea-admin“ installiert werden. Das Benchmarking-Werkzeug erzeugt DHCP-Verkehr und überprüft, ob der DHCP-Server auf die Anfragen ordnungsgemäß antworten kann.

Beispiel: Auszug aus der Ergebnisanzeige von „perfdhcp“

```
***Statistics for: DISCOVER-OFFER***
sent packets: 1348656
received packets: 69695
drops: 1278961
drops ratio: 94.8323 %
orphans: 0
rejected leases: 0
non unique addresses: 0
```

Der DHCP-Server konnte auf 1 348 656 DHCP-Discover-Nachrichten nur mit 69 695 DHCP-Offer-Nachrichten antworten. Werden nun der virtuellen Maschine des ISC-Kea-DHCPv4-Servers statt einer CPU drei CPUs zugewiesen, sinkt die Verlustrate deutlich.

Beispiel: Auszug aus der Ergebnisanzeige von „perfdhcp“ nach Zuweisung von drei CPUs

```
***Statistics for: DISCOVER-OFFER***
sent packets: 1322817
received packets: 593468
drops: 729349
drops ratio: 55.136 %
orphans: 0
rejected leases: 0
non unique addresses: 0
```

(2) DNS implementieren

Für die Implementierung des Szenarios werden hier exemplarisch folgende Konfigurationen anhand von Konfigurationsdateien der Linux-DNS-Software „bind9“ beschrieben:
- Zonenkonfiguration für „cluster.jiku.local“ und primäre/sekundäre Konfiguration für die Zone
- Delegation von „jiku.local“ zu „win.jiku.local“ und „cluster.jiku.local“
- Forwarder von „jiku.local“ zum Recursive-Resolver

Zonenkonfiguration für „cluster.jiku.local“ und primäre/sekundäre Konfiguration für die Zone
Hier geht es um den autoritativen DNS-Server für die Unterdomäne „cluster.jiku.local“, der die IP-Adresse „10.0.0.20“ erhält.



DNS-Server cluster.jiku.local

Voraussetzungen sind:
- Linux-Server,
- installierte DNS-Software „bind9“ (sudo apt install bind9),
- konfigurierte IP-Adresse,
- konfigurierter Rechnername und
- DNS-Einstellung des Betriebssystems auf den bind9-Server (/etc/resolv.conf).

Für die Zonenkonfiguration müssen drei Dateien konfiguriert werden. Die erste ist „/etc/bind/named.conf.options“:

```
1 options {
2
3  listen-on port 53 {localhost; 10.0.0.20 };
4  recursion no;
5 };
```

Die drei Einträge bedeuten:

Zeile	Schlüsselwort	Erklärung
3	listen-on	Der Standardport für DNS ist der Port 53. Dieser soll auf die Interfaces „localhost“ und IP-Adresse „10.0.0.20“ gebunden werden.
4	recursion	Wenn diesen Server Anfragen erreichen, die er nicht beantworten kann, dann soll er keine rekursive Abfrage starten. Dafür gibt es rekursive Resolver, die diese Arbeit übernehmen.

Um die Konfiguration zu überprüfen, kann das Kommando „named-checkconf“ verwendet werden:

```
$ sudo named-checkconf /etc/bind/named.conf.options
$ _
```

Wenn die Konfiguration syntaktisch richtig ist, wird es keine Meldung geben.

Als zweite Datei wird die **Zone** für die Unterdomäne „cluster.jiku.local“ in der Datei „/etc/bind/named.conf.local“ erstellt:

```
1 zone "cluster.jiku.local" {
2
3  type master;
4  file "/etc/bind/zones/db.cluster.jiku.local";
5  allow-transfer {10.0.0.21;};
6
7 };
```

Zeile	Schlüsselwort	Erklärung
3	type	Der Server ist ein primärer DNS-Server.
4	file	Pfad und Dateiname der Zonendatei, die im Folgenden noch konfiguriert wird.
5	allow-transfer	Hier wird der Zonentransfer zum Backup-Server mit der IP-Adresse 10.0.0.21 konfiguriert.

Die dritte Datei ist die eigentliche Zonendatei und dafür hat „bind9“ ein Template, das jetzt an die Stelle kopiert wird, die zuvor festgelegt wurde.

```
$ sudo cp /etc/bind/db.local /etc/bind/zones/db.cluster.jiku.local
```

Als Nächstes wird die Zonendatei editiert. Kommentare in der Datei werden mit einem Semikolon eingeleitet. Beachtenswert ist, dass alle Einträge eines Fully-Qualified-Domain-Namens mit einem Punkt enden (siehe Kapitel 1.1.1, Abschnitt [3]). Bei allen vorherigen Konfigurationsdateien wurde kein Punkt eingegeben.

```
 1 ;
 2 ; BIND data file
 3 $TTL    604800
 4 @             IN    SOA       ns01.cluster.jiku.local admin.cluster.jiku.local. (
 5                     3         ; Serial - increment it, everytime you edit this file
 6                     604800    ; Refresh
 7                     86400     ; Retry
 8                     2419200   ; Expire
 9                     604800    ; Min. Cache TTL
10                     )
11 ;
12 @             IN    NS        ns01.cluster.jiku.local.
13               IN    NS        ns02.cluster.jiku.local.
14 ; A records
15 ns01.cluster.jiku.local.    IN    A         10.0.0.20
16 ns02.cluster.jiku.local.    IN    A         10.0.0.21
17 ;
18 kctrl1.cluster.jiku.local.  IN    A         20.0.0.10
19 kctrl2.cluster.jiku.local.  IN    A         20.0.0.12
20 knode1.cluster.jiku.local.  IN    A         20.0.0.20
21 knode2.cluster.jiku.local.  IN    A         20.0.0.21
22 knode3.cluster.jiku.local.  IN    A         20.0.0.22
```

In der Datei „/etc/bind/named.conf.local“ wurde der Domänenname mit „zone "cluster.jiku.local"“ festgelegt. Hier in der Zonendatei kann jetzt mit dem „Klammeraffen“ (@-Zeichen) auf diesen Namen referenziert werden.

Zeile	Schlüsselwort	Erklärung
4	SOA	Start Of Authority (SOA) zeigt an, für welchen Domänen-Namen dieser Server autoritativ ist. Das @ wird durch die Domäne „cluster.jiku.local.“ ersetzt. Der zweite Eintrag „admin.cluster.jiku.local.“ ist eine E-Mail-Adresse, über die der Administrator der Zone erreichbar ist. Um es als E-Mail zu erkennen, muss der Punkt an der linken Stelle durch einen Klammeraffen ersetzt werden: admin@cluster.jiku.local.
3, 9	TTL	Der obere Wert in Zeile 3 „$TTL 604800“ ergibt die maximale, der Wert in Zeile 9 „604800 ; Min. Cache TTL“ (Time To Live) die minimale Zeit, die ein Abfrageergebnis im Cache des abfragenden Resolvers bleibt. Die 604 800 Sekunden sind umgerechnet 7 Tage.
5	Serial	Der Wert Serial muss bei jeder Änderung der Datei erhöht werden. Hierdurch wird dem Server angezeigt, dass die Konfiguration geändert wurde.
6–8	Optionen	Die Optionen bestimmen, wie oft ein Abgleich zwischen primären und sekundären Server stattfindet.
12–13	NS-Records	NS-Records verweisen auf den Namensserver, der für die Auflösung des FQDN zuständig ist. Durch den Klammeraffen geht es hier wieder um „cluster.jiku.local“.

Zeile	Schlüssel-wort	Erklärung
15, 16	A-Records Namensserver	Die bisherigen Konfigurationen sind alle namensbasiert. Wenn es aber zu der Verbindung kommt, ist die IP-Adresse nötig. Diese wird über A-Records für IPv4 erstellt. Für IPv6 werden entsprechende Einträge benötigt. Zur Unterscheidung werden diese als **AAAA-Records** (statt A) bezeichnet.
18–22	A-Records Rechner	Hier werden die Namen der IPv4-Adressen der Rechner konfiguriert. Als Antwort auf ein **Forward-Lookup** für einen der gelisteten Namen, wird die entsprechende IP-Adresse geliefert. Die Einträge sind beispielhaft für fünf Rechnernamen mit entsprechenden IP-Adressen aus dem Kubernetes-Cluster.

Für die Überprüfung der Zonendatei gibt es ebenfalls ein Kommandozeilentool:

```
$ sudo named-checkzone cluster.jiku.local /etc/bind/zones/db.cluster.jiku.local
zone cluster.jiku.local/IN: loaded serial 3
OK
```

Die Überprüfung ergab keinen Fehler. Mit Neustart des Servers wird die Konfiguration aktiviert.

```
$ sudo systemctl restart bind9.service
```

Um die aktuelle Konfiguration zu überprüfen, können die Kommandozeilentools „dig“ oder „nslookup“ verwendet werden.

```
$ nslookup kctrl1.cluster.jiku.local 10.0.0.20
Server: ns01.cluster.jiku.local
Address: 10.0.0.20

Authoritative answer:
Name: kctrl1.cluster.jiku.local
Address: 20.0.0.10
```

Sekundärer Server für „cluster.jiku.local“
Der sekundäre Server ist unter der IP-Adresse „10.0.0.21“ erreichbar. Die Konfiguration wird wie beim gezeigten primären Server – bis auf die Datei „/etc/bind/named.conf.local“ – vorgenommen:

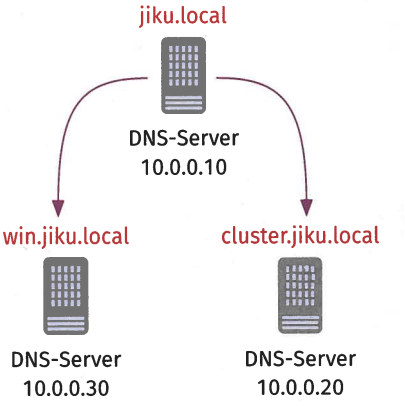
```
1 zone "cluster.jiku.local" {
2
3  type slave;
4  file "/etc/bind/zones/db.cluster.jiku.local";
5  masters {10.0.0.20;};
6 };
```

Zeile	Schlüssel-wort	Erklärung
1	zone	Die Zone ist dieselbe, der Server soll nur als sekundärer für den primären (Master) Server dienen.
3	type	Hier wird der Server zum sekundären (Slave) erklärt.

Zeile	Schlüssel-wort	Erklärung
4	file	Die Datei mit der Konfiguration ist dieselbe wie beim primären Server.
5	masters	Hier werden die IP-Adressen der primären Server eingetragen. Es kann mehrere geben, hier ist nur eine konfiguriert.

Delegation „jiku.local“

Damit der für die Domäne „jiku.local“ autoritative DNS-Server mit der IP-Adresse „10.0.0.10“ Anfragen für die Unterdomänen „win.jiku.local“ und „cluster.jiku.local“ entsprechend weiterleitet, müssen in seiner Zonendatei die **Delegationen** konfiguriert werden. Diese werden mithilfe von NS-Records erreicht. Die Datei „/etc/bind/zones_db.jiku.local“ hat dazu folgende Einträge:



Delegation „jiku.local“

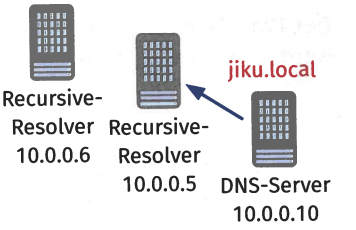
```
1 ; ... Auszug der Datei
2 ;
3 ; Delegationen
4 ;
5 cluster.jiku.local.        IN    NS   ns01.cluster.jiku.local
6 win.jiku.local.            IN    NS   ns01.win.jiku.local
7 ;
8 ; A records
9 ;
10 ns01.cluster.jiku.local.  IN    A    10.0.0.20
11 ns01.win.jiku.local.      IN    A    10.0.0.30
```

Zeile	Schlüsselwort	Erklärung
5, 6	**Delegation**	Hier findet die Delegation der Unterdomäne „cluster.jiku.local.“ auf den Server „ns01.cluster.jiku.local“ und „win.jiku.local.“ auf den Server „ns01.win.jiku.local“ statt.
10, 11	**A-Records** für die Delegation	Da die Namen der Server hier nicht bekannt sind, müssen A-Records für sie erstellt werden, über die schlussendlich die Delegation stattfindet. Bei Einsatz von IPv6 sind die entsprechenden AAAA-Records zu ergänzen.

Forwarder von „jiku.local“ zum Recursive-Resolver

Als letzter Schritt soll die Weiterleitung von dem DNS-Server mit der IP-Adresse „10.0.0.10“ zu dem Recursive-Resolver konfiguriert werden. Dazu wird die Datei „/etc/bind/named.conf.options“ des Servers editiert:

```
options {
 forwarders {
 10.0.0.5; // Primärer Recursive-Resolver
 10.0.0.6; // Sekundärer Recursive-Resolver
 };
};
```



Weiterleitung zum Recursive-Resolver

(3) NTP implementieren

Die Version 4 ist die aktuelle Version von NTP (NTPv4 wird in RFC 5905 und das ältere NTPv3 in RFC 1119 beschrieben). Diese Version unterstützt sowohl IPv4 als auch IPv6. Die Installation eines NTP-Servers unter Ubuntu kann mit dem folgenden Befehl erfolgen.

```
sudo apt install ntp
```

Während der Installation wird das NTP-Client-Paket „systemd-timesyncd" (reiner Client) deinstalliert, damit es nicht zu Konflikten kommt. Das Paket „ntp" enthält sowohl Server- als auch Client-Komponenten, da jeder Server, der nicht auf Stratum 0 betrieben wird, selbst auch als Client fungiert.

Die Konfigurationsdatei enthält bereits einige Vorgaben, die im Wesentlichen übernommen werden können. Die Parameter „pool" bzw. „server" sollten überprüft werden. Mit dem Befehl „ntpq -p" kann die Qualität der konfigurierten NTP-Server getestet werden.

```
# /etc/ntp.conf
# Speichert die Frequenzabweichung der internen Uhr von der Atomuhr-Zeit;
# Ordner muss für ntpd schreibberechtigt sein
driftfile /var/lib/ntp/ntp.drift

# Definition der Schaltsekunden die über Zeitzonendatenbank bereitgestellt werden
leapfile /usr/share/zoneinfo/leap-seconds.list

# Ordner in dem die Statistik der lokalen Synchronisationperformance gespeichert wird.
# Kann zu Testzwecken sinnvoll sein
statsdir /var/log/ntpstats/

# Zeitserver auf Stratum 1 Ebene als Pool oder als einzelner Server angeben
pool 0.ubuntu.pool.ntp.org iburst
server ptbtime1.ptb.de iburst

# Server vor ungewollten Anfragen schützen
restrict -4 default kod notrap nomodify nopeer noquery limited
restrict -6 default kod notrap nomodify nopeer noquery limited

# Interne Zugriffe werden erlaubt (vgl. default Parameter)
restrict 127.0.0.1
restrict ::1

# Erlaubt dem Netz 192.168.2.0/24 Zeit- und Statistikanfragen zu stellen
restrict 192.168.2.0 mask 255.255.255.0 nomodify notrap nopeer

# Erlaubt einer Host-Adresse 192.168.0.10/32 uneingeschränkte Anfragen zu stellen
restrict 192.168.0.10
```

Der Parameter „restrict“ kann zur Sicherung des NTP-Servers vor Angriffe wie DDoS-Attacken genutzt werden. Über diesen Parameter können ebenfalls Netzsegmente definiert werden, die für Anfragen zugelassen bzw. verboten sind.

Restrict-Parameter zur Steuerung der Zugriffsrechte	
Parameter	**Beschreibung**
-4 bzw. -6	„restrict“ bezieht sich auf IPv4 bzw. IPv6
default	setzt die Standardkonfiguration und verhindert damit alles, was nicht im Weiteren mittels „restrict“ erlaubt wird
kod	„kiss-of-death“-Paket wird gesendet, um ungewollte Anfragen zu reduzieren
notrap	unterdrückt „ntpdc“-Control-Message-Protocol-Nachrichten
nomodify	verhindert Konfigurationsänderungen
nopeer	unterdrückt Verbindungsaufbau zu anderen NTP-Servern
noquery	unterdrückt „ntpq“- und „ntpdc“-Anfragen, ohne Zeitanfragen zu verhindern
limited	verhindert missbräuchliche Anfragen von Clients, z. B. zu häufige Anfragen

Testen des NTP-Servers als Client

Es gibt unter Linux drei Möglichkeiten, den NTP-Dienst sowohl auf normalen PCs (reiner NTP-Client) wie auch auf dem eigentlichen NTP-Server zu testen. (Hinweis: In Richtung Internet arbeitet der Server als NTP-Client.)

Programmaufruf	Ausgabe
timedatectl timesync-status	`Server: 185.125.190.56 (ntp.ubuntu.com)` `...` `System clock synchronized: yes` `NTP service: active`
ntpq -n	`osboxes@osboxes:/etc$ ntpq -pn` `remote refid st t when poll reach delay offset jitter` `==========` `0.ubuntu.pool.n .POOL. 16 p - 64 0 0.000 +0,000 0,000` `1.ubuntu.pool.n .POOL. 16 p - 64 0 0.000 +0,000 0,000` `2.ubuntu.pool.n .POOL. 16 p - 64 0 0.000 +0,000 0,000` `3.ubuntu.pool.n .POOL. 16 p - 64 0 0.000 +0,000 0,000` `ntp.ubuntu.com .POOL. 16 p - 64 0 0.000 +0,000 0,000` `+188.68.36.203 131.188.3.221 2 u 69 256 377 10.527 -2,541 0,291` `+79.133.44.142 .MBGh. 1 u 139 256 377 7.807 -2,171 0,602` `+185.125.190.57 17.253.34.123 2 u 169 256 377 20.756 -1,829 0,497` `+31.209.85.242 .GPS. 1 u 198 256 377 14.224 -1,948 0,550` `+185.125.190.58 17.253.34.125 2 u 193 256 377 20.551 -2,554 0,409`
ntpstat	`synchronised to NTP server (79.133.44.142) at stratum 2` `time correct to within 21 ms` `polling server every 256s`

NTP-Server nach Neustart starten

Damit der NTP-Server nach dem nächsten Neustart automatisch gestartet wird, muss überprüft werden, ob in der Datei „/etc/default/ntp“ der korrekte Aufrufparameter „NTPD_OPTS='-g'“ vorhanden ist. Der Parameter „-g“ unterdrückt die Überprüfung des rechnereigenen Uhrchips. Die Überprüfung könnte bei zu großen Zeitabweichungen zum Stoppen des „ntpd“-Dienstes führen. Falls man den Dienst nur auf IPv4 betreiben möchte, kann „NTPD_OPTS='-4 -g'“ verwendet werden. Hier sind weitere Optionen möglich.

NTP-Client konfigurieren

Zunächst sollte überprüft werden, ob NTP bereits aktiviert wurde. Dies kann unter Linux mit den Befehlen „timedatectl“, „ntpq“ bzw. „ntpstat“, wie oben beschrieben, erfolgen. Unter Windows wird beim Ein-

tritt eines Rechners in eine Domäne die Zeitsynchronisation automatisch konfiguriert. Sollte eine manuelle Nachkonfiguration nötig sein, kann dies mit dem Befehl „W32tm.exe“ (Hilfe mit „/?“) durchgeführt werden.

(4) Webserver implementieren

Für die beispielhafte Implementierung eines Webservers wird zunächst eine neue virtuelle Maschine erstellt und eine Linux-Distribution, z. B. „Debian“ oder „Red Hat Linux“, die von „nginx“ unterstützt wird, installiert. Zu Testzwecken kann eine vorbereitete virtuelle Maschine von der Webseite „Osboxes“ (www.osboxes.org) heruntergeladen werden. In einer Produktivumgebung sollte das System allerdings selbst installiert werden, damit keine versteckten Abhängigkeiten, Hintertüren oder überflüssige Software mit betrieben werden.

Für die weiteren Beispiele wurde eine virtuelle Maschine mit der Distribution „Debian“ verwendet. Das Betriebssystem sollte vor der Installation der Webserver-Software auf den aktuellen Stand gebracht werden. Im Anschluss kann der Webserver installiert werden. Hierzu wird zunächst das Repository der „nginx“-Pakete dem System hinzugefügt. Im Anschluss wird der Webserver über das neue Repository installiert.

Der Webserver ist nach einem ersten Start mit „sudo nginx“ verfügbar, aber noch nicht konfiguriert. Es steht ein HTTP-Server zur Verfügung, der über Port 80 unverschlüsselte Verbindungen aufbauen kann. Die Konfiguration findet über die Datei „nginx.conf“ im Verzeichnis „/etc/nginx.conf“ statt. In der Konfigurationsdatei wird mit Blöcken gearbeitet, die durch geschweifte Klammern { } gekennzeichnet sind. Diese Blöcke sind ineinander verschachtelbar.

Beispiel: Mögliche Minimalkonfiguration in der Datei „nginx.conf“

```
1 http {
2     server {
3       listen                443 ssl;
4       server_name           myservice.intern;
5       ssl_certificate       myservice.intern.crt;
6       ssl_certificate_key   myservice.intern.key;
7       ssl_protocols         TLSv1 TLSv1.1 TLSv1.2;
8       ssl_ciphers           HIGH:!aNULL:!MD5;
9     }
10
11    location / {
12      root /var/www
13    }
14
15 }
```

In der ersten Zeile wird der **Konfigurationsblock** für den HTTP/S-Server geöffnet. Der verschachtelte „server“-Block (Zeilen 3–8) wird Kontext genannt. Dessen Inhalt bezieht sich immer auf den übergeordneten Block (hier „http“). In diesem Block wird der eigentliche HTTPS-Server konfiguriert. Dazu wird der Port festgelegt, auf dem der Server betrieben wird (Zeile 3). In den Zeilen 5 bis 8 werden die Parameter für die Verschlüsselung bestimmt. Die „.crt“-Datei enthält das eigentliche Zertifikat. Die „.key“-Datei enthält den zum Zertifikat zugehörigen privaten Schlüssel. Im Kontext „location“ (Zeilen 11 bis 13) wird der Pfad zum „document-root“ angegeben. Dieses Verzeichnis dient dem Webserver als Wurzelverzeichnis zum Ausliefern der Daten.

Bei einem öffentlich erreichbaren Server gibt es die Möglichkeit, kostenlose Zertifikate von „Let's Encrypt“ zu beziehen. Dazu kann die Software „Certbot“ verwendet werden. Durch diese Software lassen sich automatisch „Let's Encrypt“-Zertifikate verwenden. Die Software „Certbot“ wird unter dem Betriebssystem „Debian“ als „snap“-Paket installiert. Dazu sind die folgenden Befehle nötig:

```
1 sudo snap install --classic certbot
2 sudo ln -s /snap/bin/certbot /usr/bin/certbot
3 sudo certbot --nginx
```

Der erste Befehl installiert die Software „certbot" über das „snap"-Container-Paket. In der zweiten Zeile wird ein Link im Linux-System gesetzt, um den Befehl „certbot" ausführen zu können. Mit der letzten Zeile wird die Software „certbot" für den Webserver „nginx" gestartet.

Die zentrale Kundenanforderung ist die Bereitstellung eines Webservices. Dieser bedingt im Normalfall eine Programmier- oder Skriptsprache. An den Webserver „nginx" können verschiedene Programmier- oder Skriptsprachen angebunden werden, u. a. auch die beliebte Programmiersprache „Python".

```
1 sudo apt install python3 python3-pip
2 sudo apt install fcgiwrap
3 sudo cp /usr/share/doc/fcgiwrap/examples/nginx.conf /etc/nginx/fcgiwrap.conf
4 sudo mkdir /usr/lib/cgi-bin -p
5 sudo nano /etc/nginx/nginx.conf
6 sudo nginx -s reload
```

Durch den Befehl in Zeile 1 werden die nötigen Pakete für Python installiert. Anschließend wird das Paket „fcgiwrap" installiert. Es stellt das CGI-Gateway (Common Gateway Interface) zur Verfügung. In Zeile 3 wird die Vorlage der Konfigurationsdatei für das CGI-Gateway in den Konfigurationsordner des Webservers kopiert. Mit dem Befehl „mkdir" wird ein Verzeichnis angelegt. Wird der Parameter „-p" angegeben, werden fehlende übergeordnete Verzeichnisse ebenfalls erstellt. Anschließend muss in der Konfigurationsdatei des Webservers auf die Datei „fcgiwrap.conf" verwiesen werden. Dazu der Befehl „include fcgiwrap.conf" im Kontextblock „server" der Datei „/etc/nginx/nginx.conf" eingefügt.

Zur Überprüfung kann im Verzeichnis „/usr/lib/cgi-bin" ein kleines ausführbares Python-Skript „jiku-test.py" abgelegt werden.

Beispiel: Testskript

```
#!/usr/bin/python3
print('Content-Type: text/plain')
print('Das ist ein Test.')
```

Über den Webbrowser wird das Skript unter der Adresse „https://<IP-Adresse>/cgi-bin/jiku-test.py" ausgeführt. Mit dem Befehl „chmod 755 /usr/lib/cgi-bin/jiku-test.py" werden die Rechte zum Ausführen auf Dateiebene gesetzt („chmod" = „change mode").

Im letzten Schritt muss eine grundlegende Protokollierung des Webservers überprüft werden. Der Webserver „nginx" unterscheidet wie die meisten Webserver die Protokollierung von Fehlern, dem sogenannten „Error Log" und dem „Access Log", mit dem die Zugriffe auf den Webserver überwacht werden. Beide Dateien finden sich unter „Debian" im Pfad „/var/log/nginx". Der installierte Webserver soll die Kundenanforderungen erfüllen. Eine Anforderung kann die Anzahl möglicher aktiver Verbindungen sein, die der Webserver parallel halten und bearbeiten kann.

Das Programm „ApacheBench" kann einen Lasttest für einen Webserver durchführen. Trotz des Namenbezugs auf den Apache-Webserver arbeitet es mit allen HTTP/S-Webservern zusammen. Es kann beispielsweise unter der Distribution „Debian" mit dem Befehl „sudo apt install apache2-utils" installiert werden. Mit dem Befehl „ab -n 5000 -c 200 https://www.example.com/index.html" werden insgesamt 5000 Verbindungen („number") zur Datei „index.html" auf der Webseite „example.com" aufgebaut. Dabei werden maximal 200 Verbindungsanfragen („concurrent") gleichzeitig gestellt.

Beispiel: Gekürzte Ausgabe des Befehls „ab“

```
…
Concurrency Level: 200
Time taken for tests: 57.962 seconds
Complete requests: 5000
Failed requests: 0
Transfer rate: 135.63 [Kbytes/sec] received
…
Percentage of the requests served within a certain time (ms)
50% 1788
66% 2352
75% 2708
80% 3035
90% 4020
95% 4943
98% 7981
99% 8727
100% 17282 (longest request)
```

Die Ausgabe des Befehls zeigt, dass der Server mit der Anzahl der gleichzeitigen Verbindung umgehen kann. Es kommt zu keiner fehlgeschlagenen Verbindung („Failed requests: 0“). Der Server antwortet mit einer Datenrate von etwa 135 KBytes/sec. Zusätzlich wird die Zeit (in ms) angezeigt, die der Server benötigt, um zu antworten.

Durch das sukzessive Erhöhen des Wertes des Parameters „-c“ kann der Server an seine Lastgrenzen gebracht werden. Durch ein Erhöhen des Wertes der Gesamtverbindungen mit „-n“ ist es möglich, das Verhalten des Servers über einen längeren Zeitraum zu beobachten.

(6) E-Mail-Server implementieren

Für die Installation des E-Mails-Servers „postfix“ in einer virtuellen Maschine wird zunächst eine neue virtuelle Maschine erstellt und eine Linux-Distribution, z. B. „Debian“ oder „Red Hat Linux“, das von „postfix“ unterstützt wird, installiert. Das Betriebssystem sollte vor der Installation weiterer Software auf den aktuellen Stand gebracht werden. Ein E-Mail-Server sollte eine feste IP-Adresse verwenden. Die Adresse wird im Betriebssystem mit dem Werkzeug „netplan“ in der Datei „/etc/netplan/50-cloud-init.yaml“ konfiguriert. Mit dem Befehl „ip a“ kann die IP-Konfiguration des Systems überprüft werden. Der „postfix“-Server ist im Standard-Repository von „Debian“ enthalten und kann direkt installiert werden.

Eine Anforderung ist die verschlüsselte Kommunikation mittels TLS (Transport Layer Security). Liegen keine Zertifikate vor, die von einer öffentlichen Zertifizierungsstelle signiert sind, kann zu Testzwecken ein Zertifikat selbst signiert werden. Dieses wird aber von anderen Systemen zunächst nicht als vertrauenswürdig eingestuft.

Ablauf:

```
1 openssl genrsa -des3 -out mycert.key 2048
2 openssl req -new mycert.key -out mycert.csr
3 openssl x509 -req -days 365 -in mycert.csr -signkey mycert.key -out
mycert.crt
4 openssl req -new -x509 -extensions va_ca -keyout cakey.pem -out cacert.
pem -days 365
```

Im ersten Schritt wird auf dem Server ein privater Schlüssel generiert (Zeile 1). Der private Schlüssel wird in der Datei „mycert.key“ abgespeichert. Anschließend wird aus dem privaten Schlüssel ein von diesem

Schlüssel abhängiger Certificate Signing Request (CSR) erstellt (Zeile 2). In der dritten Zeile wird ein selbstsigniertes Zertifikat erstellt. Dazu wird der CSR mit dem eigenen privaten Schlüssel signiert. Abschließend werden in Zeile 4 die Stammzertifikate der eigenen CA erstellt. Die entstehenden Schlüssel müssen in die richtigen Verzeichnisse verschoben werden. Unter „Debian" sind das beispielsweise die Verzeichnisse „/etc/ssl/private" für die privaten Schlüssel und „/etc/ssl/certs" für die Zertifikate. Anschließend muss die Konfigurationsdatei der „postfix"-Software angepasst werden. Dabei müssen vor allem die Pfade zu den Zertifikatsdateien richtig gesetzt und weitere SSL-Parameter ergänzt werden. Nach der Konfiguration muss der Server neu gestartet werden.

Der Mail Transfer Agent (MTA) „postfix" kann nun zum Senden von E-Mails verwendet werden. Zum Abrufen von E-Mails mittel IMAP wird ein Mail Delivery Agent (MDA) benötigt. Ein bekannter MDA ist die Software „dovecot". Die Software kann ebenfalls über das Standard-Repository installiert werden.

Bei der Software „postfix" findet die Kofiguration in einer Datei statt, bei „dovecot" hingegen ist die Konfiguration, je nach Konfigurationsparameter, auf mehrere Dateien verteilt. Bei der Konfiguration von „dovecot" müssen ebenfalls SSL-Parameter in der Datei „10-ssl.conf" angepasst werden. Zusätzlich muss noch angegeben werden, dass „postfix" als E-Mail-Server verwendet wird. Dazu wird die Datei „10-master.conf" bearbeitet. Außerdem muss das Verzeichnis angegeben werden, in dem die E-Mails der Benutzer einsortiert werden. Dies wird in der Datei „10-mail.conf" angegeben. Nach der Konfiguration muss der „dovecot"-Server neu gestartet werden.



Eintreffen einer E-Mail

Testweise können auf dem Server mit dem Befehl „adduser" zwei neue Benutzer erzeugt werden. Von einer Client-VM lässt sich mittels Thunderbird auf den E-Mail-Server zugreifen. Dazu werden die beiden Benutzer in Thunderbird eingerichtet und das selbstsignierte Zertifikat des Servers bestätigt.

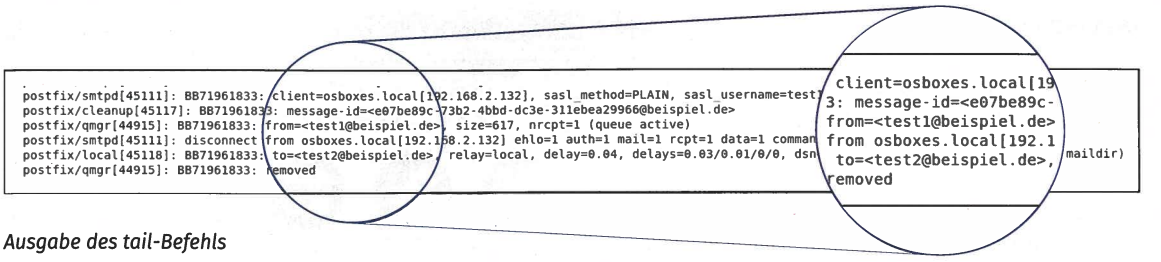
Bei der Konfiguration der Software wurde das Verzeichnis angegeben, in dem die Mailbox eines Benutzers auf dem E-Mail-Server abgelegt wird. Dieses Verzeichnis kann auch manuell auf E-Mails geprüft werden und neue E-Mails können angezeigt werden.

```
[redacted] /home/test2/Maildir/cur# cat 1659941502.V804I380016M917642.osboxes\:2\,
Return-Path: <test1@beispiel.de>
X-Original-To: test2@beispiel.de
Delivered-To: test2@beispiel.de
Received: from [192.168.2.132] (osboxes.local [192.168.2.132])
        by beispiel.de (Postfix) with ESMTPA id DB5FC61834
        for <test2@beispiel.de>; Mon,  8 Aug 2022 02:51:42 -0400 (EDT)
Message-ID: <7e10d115-13e9-394e-4e8e-a4e29463733e@beispiel.de>
Date: Mon, 8 Aug 2022 02:51:42 -0400
MIME-Version: 1.0
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101
 Thunderbird/91.1.2
Content-Language: en-US
To: test2@beispiel.de
From: Test1 <test1@beispiel.de>
Subject: TEST-Betreff
Content-Type: text/plain; charset=UTF-8; format=flowed
Content-Transfer-Encoding: 7bit

TEST - Text in der E-Mail
```

Manuell ausgelesene E-Mail

Die log-Datei des Mail-Servers wird unter „/var/log/mail.log" geführt. Mit dem Befehl „tail /var/log/mail.log" können die Einträge der log-Datei zur Laufzeit angezeigt werden.



Ausgabe des tail-Befehls

Hier wird der Versand und der Empfang einer E-Mail von Benutzer „test1@beispiel.de“ zum Benutzer „test2@beispiel.de“ dargestellt.

(7) WebRTC implementieren

Für die Installation eines BigBlueButton-Servers (BBB-Servers) ist die empfohlene BBB- und Linux-Version zu nutzen (im Folgenden beispielhaft für BBB 2.5 und Ubuntu 20.04). Es wird ein Installationsskript angeboten, welches alle notwendigen Installationsschritte übernimmt. Zunächst muss das Installationsskript „bbb-install.sh“ selbst auf das Serversystem heruntergeladen und ausführbar gemacht werden.

```
wget https://ubuntu.bigbluebutton.org/bbb-install-2.5.sh
chmod +x bbb-install-2.5.sh
```

Im Anschluss kann der BBB-Server installiert werden. Hierzu werden die in der Tabelle beschriebenen Parameter an das Installationsskript „bbb-install.sh“ übergeben.

„bbb-install.sh“-Parameter	
Parameter	**Erklärung**
-v focal-250	Angabe der Versionen von Ubuntu (hier: 20.04) und BBB (hier: Version 2.5)
-s	Hostname des Servers (z. B. „bbb.local“); muss ersetzt werden durch den tatsächlichen Hostname
-e	E-Mail-Adresse für „Let’s Encrypt“ (z. B. „bbb@beispiel.de“)
-a	zur Installation der API-Demos; diese sollten nur zu Testzwecken installiert werden; Deinstallation mit „apt purge bbb-demo“
-w	zur Installation der Firewall
-g	zur Installation von „Greenlight“; hiermit wird ein Meeting-Raum zur Verwaltung angeboten

Der Parameter „-a“ zur Installation der API-Demos sollte nur zu Testzwecken genutzt werden. Die Demos ermöglichen jedem Nutzer den uneingeschränkten Zugriff auf den BBB-Server. Der Parameter stellt damit ein potenzielles Sicherheitsrisiko dar. Im Anschluss an die grundlegenden Tests sollten die API-Demos mit dem Befehl „apt purge bbb-demo“ unbedingt deinstalliert werden (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.3.1 und Kapitel 4.5.2).

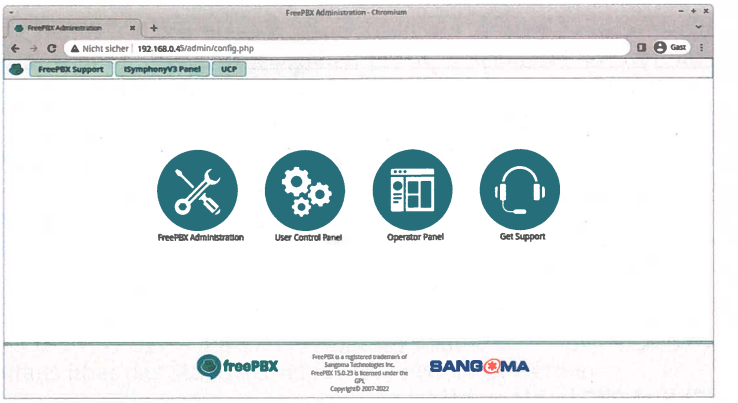
Die Firewall kann mittels Uncomplicated Firewall (UFW) mit den folgenden Befehlen konfiguriert werden. Gegebenenfalls muss das UFW-Paket zunächst mit „sudo apt install -y ufw“ installiert werden. Sofern „ssh“ nicht benötigt wird, kann die entsprechende Befehlszeile entfallen.

```
sudo ufw allow "Nginx Full"
sudo ufw allow 16384:32768/ud
# optional
sudo ufw allow OpenSSH
ufw --force enable
```

(8) VoIP implementieren

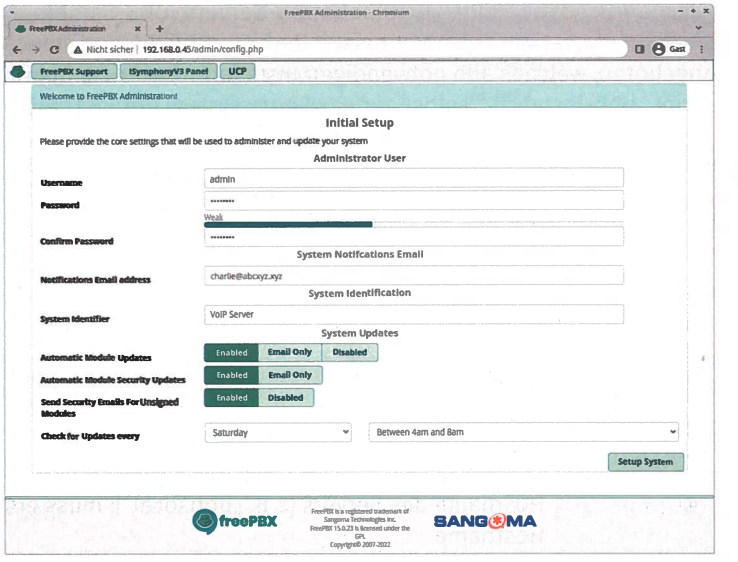
FreePBX implementieren

Die FreePBX ist eine auf Asterisk basierende Distribution, die weitgehend über ein Webfrontend konfiguriert werden kann. Die Installation der FreePBX-Distribution wird als ISO-Image zum Download angeboten. Für den produktiven Einsatz sollte eine STABLE-Version genutzt werden.



FreePBX-Webfrontend

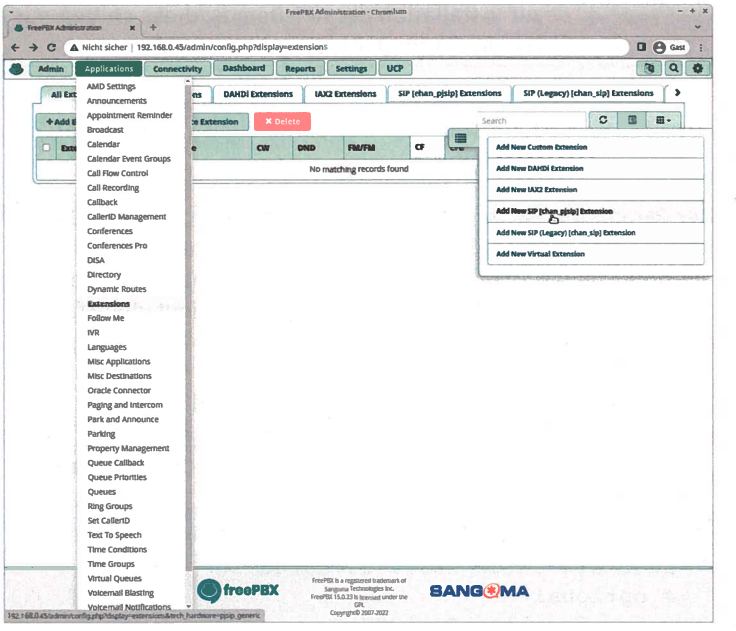
Die vorgegebenen Standardeinstellungen können übernommen werden. Während der Installation muss das Passwort für den Benutzer „root“ festgelegt werden. Dieser wird für den Login am Terminal benötigt. Das Passwort sollte sehr sicher gewählt werden. Nach erfolgreicher Installation kann man den Benutzer „root“ im Terminal anmelden und mit dem Befehl „ip addr“ die IP-Adresse des Servers ermitteln.



FreePBX-Ersteinrichtung

Der FreePBX-Server ist nun per Browser erreichbar. Es müssen noch der Administrator-Benutzer und ein Passwort für das Webfrontend festgelegt werden. Der Dialog hierfür wird über „FreePBX Administration“ aufgerufen. Im Anschluss wird die Firewall aktiviert.

Die Nebenstellen bzw. Teilnehmer werden als sogenannte Extensions über den Menüpunkt Applications → Extensions eingerichtet. Unter „Add Extension PJSIP“ werden die „User Extension“ und der „Display Name“ festgelegt. Die User Extension stellt dabei die Rufnummer und der „Display Name“ den Anzeigenamen des Teilnehmers dar. Das Passwort wird als „Secret“ angegeben. Die User-Manager-Settings sind nur für zusätzliche Anwendungen wie das User Control Panel (UCP) relevant. Zu Testzwecken sollten mindestens zwei Teilnehmer angelegt werden.



Neue Nebenstelle anlegen

Auf der Clientseite kann beispielsweise die Open-Source-Software

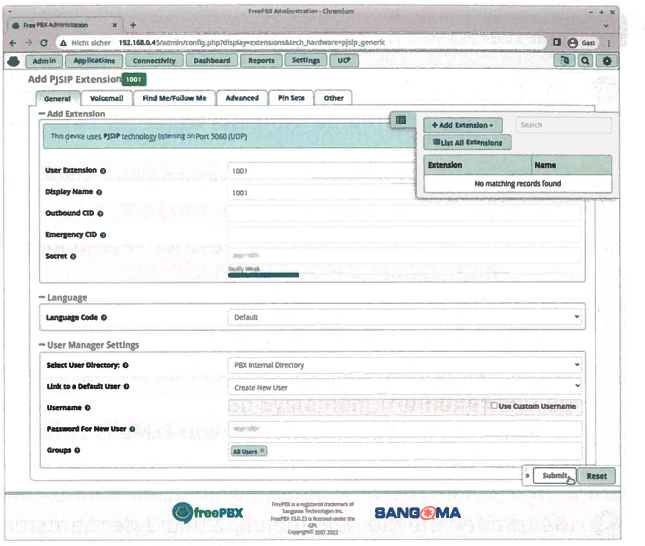
„Linphone“ eingesetzt werden. Diese ist auf fast allen Betriebssystemen verfügbar. In den Einstellungen wird die User Extension im Feld „Ihre SIP-Identität“ eingegeben (z. B. „1001@192.168.0.45“). Als „SIP-Proxy-Adresse“ wird entweder der FQDN oder die IP-Adresse des Servers eingetragen. Sofern vom Standard-Port 5060 abgewichen wird, ist dieser mittels „:“ ebenfalls anzugeben, z. B. „<sip:192.168.0.45:5070>“. Als „Übertragung“ wird UDP ausgewählt. Kommt SSL zum Einsatz, so ist hier „TLS“ auszuwählen.

Die korrekte Anmeldung kann unter „Reports → Asterisk Info“ überprüft werden. Im Bereich „Channels“ sollte als Ressource nun der Teilnehmer mit dem Status „ONLINE“ angezeigt werden. Weitere Informationen erhält man über den Eintrag „Peers“ an der rechten Seite. Im Bereich „PJSIP“ werden detaillierte Informationen zu den Teilnehmern angezeigt. Telefonate werden hier als „in use“ bzw. mit „up“ gekennzeichnet. Während des Verbindungsaufbaus wird „Ringing“ beim angerufenen Teilnehmer angezeigt.

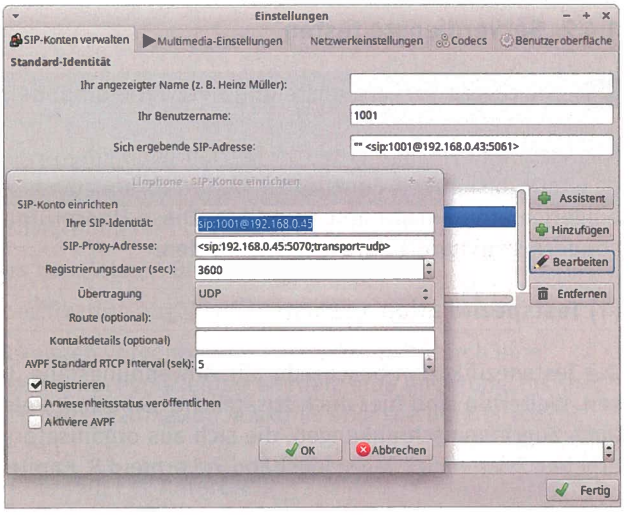
Der lokale Adressbereich (hier: 192.168.0.0) sollte zu Testzwecken für das Sicherheitstool „fail2ban“ deaktiviert werden. Ansonsten kann es dazu führen, dass bestimmte IP-Adressen geblockt werden, wenn zu viele oder fehlerhafte Anfragen am FreePBX-Server eingehen. Wenn das System einwandfrei funktioniert, kann der lokale Adressbereich wieder entfernt werden.

```
nano /etc/fail2ban/jail.local

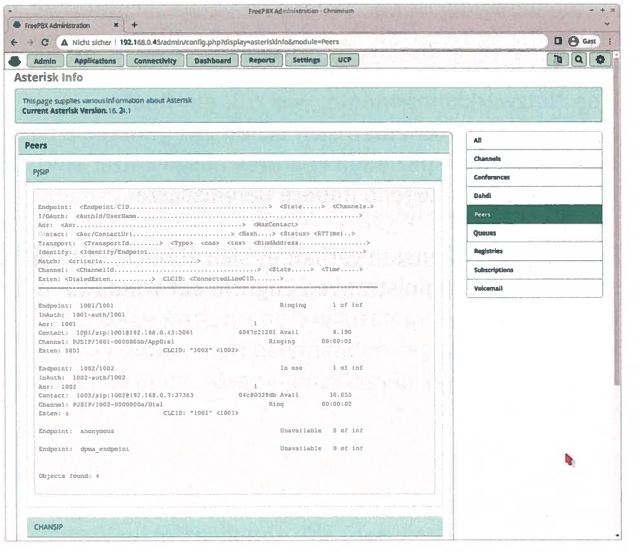
[DEFAULT]
ignoreip = 127.0.0.1
192.168.0.0
```



Detailansicht Nebenstelle



Konfiguration Linphone



Asterisk-Protokollierung der Telefonate

Kompetenzcheck ✓

1. Berechnen Sie für eine Lease-Time von fünf Tagen. Berechnen Sie die Konfigurationsparameter renew-Time und rebind-Time als Werte für eine Konfigurationsdatei.

2. Beschreiben Sie den Unterschied zwischen einer Delegation und einem Forwarder bei DNS.

3. Erklären Sie die Funktion des „restrict"-Parameters einer NTP-Konfiguration in eigenen Worten.

4. Geben Sie an, welche Aussagen richtig sind.
   a) Die Pakete „nginx", „cherokee", „apache" und „lighttpd" sind Webserver-Pakete.
   b) Mit dem Befehl „systemcontrol status nginx" kann geprüft werden, ob der Webserver fehlerfrei bereitsteht.
   c) Ein MDA wird zum Transport von E-Mails zwischen zwei Servern verwendet.
   d) Eine VoIP-Telekommunikationsanlage verwaltet die Zugangsdaten der Nebenstellen.

5. Bearbeiten Sie die Aufgaben 1, 2 und 3 der Lernsituation 4 im Arbeitsbuch.

A1
A2
A3

1.4.2 Serverdienste testen

S **Sie testen Serverdienste und Netzwerkkomponenten.**

Nachdem alle Komponenten in Betrieb genommen wurden, muss im Anschluss sichergestellt werden, dass alle bisherigen Funktionen weiterhin unverändert genutzt werden können. Zudem sind die neu eingerichteten Funktionen zu prüfen. Diese Überprüfungen sollten nicht nur stichpunktartig erfolgen, sondern strukturiert durchgeführt werden.

(1) Testspezifikation

Die Testspezifikation beschreibt alle Anwendungsfälle (Use Cases), die vom System erfüllt werden müssen. Weiterhin sind hier auch zusätzliche sicherheitsrelevante Aspekte zu beschreiben. Hierunter fallen auch Zugriffseinschränkungen, die sich aus organisatorischen Gründen oder durch rechtliche Vorgaben ergeben (siehe auch Jahrgangsband 2, Lernfeld 8, Kapitel 3.8).

Im Bereich der Software-Entwicklung werden diese Tests weitgehend automatisiert durchgeführt. Diese Automatisierung ist bei IT-Systemen etwas schwieriger, da hier Netzwerkkomponenten und Server getestet werden müssen, die über unterschiedliche Kommunikationswege angesprochen werden.

Eine Netzwerkkomponente, wie z. B. ein Router, kann zwar meist auch über „ssh" kontaktiert werden, allerdings unterscheiden sich schon zwei Router eines Herstellers hinsichtlich ihrer Befehlsstrukturen, wenn diese mit speziellen Features ausgestattet sind. Router unterschiedlicher Hersteller haben so gut wie immer grundlegend andere Befehlssätze.

Bei den Serverdiensten verhält es sich ähnlich. Wird ein DHCP-Server benötigt, so unterscheiden sich die notwendigen administrativen Zugriffe auf Windows- oder ein Linux-Server ebenfalls.

Zu berücksichtigende Aspekte von Testungen		
Aspekt	**Erklärung**	**Beispiele**
Funktionen aus Nutzersicht	Hierunter fallen alle Anforderungen, die Nutzer an das System stellen.	automatische IP-Adressvergabe (DHCP); Zugriff auf Webfrontend, Dateiablage
Performance	Meist wird dieser Aspekt auch Last- und Leistungstest genannt.	Skalierungsfähigkeit, sprich: Funktionsfähigkeit des Systems auch bei großer Nutzerzahl bzw. Zuverlässigkeit des Datendurchsatzes; Einhaltung von Übertragungsraten, Antwortzeiten usw.
Sicherheit	Hierzu gehören Zugriffseinschränkungen genauso wie Datensicherheit gegen unbefugten Zugriff von außen wie innen.	Ausführung von Firewall-/VLAN-Regeln; Einschränkung von Rechten durch Authentifikationen an Systemen; Trennung von Administratorzugriffen und Nutzerzugriffen

Wenn man die einzelnen Testfälle identifiziert hat, können diese beschrieben werden. Hierzu sollten für die unterschiedlichen Testbereiche klar strukturierte Vorlagen erstellt werden, damit die Orientierung erleichtert wird.

Beispiel: Vorlage einer Testspezifikation

JKU IT-Solutions	
Testspezifikation	
Projektname (formal)	z. B. Projekttitel, Projektnummer oder Bezug zum Pflichtenheft
Ersteller (formal)	Name des Erstellers bzw. letzten Bearbeiters
Ablageort (formal)	Wo wird die Testspezifikation abgelegt? (Dateiname)
Version/Datum letzte Änderung (formal)	Meist werden Änderungen vorgenommen, diese können durch eine Versionierung schneller nachvollziehbar werden; eine Änderungshistorie sollte innerhalb der Testspezifikation Auskunft über die bereits getätigten Änderungen geben
Zielsetzung des Testfalls	Motivation für den Testfall; z. B. Anforderung aus dem Pflichtenheft oder gesetzliche Vorgabe
Ausgangssituation	Um einen Test aussagekräftig zu machen, muss für jeden Testdurchlauf die identische Ausgangssituation hergestellt werden. Diese sollte hinreichend genau beschrieben sein.
Testschritte	Die Durchführung des eigentlichen Tests wird Schritt für Schritt mit allen wesentlichen Aspekten beschrieben.
Testergebnis	Das erwartete Ergebnis des Tests wird hinreichend beschrieben. Hier können beispielsweise auch Fehlermeldungen dargestellt werden, die wegen Zugriffseinschränkungen erfolgen müssen.

Jeder Testfall wird mit Zielsetzung, Ausgangssituation, Testschritten und erwartetem Testergebnis beschrieben. Es kann sinnvoll sein, mehrere Testfälle, die inhaltlich nah beieinander liegen oder in einem Testlauf durchgeführt werden sollen, in einer Testspezifikation zusammenzufassen (siehe auch Jahrgangsband 2, Lernfeld 9, Kapitel 4.7).

(2) Testdokumentation

Die eigentliche Testdurchführung wird in einem Testprotokoll festgehalten. Die formalen Aspekte werden aus der Testspezifikation übernommen. Pro Testfall werden nach der Durchführung der Tests das tatsäch-

liche Ergebnis und die Abweichungen notiert. Entspricht das Testergebnis dem erwarteten Ergebnis, war der Test erfolgreich. Abweichungen müssen im Anschluss als Fehler klassifiziert werden.

Fehlerklassifizierung		
Fehlertyp	**Priorität**	**Beispiel**
kritischer Fehler	hoch	System stürzt ab; Vollzugriff aus dem Internet (kein Schutz)
geringe Abweichung	mittel	Anzeigefehler, die Funktion ist aber gewährleistet; „kosmetische" Fehler ohne Auswirkung auf die Funktionen
keine Abweichung (Verbesserungsmöglichkeit)	niedrig	Anzeige und Funktion korrekt, aber irreführend; könnte als Fehler missinterpretiert werden

Die Fehler sollten im Anschluss an die umsetzenden Personen gemeldet werden. Hierzu können sogenannte Ticketsysteme verwendet werden, bei denen sowohl umsetzende Personen wie Tester gemeinsam Testergebnisse klassifizieren und dokumentieren können. Mit GitLab wird ein weiter Teil dieser Funktionen abgedeckt. Es werden Installationspakete für alle großen Linux-Distributionen angeboten. Wurde aus Sicht der umsetzenden Person ein Fehler behoben, so kann der Tester hierüber informiert werden und einen neuen Testdurchlauf vornehmen.

(3) Automatisierung von Tests

Der Automatisierungsgrad von Tests hängt sehr stark von der zu testenden Infrastruktur ab. Bei häufig wiederkehrenden Tests können Testskripte zum Einsatz kommen. Werden Konfigurationssysteme wie „Ansible" verwendet, können innerhalb des System auch grundlegende Tests definiert werden. Bei der Erstellung derartiger Testskripte sollten die Parameter von den eigentlichen Testabläufen getrennt abgelegt werden. Beispielsweise könnte für einen Zugriffstest in Form von ping-, SSH- und HTTP-Anfragen ein Shell-Skript erstellt werden. Die jeweiligen anzufragenden Ziele, werden in eine eigenen Textdatei gespeichert. Damit wird das Shell-Skript ohne große Anpassung wiederverwendbar (siehe auch Lernfeld 10b, Kapitel 1.5).

```
# Inhalt ip.txt
192.168.0.1
10.0.2.1
...

# Shell-Testskript testet, ob die IP-Adressen aus ip.txt erreichbar sind
#!/bin/bash
cat ip.txt | while read ipaddr

do
 ping -c 1 "$ipaddr" > /dev/null
 if [ $? -eq 0 ]; then
   echo "IP: $ipaddr ist erreichbar"
 else
   echo "IP: $ipaddr ist offline"
 fi
done

# mögliche Ausgabe
IP: 192.168.0.1 ist offline
IP: 10.0.2.1 ist erreichbar
```

Kompetenzcheck

1. Geben Sie mindestens zwei Aspekte an, die bei Tests berücksichtigt werden.
2. Nennen Sie vier Informationen, die zur Beschreibung eines Testfalls in einer Testspezifikation benötigt werden.
3. Beschreiben Sie eine systematische Vorgehensweise für den Umgang mit Abweichungen bei Testdurchläufen.
4. Nennen Sie eine Möglichkeit, die Testqualität zu verbessern.
5. Bearbeiten Sie die Aufgabe 4 der Lernsituation 4 im Arbeitsbuch.

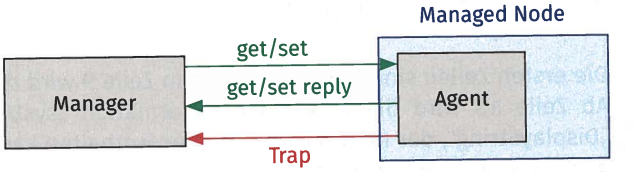
A4

1.4.3 Serverdienste überwachen

S **Sie sollen verschiedene Techniken kennenlernen, mit denen Serverdienste überwacht werden können.**

(1) Simple Network Management Protocol (SNMP)

Das Simple Network Management Protocol (SNMP) ist ein Protokoll, um Managementinformationen von Netzwerkelementen (Managed Nodes) abzufragen und zu ändern. Netzwerkelemente sind alle Geräte, die an ein Netzwerk angeschlossen werden können, z. B. Switche, Router, Server, Drucker etc. Das SNMP wurde von der IETF in mehreren RFCs standardisiert.



SNMP – Manager/Agent-Modell

Das Grundprinzip der Kommunikation besteht aus Anfragen eines Managers, die von einem Agenten beantwortet werden. Grundlegend sind Get- und Set-Anfragen für die Abfrage und Konfiguration von Managementinformationen vorgesehen. Um mehr als nur einen Parameter zu verändern, wurden mit Version 2 des Protokolls sogenannte **Bulk-Anweisungen** eingeführt. Mit Get-Bulk und Set-Bulk können mehrere Managementinformationen mit einem Befehl abgerufen oder manipuliert werden. Als weitere Kommunikation kann der Agent sogenannte **Traps** als Spontannachricht (z. B. bei Auftreten von Warnungen oder Fehler im Managed Node) an den Manager senden. Der Standardport **161** wird für SNMP-Get-Set-Abfragen des Managers verwendet. Die Traps des Agenten erreichen den Manager auf Port **162**.

In den ersten beiden Versionen von SNMP war nur eine sehr begrenzte Sicherheitsfunktion vorhanden. Die unverschlüsselten Protokollabläufe wurden durch sogenannte **Community-Strings** abgesichert. Im Agent wird ein Community-String für Get, typischerweise der Wert „public", und ein anderer für Set, typischerweise „private" gesetzt. Der Agent beantwortet eine Anfrage des Managers nur, wenn diese den konfigurierten Community-String mitsendet. Da die Anfrage unverschlüsselt gesendet wird, kann jeder Knoten im Netzwerk mitlesen und so den Community-String herausfinden. Mit Version 3 von SNMP wurde eine Verschlüsselung eingeführt, die jedoch mehr Konfigurationsaufwand bedeutet. Sie wird daher eher für Netzelemente verwendet, die über SNMP kritische Konfigurationen zulassen. Beispielsweise würde die Abfrage des Tonerstatus eines Druckers solch einen Aufwand nicht rechtfertigen.

Management Information Base (MIB)
Die Management Information Base (MIB) beschreibt die Daten, die über einen Agenten in einem Netzwerkelement abgefragt oder verändert werden können. Eine wichtige MIB, die alle Netzwerkelemente implementieren müssen, ist die sogenannte MIB-II. Sie wird im RFC 3418 beschrieben. Der folgende Ausschnitt aus der MIB-II zeigt den Aufbau.

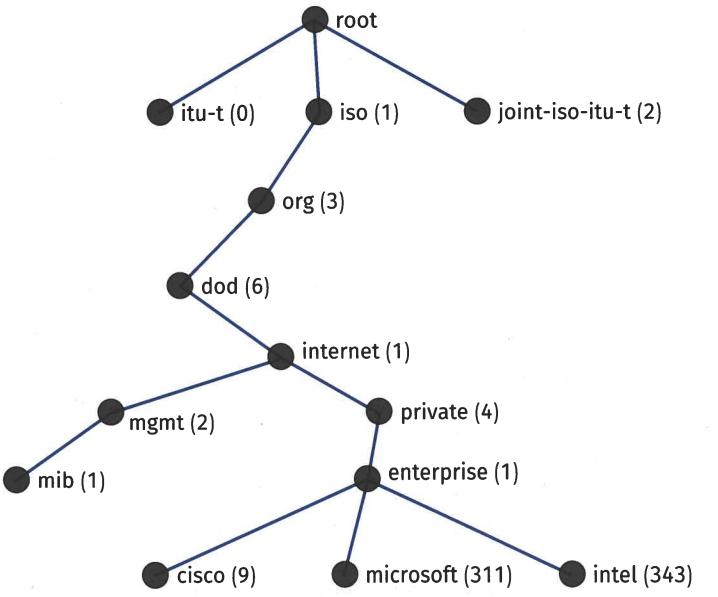
```
1  -- MIB module extracted from RFC 3418
2  -- Module SNMPv2-MIB (RFC 3418:12/2002)
3  --
4  -- Copyright (C) The Internet Society (2002). This version of
5  -- this MIB module is part of RFC 3418;
6  -- see the RFC itself for full legal notices.
7  --
8
9  SNMPv2-MIB DEFINITIONS ::= BEGIN
10
11 ...
12
13 system OBJECT IDENTIFIER ::= { mib-2 1 }
14
15    sysDescr OBJECT-TYPE
16        SYNTAX      DisplayString (SIZE (0..255))
17        MAX-ACCESS  read-only
18        STATUS      current
19        DESCRIPTION
20            "A textual description of the entity. This value should
21            include the full name and version identification of
22            the system's hardware type, software operating-system,
23            and networking software."
24        ::= { system 1 }
```

Die ersten Zeilen sind ein Kommentar. In Zeile 9 wird der Name der MIB, hier „SNMPv2-MIB", festgelegt. Ab Zeile 15 wird die Managementinformation „sysDescr" beschrieben. Es ist ein Objekt vom Typ „DisplayString", der maximal 255 Zeichen enthalten kann. Das Objekt hat als Zugriffsmöglichkeit („MAX-ACCESS") den Wert „read-only". Es kann also nur mit einem Get-Aufruf abgefragt werden. Im Vergleich dazu kann mit einem „read-write" als Zugriffswert auch eine Set-Operation durchgeführt werden.

Object Identifier (OID)

Jede MIB und jede Managementinformation wird durch einen eindeutigen OID adressiert. OIDs werden auch im Zusammenhang mit Verzeichnisdiensten verwendet. Die Objekte im **Schema** eines Active Directory werden mit OIDs identifiziert. Ein weiteres Beispiel für den Einsatz von OIDs ist die Public Key Infrastructure (PKI). Hier werden beispielsweise Zertifikate durch OID eindeutig benannt.

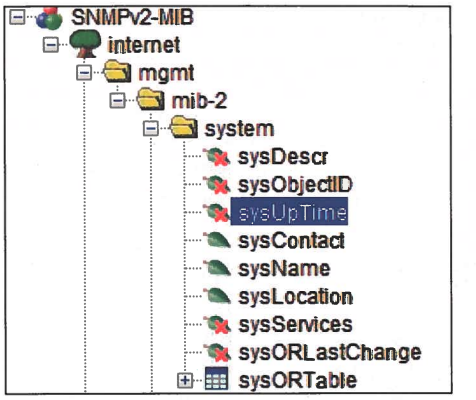
Eine OID kann durch eine hierarchische Baumstruktur dargestellt werden. Jeder Knoten wird mit einer dezimalen Zahl identifiziert. Der Pfad entlang der Knoten führt zu einem Endknoten, der durch die OID beschrieben wird. Sie ist eine punktgetrennte Zahlenfolge aller Zahlen auf dem Weg zum Endpunkt.



Aufbau eines MIB-Baums für SNMP

Beispiel: Folgt man der Baumstruktur zur MIB-2, erhält man die OID „1.3.6.1.2.1“. Im Quelltext auf Seite 114 steht in Zeile 13 „system OBJECT IDENTIFIER ::= { mib-2 1 }“. Hier wird unter der „mib-2“ die OID „system“ mit dem Wert 1 gesetzt. Die gesamte OID von „system“ lautet also „1.3.6.1.2.1.1“. Darunter hat „sysDescr“ die OID „1.3.6.1.2.1.1.1“. Weitere Objekte der MIB-2 sind:

- 1.3.6.1.2.1.1.1 – sysDescr (nur lesbar; enthält vom Hersteller vorgegebene Beschreibung des Systems)
- 1.3.6.1.2.1.1.3 – sysUpTime (nur lesbar; enthält die Zeit, zu der die Netzwerkeinheit des Gerätes letztmals neu initialisiert wurde)
- 1.3.6.1.2.1.1.4 – sysContact (les-/schreibbar; hier können Kontaktinformationen eingetragen werden)
- 1.3.6.1.2.1.1.5 – sysName (les-/schreibbar; hier kann ein Systemname vergeben werden)



Standardisierte MIB-2

Neben standardisierten MIB können Firmen auch eigene MIB erstellen. Dafür gibt es den Zweig „enterprise“ unter „1.3.6.1.4.1“. Firmen erhalten einen eigenen Knoten, den sie bei der IANA beantragen können.

Beispiele: Zuordnung von OID zu einem bestimmten Unternehmen:

- 1.3.6.1.4.1.9 – Cisco
- 1.3.6.1.4.1.311 – Microsoft
- 1.3.6.1.4.1.343 – Intel Corporation

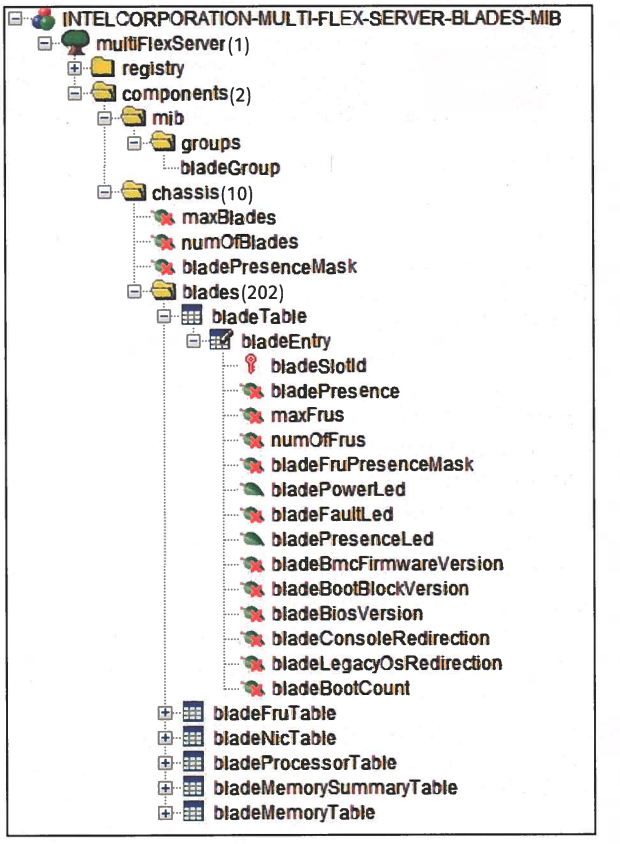
Intel-Flex-Server-MIBs

Als Beispiel wird die herstellereigene Intel-Flex-Server-MIB analysiert.

Unter der Baumstruktur „.iso.org.dod.internet.private.enterprise.intel.products.modularsystems.multiFlexServer.components.chassis.blades“ mit der OID .1.3.6.1.4.1.343.2.19.1.2.10.202.1 ist die „bladeTable“ abgelegt.

Hier können z. B. folgende Objekte ausgelesen werden:

- bladeBmcFirwareVersion (die Version der Firmware des Systems)
- bladeBiosVersion (die BIOS-Version des Systems)



Herstellerspezifische MIB

Weitere Tabellen geben beispielsweise Informationen über die Prozessoren und den Arbeitsspeicher.

(2) Server Monitoring

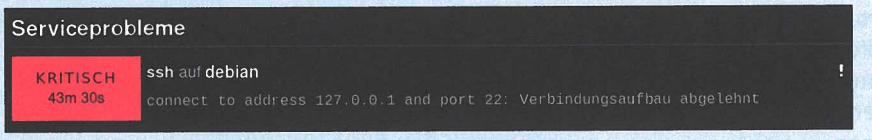
Eine Überwachung einzelner oder mehrerer Server oder des ganzen Netzwerks ist zur Sicherstellung der Verfügbarkeit und der Performance des überwachten Systems nötig. Damit wird eine mühsame Fehlersuche vermieden und der Administrator erhält einen guten Überblick über das Netzwerk. Dazu können

wie oben beschrieben beispielsweise die Auslastung eines Systems oder zur Verfügung stehende Datenraten einer Router-Schnittstelle überwacht und protokolliert werden.

Um ein System kontinuierlich zu überwachen, ist das manuelle Abfragen von Werten in der MIB des Systems nicht geeignet. Für die umfangreiche Überwachung eines ganzen Netzwerks inklusive der Koppelelemente wie Switches oder Router bietet sich ein Monitoring-System an. Dieses überwacht festgelegte Geräte und meldet dem Administrator Warnungen und Fehler.

Warn- und Fehlermeldungen

Bei einer **Warnung** kann der Administrator reagieren, um einen kritischen Systemzustand zu vermeiden und einen Fehler zu verhindern. Typischerweise wird eine Warnung erzeugt, wenn bei einem zu überwachenden System ein bestimmter Schwellwert überschritten wird. Wird beispielsweise ein Dateiserver beobachtet, könnte der Administrator die Belegung der Speichermedien überwachen. Unterschreitet der freie Speicherplatz einen vorher definierten Wert, wird der Administrator z. B. per Trap informiert. Er kann nun auf die Warnung reagieren, ohne dass es beim Benutzer zu einem Fehler kommt.



Beispiel für einen kritischen Fehler

Anders als bei einer Warnung liegt bei einer **Fehlermeldung** ein kritischer Systemzustand (siehe Abbildung) vor. Das System ist beispielsweise nicht erreichbar oder das Speichermedium des Datei-Servers ist voll und es können keine weiteren Daten mehr abgespeichert werden. Bei einem Fehler können Benutzer nicht mehr im vollem Umfang mit dem System arbeiten.

Für die Überwachung können verschiedene Protokolle eingesetzt werden. Häufig wird das Protokoll „SNMP“ verwendet (siehe auch Kapitel 1.4.3), aber auch Protokolle wie ICMP (Internet Control Message Protocol) zur Überwachung der Verfügbarkeit, WMI (Windows Management Instrumentation), z. B. zum Auslesen von Ereignisprotokollen bei Windows-Systemen, oder MQTT (Message Queueing Telemetry Transport) zum Überwachen von IoT-Geräten werden bei Monitoring-Systemen eingesetzt.

Zur Überwachung von Netzwerken ist eine große Anzahl von Software verfügbar. Neben herstellereigenen Programmen wie z. B. der Software „Junos OS Tool“ von „Juniper“ gibt es herstellerunabhängige kommerzielle Programme, z. B. das Monitoring-System „PRTG“ der Firma „Paessler“. Alternativ gibt es auch kostenfreie Open-Source-Programme, wie z. B. „Nagios“ oder dessen Entwicklungszweig „Icinga“.

Icinga ist eine beliebte Open-Source-Monitoring-Software. Mit dieser Software können Serverdienste im Netzwerk, Ressourcen von Systemen und Netzwerkelemente überwacht werden. Die zu überwachenden Parameter werden zustandsspezifisch visualisiert und im Falle von Warnungen bzw. Fehlern wird der Administrator informiert. Icinga steht in der Version 2 für eine große Anzahl an Betriebssystemen bereit. Nach der Installation der „incinga2“-Pakete, des Webfrontend „incingaweb2“ und einer Datenbankanbindung, z. B. „mariadb“, ist Incinga einsatzbereit. Für die verschiedenen Überwachungsmöglichkeiten verwendet Incinga aufgabenspezifische Plugins.

Je nachdem, welche Eigenschaft eines Systems überwacht werden soll, kommt ein anderes Plugin zum Einsatz. Für die Überprüfung der Erreichbarkeit mittels „ping“-Befehl wird beispielsweise das Plugin „check_ping“ eingesetzt, während die CPU-Last mit dem nachzuinstallierenden Plugin „check_snmp_load“ überwacht wird. Ein Plugin ist ein Skript, das entsprechend parametrisiert, auch ohne Incinga funktionsfähig ist.

```
[redacted]/usr/lib/nagios/plugins$ ./check_snmp_load.pl -H 192.168.178.29 -C public -w 70 -c 90
1 CPU, load 25.0% < 70% : OK
```

Skriptaufruf, um die CPU-Last zwischen Schwellwerten abzufragen

Das Skript „check_snmp_load" soll die CPU-Last auf dem Zielsystem ausgeben. Zur Ausführung des Skriptes sind die Schwellwerte für Warnungen und Fehler angegeben. Auf der Protokollebene wird ein SNMP-Get-Request für einen bestimmten MIB-Wert, in diesem Fall der Eintrag, der die CPU-Last anzeigt, gesendet. Die Antwort, die zweite Nachricht im Wireshark-Mitschnitt, liefert den angeforderten Wert zurück. Die Interpretation des Wertes, also ob der Schwellwert für Warnung („-w 70" ab 70 %) oder Fehler („-c 90" ab 90 %) überschritten ist, übernimmt das Skript.

```
No.  Source            Destination       Protocol  Length  Info
  … 192.168.178.55     192.168.178.29    SNMP      90      get-next-request 1.3.6.1.2.1.25.3.3.1.1.196608
  … 192.168.178.29     192.168.178.55    SNMP      91      get-response 1.3.6.1.2.1.25.3.3.1.2.196608
Frame 36: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: Synology_1e:32:a2 (00:11:32:1e:32:a2), Dst: PcsCompu_ad:f8:8b (08:00:27:ad:f8:8b)
▸ Internet Protocol Version 4, Src: 192.168.178.29, Dst: 192.168.178.55
▸ User Datagram Protocol, Src Port: 161, Dst Port: 41223
▾ Simple Network Management Protocol
    version: version-1 (0)
    community: public
  ▾ data: get-response (2)
    ▾ get-response
        request-id: 1629049634
        error-status: noError (0)
        error-index: 0
      ▾ variable-bindings: 1 item
        ▸ 1.3.6.1.2.1.25.3.3.1.2.196608: 25
```

Mitschnitt der SNMP-Kommunikation

Um ein System, z. B. einen Server, automatisch zu überprüfen, muss das System in Icinga eingetragen werden. Dazu gibt es die Datei „hosts.conf", in der die zu überwachenden Systeme als Objekte hinterlegt sind. Die Datei liegt im Verzeichnis „/etc/icinga2/conf.d".

```
1 object Host "Webserver_Katasteramt" {
2         import "generic-host"
3         address = "192.168.178.38"
4         vars.os = "Linux"
5         vars.http_vhost["myserver"]={
6            http_vhost = "192.168.178.38"
7         }
8 }
```

Im Beispiel oben soll ein Webserver überwacht werden. In der ersten Zeile wird der angezeigte Name „Webserver_Katasteramt" definiert. Anschließend wird mit dem Import „generic-host" die Überprüfung durch das Skript „check_ping" hinzugefügt. In Zeile 3 wird die IP-Adresse des zu überwachenden Systems angegeben. Mit der Angabe „vars.os="Linux"" (Zeile 4) können Linux-spezifische Tests aktiviert werden. Die Zeilen 5 und 6 geben an, dass es sich um einen Webserver handelt. Da mehrere Domains auf einer IP-Adresse verfügbar sein können, kann mit dem Parameter „http_vhost" die URL angegeben werden. Nach dem Anlegen des Objekts und dem Neustart des Dienstes wird der eingetragene Webserver in Icinga angezeigt.



Die überwachten Dienste und Geräte laufen fehlerfrei.

(3) Cluster überwachen

Ein Cluster besteht aus mehreren Knoten, die einen oder mehrere Dienste zur Verfügung stellen. Dabei wird die Dienstbereitstellung von jeweils einem Knoten übernommen. Sollte dieser Knoten ausfallen, übernimmt ein anderer Knoten die Bereitstellung (Active/Passive-Cluster). Die Knoten können zur Lastverteilung gemeinsam einen Dienst bereitstellen.

Ein Knoten eines Clusters kann ein physisches System, aber auch ein Container oder eine virtuelle Maschine sein. Jeder Knoten eines Cluster kann separat überwacht werden. Die Überwachung erfolgt wie bei einem Einzelsystem (siehe Kapitel 1.4.2). Dabei können die physische Verfügbarkeit des Systems, die Verfügbarkeit eines Dienstes oder die CPU-Auslastung eines Knotens oder Clusters überprüft werden.

Allerdings gibt es noch spezifische Aspekte, die bei einem Cluster überwacht werden können. Eine Überwachung eines Clusters kann anzeigen, ob gerade ein Failover stattfindet und ob dieser erfolgreich ist. Außerdem kann eine Monitoring-Software feststellen, ob in einem Cluster bei einem Notfall genügend Ressourcen für die Durchführung eines Failovers vorhanden sind.

Idealerweise werden die Dienste, die durch die Cluster-Knoten angeboten werden, durch die Monitoring-Software angezeigt. Neben der Auslastung der einzelnen Knoten des Clusters sollte auch eine Auslastung des kompletten Clusters dargestellt werden. Ein Administrator kann so im Bedarfsfall dem Cluster weitere Ressourcen, z. B. virtuelle Maschinen oder Container, hinzufügen, um das Gesamtsystem zu skalieren.

Zur Überwachung von Kubernetes-Clustern kommt häufig die Software „Prometheus“ zum Einsatz. Auf einem zu überwachenden System wird dabei ein sogenannter Exporter eingesetzt (ähnlich dem Agent bei SNMP). Dieser sammelt bestimmte Systemwerte. Durch den Prometheus-Server werden die Exporter abgefragt und liefern die gesammelten Daten. Am Server werden die Daten, teilweise zusammengefasst, in einer Datenbank gespeichert. Wie bei SNMP-Monitoring können bei der Clusterüberwachung Meldungen versendet werden, wenn bestimmte Schwellwerte erreicht werden. Die Darstellung findet über eine eigenständige Software statt. Häufig werden die überwachten Systeme und deren Systemwerte mit der Software „Grafana“ grafisch aufbereitet und dargestellt.

Kompetenzcheck ✓

1. Nennen Sie die Komponenten, die in einem SNMP-System zum Einsatz kommen.
2. Erklären Sie den Unterschied zwischen MIB und OID in eigenen Worten.
3. Geben Sie mindestens zwei Informationen an, die in einem Monitoring-System gesammelt werden.
4. Geben Sie mindestens zwei Protokolle an, die zur Überwachung eingesetzt werden.
5. Bearbeiten Sie die Aufgabe 5 der Lernsituation 4 im Arbeitsbuch.

A5

1.4.4 Dokumentation der IT-Systeme

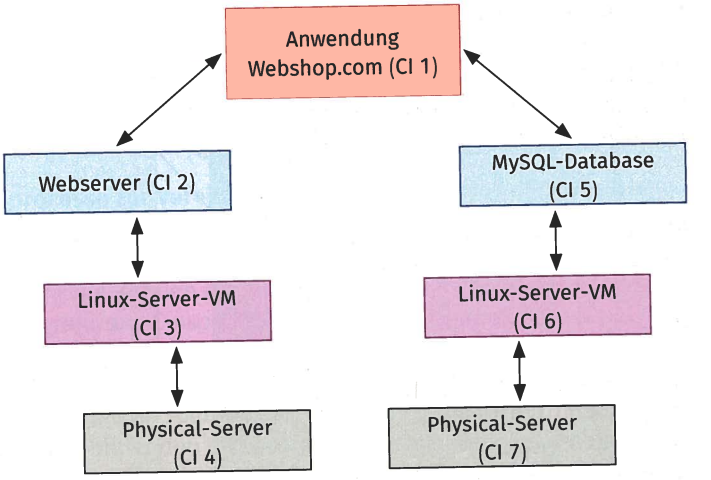
S **Nach der Implementierung der Systeme müssen Sie die Schritte zur Implementierung und die Konfigurationsparameter dokumentieren.**

In einem Unternehmen wie der JIKU IT-Solutions GmbH ist es zwingend notwendig, die gesamte IT-Landschaft zu dokumentieren. Ziel soll es sein, eine gemeinsame Sicht auf das System zu erhalten. Auch bei der Erweiterung des Systems oder der Fehlersuche ist eine Dokumentation hilfreich. Weitere Aspekte sind die Inventarisierung und die Lizenzverwaltung. Dies ist z. B. für die Bestimmung der vorhandenen bzw. anzuschaffenden Softwarelizenzen wichtig.

Bei der Entwicklung von IT-Systemen als Hilfswerkzeuge sowie beim Einsatz von IT-Systemen als zentrale Bestandteile von Geschäftsprozessen im Unternehmen stehen der Servicegedanke und die schnelle Entstörung bei Ausfällen im Vordergrund (siehe auch Jahrgangsband 2, Lernfeld 6, Kapitel 1.1.2).

Um ein standardisiertes Vorgehen zu etablieren, wurde in der ISO 20000 International IT Service Management (ITSM) eine Serviceinfrastruktur beschrieben. Kern des Servicemanagements ist die Dokumentation der IT-Infrastruktur, die in einer **Configuration-Management-Data-Base (CMDB)** vorgenommen wird. Die CMDB ist ein Repository, in dem alle Soft- und Hardwares eines Unternehmens sowie deren Abhängigkeiten gespeichert werden.

Grundlage der CMDB sind sogenannte **Configuration Items (CI)**, die sich in **Physical Entities** und **Logical Entities** unterteilen. Physical Entities sind beispielsweise Computer, Switch oder Serverschrank. Logical Entities sind beispielsweise eine Instanz einer Datenbank, ein Dienst, eine virtuelle Maschine oder eine Lizenz. Configuration Items werden über **CI-Relationships** miteinander verbunden.



CMDB – Webshop mit CI-Relationships

Die Abbildung zeigt die Anwendung „Webshop.com", die als (CI1) mit dem Webserver (CI2) und der MySQL-Datenbank (CI5) über CI-Relationships verbunden ist. Der Webserver (CI2) ist mit der Linux-Server-VM (CI3) und dem Physical-Server (CI4) verbunden. Auf der anderen Seite ist die MySQL-Datenbank (CI5) mit der Linux-Server-VM (CI6) und diese mit dem Physical-Server (CI7) verbunden. In dem Beispiel sind die Linux-Server-VMs (CI3) und (CI6) unterschiedliche Instanzen. Würden der Webserver und die Datenbank auf der gleichen VM laufen, würde durch die CI-Relation die Abhängigkeit klar werden. Allgemein beschreiben die CI-Relations die Abhängigkeiten der CIs voneinander. Fällt beispielsweise der Physical-Server (CI7) aus, würden der entsprechende Linux-Server und die MySQL-Datenbank und letztendlich die Anwendung „Webshop.com" davon betroffen sein.

Die gezeigten Abhängigkeiten geben auch bereits eine Reihenfolge vor, wie eine IT-Dokumentation sinnvoll durchzuführen ist, und zwar der Reihenfolge nach:

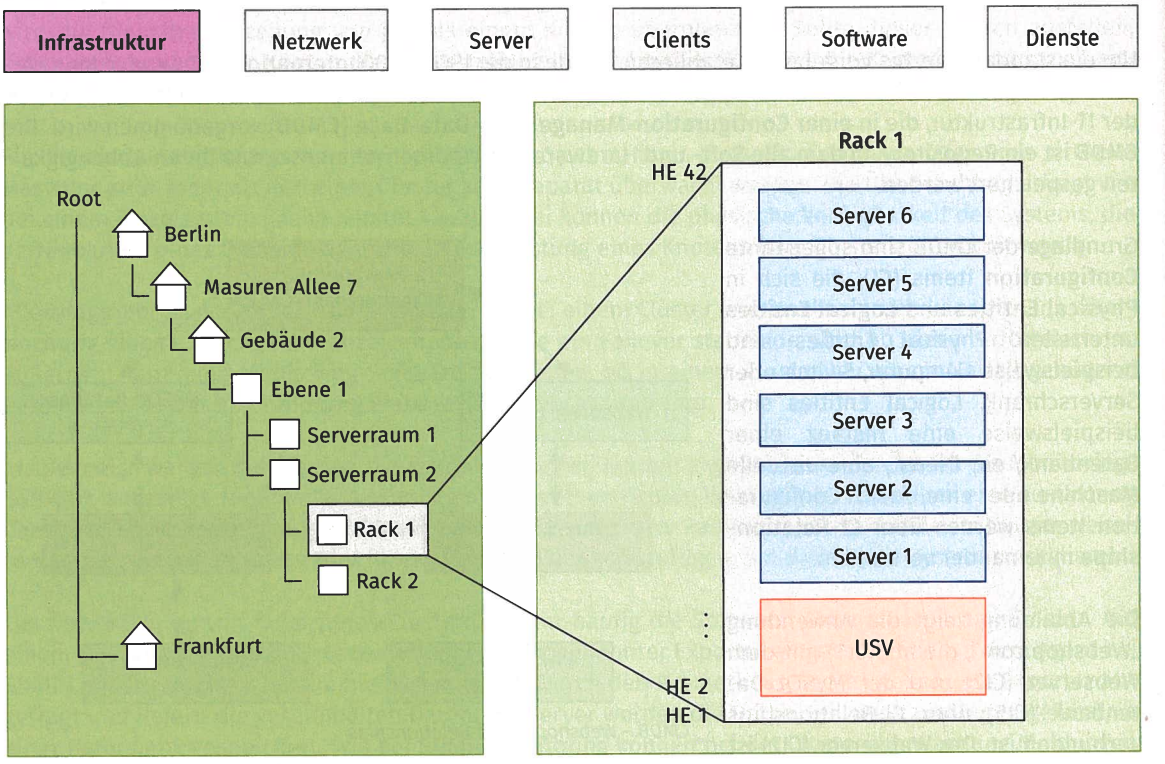
1. Infrastruktur
2. Netzwerk
3. Server
4. Clients
5. Software und Lizenzen
6. Anwendungen/IT-Dienste

Die nachfolgende Abbildung zeigt schematisch eine CMDB-Software. Im oberen Teil können die Bereiche ausgewählt werden, die dokumentiert werden sollen, analog zu den zuvor aufgezählten Bereichen. Im dem Beispiel ist die **Infrastruktur** ausgewählt.

Im linken Teil des Fensters können Orte strukturiert abgelegt werden. In dem Beispiel gibt es je einen Standort in Berlin und Frankfurt. Sollten an einem Ort mehrere Standorte vorhanden sein, kann das zusätzlich mit der genauen Adresse unterteilt werden.

In dem Beispiel ist ein Rack im Serverraum 2 ausgewählt, das grafisch mit den Höheneinheiten (HE) dargestellt wird. Entsprechend wurde das Rack mit den montierten Einheiten befüllt. Jede montierte Einheit

wird mit seinen technischen Daten und den Verbindungen (z. B. Stromversorgnung und Netzwerkanbindung) dokumentiert.



Beispiel für die schematische Oberfläche einer CMDB-Software

Die exemplarisch gezeigte Dokumentation setzt sich fort für:

- Netzwerk:
  - Elemente, Dosen, Verkabelung
  - logische Unterteilung: VLAN, IP-Netzwerke etc.
- Server/Clients:
  - installiertes Betriebssystem, Software
  - Raumplan, wo der Client steht
  - Anschluss an Netzwerk, Stromversorgung
- Software/Lizenzen
- Dienste/Anwendungen

Was wie ein großer Aufwand erscheint, zahlt sich bei Wartungsarbeiten oder unvorhergesehenen Ausfällen aus. Durch die Dokumentation der Verkettungen können sofort alle Abhängigkeiten erfasst werden. Meldet ein Nutzer den Ausfall des E-Mail-Systems, können somit sofort alle dazu nötigen Systeme systematisch überprüft werden (siehe auch Jahrgangsband 1, Lernfeld 4, Kapitel 4).

Neben der Dokumentation für das IT-Servicemanagement ist eine detaillierte Dokumentation auch für die Schutzbedarfsanalyse im eigenen Arbeitsbereich notwendig. Grundlage des Informations-Sicherheits-Manangement-Systems (ISMS) und der daraus folgenden Schutzbedarfsanalyse ist eine Dokumentation der IT-Systeme (Strukturanalyse). Hier findet eine Verknüpfung mit dem IT-Servicemanagement statt. Softwareprodukte, die eine CMDB für das IT-Servicemanagement anbieten, verfügen oft auch über entsprechende Erweiterungen zur Durchführung der Schutzbedarfs- und Risikoanalyse gemäß IT-Grundschutz bzw. ISO-27001-Zertifizierung auf Basis der dokumentierten IT-Infrastruktur.

Kompetenzcheck

1. Geben Sie mindestens drei Aspekte an, nach denen IT-Systeme dokumentiert werden können.
2. Nennen Sie mindestens zwei Vorteile, die sich aus einer vollständig gepflegten Dokumentation ergeben.
3. Nennen Sie zwei Standards, die eine Dokumentation der IT-Systeme erfordern.
4. Bearbeiten Sie die Aufgabe 6 der Lernsituation 4 im Arbeitsbuch.

A6

1.5 Automatisierungen von Administrationsprozessen gemäß kundenspezifischen Rahmenbedingungen

S Sie automatisieren Administrationsprozesse zur Vereinheitlichung der Abläufe und zur Minimierung von Fehlern.

Ein großer Teil der Aufgaben bei der Betreuung von Kunden der JIKU IT-Solutions ist die Bearbeitung von Routineaufgaben. Hierzu gehören unter anderem das Einspielen von Software-Updates, die Durchführung einer Datensicherung oder das Verwalten von Benutzern. Viele dieser Tätigkeiten lassen sich so vorbereiten, dass die eigentliche Ausführung nicht von Menschen, sondern von einem Computer durchgeführt werden kann. Von den Mitarbeitern der JIKU IT-Solutions wird erwartet, solche Automatisierungen anzuwenden. Deshalb werden auch die Auszubildenden frühzeitig in den Verfahrensweisen der Automatisierung geschult. Die Vorgehensweise, verschiedene Tätigkeiten zu automatisieren, wird im Folgenden exemplarisch vorgestellt.

1.5.1 Administration automatisieren

S Sie sollen sich in die Automatisierung der Administration einarbeiten.

Administration ist ein kontinuierlicher Prozess, sodass hier viel Arbeitszeit einzuplanen ist. Eine Automatisierung dieser Aufgaben ist daher wünschenswert. Im Folgenden werden Tools und mögliche Vorgehensweisen zur Automatisierung von Administrationsaufgaben vorgestellt.

(1) Skriptsprachen

Einem Computer können Aufgaben durch Befehle beschrieben werden. Einzelne Befehle werden in Programmiersprachen zusammengefasst. Im Bereich der Administration werden meist Skriptsprachen verwendet, da diese leicht anpassbar sind.

Skriptsprachen werden von sogenannten Interpretern ausgeführt. Das heißt, während das Programm abläuft, wird jeder Befehl vom Interpreter in sogenannte OpCodes, also vom Betriebssystem direkt ausführbare binäre Befehle übersetzt. Diese OpCodes werden im Anschluss durch das Betriebssystem an die CPU weitergeleitet und dort ausgeführt. Im Gegensatz dazu wird bei einer Compilersprache der Quellcode zum Zeitpunkt der Programmentwicklung durch den Compiler in binäre Befehle übersetzt, die zu einem späteren Zeitpunkt direkt von der CPU ausgeführt werden können.

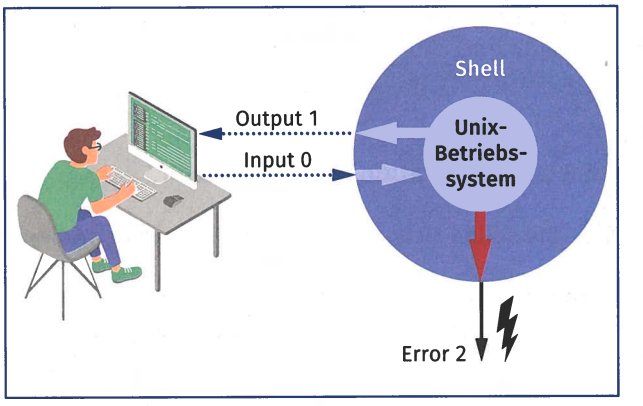
Just-in-Time-Compiler (JIT-Compiler) arbeiten ähnlich wie Interpreter und Compiler. Sie übersetzen den Quellcode unmittelbar vor der Ausführung. Hierdurch können sie plattformunabhängig eingesetzt werden, sofern eine entsprechende Umsetzung auf der jeweiligen Plattform existiert.

Gegenüberstellung von Programmiersprachen	
Name	**Beschreibung**
Shell	Kommandozeileninterpreter unter Unix-Betriebssystemen
Python	objektorientierte Interpreter-Sprache; leicht zu erlernen; sehr umfangreiche Bibliotheken; auf fast allen Betriebssystemen verfügbar
Perl	objektorientierte Interpreter-Sprache; relativ alt, daher sehr weit verbreitet; sehr umfangreiche Bibliotheken
PowerShell	Kommandozeileninterpreter; Programmiersprache für Windows (Linux); verhältnismäßig jung; vordefinierte Objekte für Zugriff auf Betriebssystem-funktionen
C/C++, Pascal/Delphi	Compilersprache, C/C++ gelten als sehr schnell
Java, C#	JIT-Compiler; weitgehend plattformunabhängig

Der Kommandozeileninterpreter „Shell" hat die Besonderheit, dass hier zwar grundlegende Befehle integriert sind, aber meist ausführbare Programme wie „grep" benutzt werden, um spezielle Aufgaben zu erfüllen. Häufig werden diese eigenständigen Programme als „Shell"-Befehl bezeichnet. Die „Shell" selbst ruft aber diese Programme auf. Ohne diese externen Befehle verfügt „Shell" über grundlegende Programmkonstrukte wie Variablen, Vergleiche, Schleifen und Funktionen.

(2) „Shell" unter Linux

Unter Unix-Betriebssystemen gibt es sehr viele Kommandozeileninterpreter. Der älteste Kommandozeileninterpreter ist die „sh", die von Stephen R. Bourne 1977/78 entwickelt wurde. In aktuellen Linux-Distributionen wird häufig die „bash" (Bourne-again Shell) als Standard eingesetzt. Das in Form der „Manpage" mitgelieferte Handbuch umfasst gut 4 500 Zeilen. Im Folgenden wird die „Bash" als „Shell" zugrunde gelegt. Wenn eine anderer Kommandozeileninterpreter verwendet wird, können die Befehle leicht abweichen. Mit dem folgenden Befehl kann man die „Login-Shell" anzeigen lassen.



Unix Input/Output-Modell

```
$ echo $SHELL
/bin/bash
```

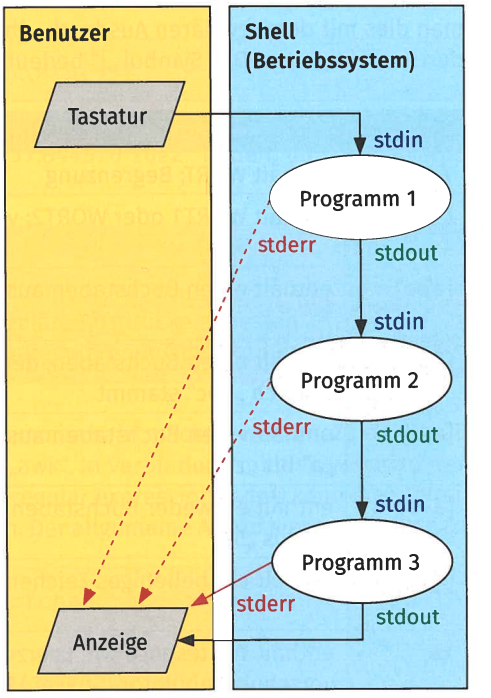
Die „Shell" stellt die grundlegende Verbindung zwischen dem Betriebssystem (Kernel) und dem Benutzer her. Dabei werden Zeichen oder Befehle zwischen dem Benutzer und dem eigentlichen Betriebssystem ausgetauscht. Sie eignen sich besonders zur Automatisierung von Aufgaben. Grafische Benutzerschnittstellen (GUI) erfordern meist Eingaben per Maus oder Touchscreen und sind weniger zur Automatisierung geeignet.

Das grundlegende Konzept der „Shell" besteht aus drei Wegen zwischen Benutzer und Betriebssystem. Per Standard INPUT („stdin, $0"; „$0" ist die Variable, in der „stdin" gespeichert bzw. ausgelesen werden kann) werden Anweisungen an das Betriebssystem durchgereicht. Das Betriebssystem kann bei erfolgreicher Ausführung der Anweisungen über den Standard OUTPUT („stdout, $1") das Ergebnis zurückliefern. Im Fehlerfall kann das Betriebssystem per Standard ERROR („stderr, $2") Fehlermeldungen ausgeben. Programme können weitere Ein- und Ausgabemöglichkeiten haben.

Die „Shell“ hat eingebaute Befehle, mit deren Hilfe man bereits grundlegende Informationen über das System abfragen oder Änderungen herbeiführen kann.

Auswahl eingebauter bash-Befehle	
Name/Befehl	**Beschreibung**
echo	Ausgabe von Variableninhalten (vgl. print in anderen Programmiersprachen)
$VAR	Variable VAR wird mit $ gekennzeichnet
for-Schleife	führt eine Folge von Befehlen aus; meist über ein Inkrement oder eine Liste gesteuert
if	lässt Programme in Abhängigkeit von Vergleichen verzweigen
while-Schleife	führt eine Folge von Befehlen aus, bis ein Abbruchkriterium erfüllt wird
function (Unterfunktion)	häufig verwendete Befehlsfolgen lassen sich in Funktionen ablegen

Die eingebauten „shell“-Befehle stellen die Grundfunktionalität dar. Durch Verwendung von weiteren Programmen lassen sich komplexe Aufgaben mit wenigen Zeilen Programmcode erledigen. Die große Stärke sind dabei die sogenannten Pipelines, also Verkettungen von Befehlen bzw. Programmen. Dabei wird der Inhalt von „stdout“ eines Befehls zum Inhalt von „stdin“ des nächsten Befehls. Die nebenstehende Abbildung zeigt dies für drei Programme.



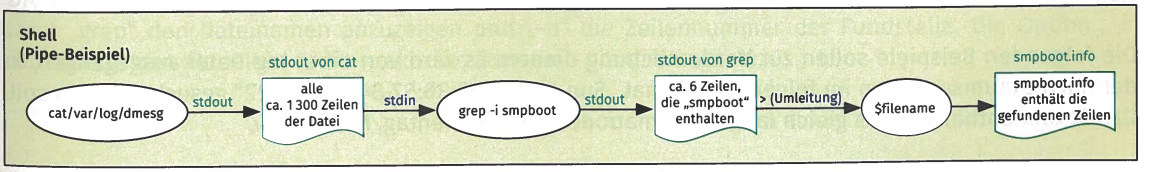
Verkettung der Linux-Kernel-Standardkanäle

(3) Exemplarisch Teilaufgaben lösen

Ein Skript ist zunächst eine Textdatei, die meist mit der Endung „.sh“ versehen wird. Die Endung ist unter Linux anders als bei Windows nicht ausschlaggebend für die Ausführbarkeit. Um unter Linux eine Datei als ausführbar zu kennzeichnen, wird der Befehl „chmod +x <SKRIPTNAME.sh>“ verwendet („chmod“ = „change mode“). Dieser setzt das x-Bit (eXecutable) im Dateisystem.

Folgende Datei listet beispielsweise alle Nachrichten zum symmetrischen Multiprozessorsystem (SMP) im Bootvorgang (smpboot) aus den log-Dateien auf und speichert diese in der Datei „smpboot.info“.

```
1 #!/bin/bash
2 filename = smpboot.info
3 cat /var/log/dmesg | grep -i smpboot > $filename
4
5 echo "SMP-Boot Infos sind in $filename gespeichert."
```



Beispiel einer Pipe-Verkettung

Zunächst wird der „Shell" in der ersten Zeile der Skriptinterpreter in einem Kommentar (#: Kommentarzeichen) mitgeteilt. Dies wird mit Hilfe des Ausrufezeichens und des Aufrufpfads (hier: /bin/bash) erreicht. In diesem Fall handelt es sich um die „Bash" selbst. Hier könnte auch „#!/bin/python3" für den Python-3-Interpreter genutzt werden. In der Variablen „filename" (Zeile 2) wird der Dateiname (smpboot.info) der Ausgabedatei abgelegt. Der „cat"-Befehl (Zeile 3) gibt den Inhalt einer Datei auf der „stdout" aus. Da der Inhalt aber zunächst auf das Schlüsselwort „smpboot" gefiltert werden soll, wird der „stdout" des „cat"-Befehls mittels „|" (pipe) auf „stdin" des „grep"-Befehls weitergeleitet. Das Ergebnis dieser Filterung wird aus der „stdout" des „grep"-Befehls mit „>" in die Datei „$filename" (smpboot.info) umgeleitet. Würde man „> $filename" entfernen, so würde das Ergebnis im Terminalfenster angezeigt und keine Datei erzeugt werden.

(4) Regular Expressions (RegEx)

Regular Expressions (RegEx), d. h. reguläre Ausdrücke, werden an vielen Stellen in der Informationstechnik eingesetzt, um Texte nach bestimmten Kriterien zu filtern. Insbesondere dann, wenn man einen Text nicht nur nach einem bestimmten Suchbegriff durchsuchen möchte. Möchte man beispielsweise in einem Text alle Stellen finden, in denen entweder das Wort „heute" oder „morgen" enthalten ist, dann könnte man dies mit dem regulären Ausdruck „/heute|morgen/" beschreiben. Der eigentliche Suchbereich wird durch „/" begrenzt. Das Symbol „|" bedeutet ODER.

Symbol	Beschreibung	Beispiel
/	enthält WORT; Begrenzung	/WORT/ findet WORT
|	enthält WORT1 oder WORT2; verknüpft Suchbegriff	/WORT1|WORT2/ findet WORT1 oder WORT2
[abc]	enthält einen Buchstaben aus dem Bereich „abc"	/WORT[abc]/ findet WORTa, WORTb, WORTc
[^abc]	enthält einen Buchstaben, der nicht aus dem Bereich „abc" stammt	/WORT^[abc]/ findet WORTd aber nicht WORTa
[a-z]	enthält einen Buchstaben aus dem Bereich „a" bis „z"	/WORT[a-z]/ findet WORTa bis WORTz
[a-c1-5]	enthält entweder Buchstaben „a" bis „c" oder Ziffer „1" bis „5"	/WORT[a-c1-5]/ findet z. B. WORTa oder WORT4
[.]	enthält ein beliebiges Zeichen	/W.RT/ findet WoRT oder W1RT oder WART
\s	enthält Whitespace wie Leerzeichen, Zeilenvorschub, Tabulator (Space; \S negiert; also kein Whitespace)	/WORT\sTROW/ findet WORT TROW, aber nicht WORTTROW
\d	enthält Ziffer \d (digit; \D negiert; also keine Ziffer)	/W\dRT/ findet W1RT aber nicht WORT
\w	enthält Wort \w (\W negiert; also kein Wort)	/\w/ findet WORT, aber nicht W1RT

Ausgewählte RegEx-Ausdrücke

Weitere Operatoren ermöglichen es, die genaue Anzahl zu beschreiben. Die Zeichen „?" (keinmal oder einmal), „*" (keinmal oder mehr), „+" (einmal oder mehr), „{3}" (genau „3"), „{3,}" (dreimal oder mehr) oder „{3,5}" (drei- bis fünfmal) werden hierzu hauptsächlich verwendet.

Die folgenden Beispiele sollen zur Verdeutlichung dienen. Es wird von einer log-Datei ausgegangen, in der nach Datumsangaben im folgenden Format „Sun Jun 19 17:36:57.844376 2022" gesucht werden soll. Das Datum enthält jeweils gleich lange Informationen wie Wochentag, Monat usw.

```
...
[Sun Jun 19 17:36:57.844376 2022] [mpm_event:notice] [pid 15861:tid
140281039494208] AH00489: Apache/2.4.41 (Ubuntu) configured -- resuming
normal operations
[Sun Jun 19 17:36:57.844489 2022] [core:notice] [pid 15861:tid
140281039494208] AH00094: Command line: '/usr/sbin/apache2'
```

Ein regulärer Ausdruck könnte so aussehen:

```
/(Sun|Mon|Tue|Wed|Thu|Fri|Sat)\s(Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|
Nov|Dec)\s\d{2}\s\d{2}:\d{2}:\d{2}.\d{6}\s\d{4}/
```

Die Aufzählung der Wochentage und Monate ist aufwendig und ggf. auch fehleranfällig, da hier von englischer Schreibweise ausgegangen wurde. Man könnte dies kürzen, indem man die Aufzählung durch „\w" ersetzt. Hier könnten allerdings auch beliebige Wörter eingesetzt werden.

```
/\w{3}\s\w{3}\s\d{2}\s\d{2}:\d{2}:\d{2}.\d{6}\s\d{4}/
```

Der vorangegangene Ausdruck würde auch „TIP TOP 24 17:36:57.844376 2022" finden, da „\w{3}" jedes beliebige drei Buchstaben lange Wort erlaubt.

Testen von regulären Ausdrücken

Die Seite https://regexr.com/ kann genutzt werden, um reguläre Ausdrücke zu testen. Die hier vorgestellten Parameter stellen nur einen sehr kleinen Teil von regulären Ausdrücken dar.

Reguläre Ausdrücke können in Verbindung mit weiteren Programmen sehr mächtig werden. Zu den wichtigsten Programmen dieser Art zählen „grep", „sed", „cut" und „awk". In Verbindung mit den vorgestellten Pipelines ergeben sich viele Möglichkeiten: Mit „grep" (global/regular expression/print) kann man Dateien oder auch Streams nach regulären Ausdrücken durchsuchen. Der allgemeine Aufruf sieht wie folgt aus:

```
grep [Optionen] Suchbegriff [Datei bzw. zu durchsuchender Bereich]
Beispiel:
grep -i schmitz
```

Als Suchbegriff wird hier „schmitz" verwendet wobei „-i" die Groß- und Kleinschreibung ignoriert. Es kann selbstverständlich auch ein regulärer Ausdruck verwendet werden. Möchte man beispielsweise in einem Ordner mit sehr vielen log-Dateien eines Webservers die Zugriffe auf eine bestimmte Unterseite (impressum.html) herausfinden, so könnte man folgenden Ausdruck nutzen:

```
grep -H -n -r impressum.html /var/log/*
```

Die Optionen „-H" und „-n" geben zusätzliche Informationen zur Fundstelle aus. Die Option „-H" veranlasst „grep" den Dateinamen anzuzeigen und „-n" die Zeilennummer der Fundstelle. Die Option „-r" durchsucht den aktuellen Ordner rekursiv, also auch die Unterordner.

```
grep -H -n -r impressum.html /var/log/*

Ausgabe: (Dateiname:Zeile:Inhalt der Zeile)
/var/log/apache2/access.log:4:127.0.0.1 - - [19/Jun/2022:17:46:46 +0200]
"GET /impressum.html HTTP/1.1" 200 3477 "-" "Mozilla/5.0 (X11; Ubuntu;
Linux x86_64; rv:101.0) Gecko/20100101 Firefox/101.0"
/var/log/apache2/access.log:5:192.168.0.43 - - [19/Jun/2022:17:48:03
+0200] "GET /impressum.html HTTP/1.1" 200 3477 "-" "Mozilla/5.0 (X11;
Ubuntu; Linux x86_64; rv:101.0) Gecko/20100101 Firefox/101.0"
/var/log/apache2/access.log:6:192.168.0.43 - - [19/Jun/2022:17:48:03
+0200] "GET /icons/ubuntu-logo.png HTTP/1.1" 200 3623
"http://192.168.0.43/impressum.html" "Mozilla/5.0 (X11; Ubuntu; Linux
x86_64; rv:101.0) Gecko/20100101 Firefox/101.0"
/var/log/apache2/access.log:7:192.168.0.43 - - [19/Jun/2022:17:48:03
+0200] "GET /favicon.ico HTTP/1.1" 404 490 "http://192.168.0.43/impressum.
html" "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:101.0) Gecko/20100101
Firefox/101.0"
...
```

grep-Optionen	
Option	**Beschreibung**
-n	gib die Zeilennummer der Fundstelle aus
-H	gib den Dateinamen der Fundstelle aus
-r	folge dem Dateipfad rekursiv (alle Unterordner)
-v	inverse Suche (suche Zeilen, in denen der Begriff nicht enthalten ist)
-w	ganzes Wort suchen
-i	ignoriere Groß- und Kleinschreibung bei Suche
^SUCHBEGRIFF	am Anfang der Zeile; z. B. ^Hier, Zeile beginnt mit „Hier“
SUCHBEGRIFF$	am Ende der Zeile; z. B. dort$, Zeile endet mit „dort“

Kennt man den Aufbau der Ausgabe, so kann man diese weiter verfeinern. Die Ausgabe enthält zunächst den vollständigen Dateipfad gefolgt von der Zeilennummer und dem Inhalt der Fundstelle. Im obigen Beispiel folgt nun die aufrufende IP-Adresse (hier: 127.0.0.1 bzw. 192.168.0.43) sowie der Aufrufzeitpunkt. Mit dem Befehl „awk“ kann man diese IP-Adressen inkl. dem Aufrufzeitpunkt in eine neue Datei schreiben und zwar so, dass die IP-Adresse, das Datum und die Uhrzeit in eigene Spalten geschrieben werden.

```
grep -H -n -r impressum.html /var/log/apache2/access.log | awk -F"[\
[:' ']" '{print "filename;Line;IP;Date;Time"}{print $1";"$2";"$3";"$6";
"$7":"$8":"$9}' > /tmp/wm/test.csv
```

Neben den Funkionen des reinen Suchens und der Filterung nach bestimmten Begriffen kommt es häufig vor, dass bestehende Dateien geändert werden müssen. Soll dies in einem Skript durchgeführt werden, so kann hierzu der Streameditor „sed“ (von **Stream**e**d**itor) verwendet werden. Ein Streameditor kann neben dem reinen Suchen (siehe „grep“) auch gefundene Stellen ändern (Search and Replace) oder Zeichen einfügen bzw. löschen. Der allgemeine Aufruf von „sed“ sieht wie folgt aus:

```
sed [Optionen] {Skript/Kommandos} [Datei bzw. Input aus Pipe]
Beispiel
sed -n '3,5p' datei.txt
```

Das Beispiel würde die Zeile 3 bis Zeile 5 der Datei datei.txt im Terminal ausgeben. Die Option „-n“ verhindert, dass der Arbeitspuffer ausgegeben wird. Es wird somit nur das Ergebnis des Kommandos angezeigt. Der Bereich „3,5p“ gibt zunächst die sogenannte Adresse (Zeile 3 bis 5) und im Anschluss das Kommando (p für print) an. Mit „datei.txt“ wird hier explizit eine Datei als Input angegeben. Der „sed“-Befehl könnte auch innerhalb einer Pipe verwendet werden, dann würde „stdin“ als Input genutzt werden.

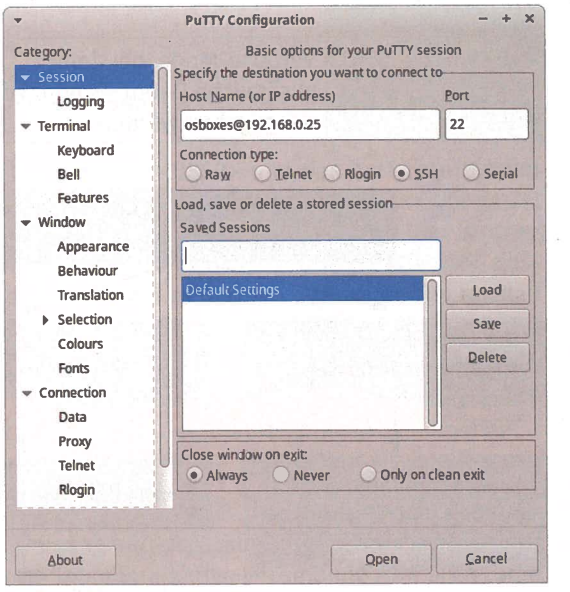
Beispiel: In einer strukturierten Datei „datei.txt“ soll das Kürzel „BAY“ durch „Bayern“ ersetzt werden, aber nur in den Zeilen, in denen die PLZ mit 81 beginnt. Zusätzlich möchte man die erste Zeile als Überschrift ausgeben.

```
sed -n -e '1p' -e '/81/ s/BAY/Bayern/p' datei.txt
```

Die Option „-e“ steht für „execute“ und weist „sed“ an, den anschließenden Bereich in ' ' als Kommando auszuführen. „1p“ druckt die erste Zeile (hier: mit „1“ als Adresse und „p“ als Kommando „print“). Damit wird die Überschriftenzeile ausgegeben. „/81/“ (81 als Adresse) und s/ ersetzt (Kommando „s/“ wie substitute) den Text „BAY“ durch „Bayern“. Das anschließende „/p“ sorgt dafür, dass das Ergebnis des Befehls ausgegeben wird. Die Datei „datei.txt“ wird dabei in diesem Beispiel nicht geändert, da der print-Befehl das Ergebnis lediglich in „stdout“ ausgibt. Will man das Ergebnis dauerhaft speichern, so muss man die Ausgabe in eine Datei, z. B. „ergebnis.txt“ mittels „> ergebnis.txt“, umleiten. Die ursprüngliche Datei bleibt weiterhin unverändert.

(5) Remote-Zugriff mit SSH

Beim Zugang zu einem entfernten System sollte dies immer über eine gesicherte Verbindung geschehen. Hierzu können VPN-Verbindungen über das Internet verwendet werden. Allerdings verschlüsselt eine VPN-Verbindung nur bis zum jeweiligen Tunnelendpunkt. Dies ist meist ein VPN-Gateway. Die Verbindung bis zum Gerät, das konfiguriert werden soll, muss zusätzlich geschützt werden. Ansonsten könnte ein Benutzer im lokalen Netzwerk des zu konfigurierenden Gerätes die administrativen Zugangsdaten erspähen.



Putty-Konfiguration

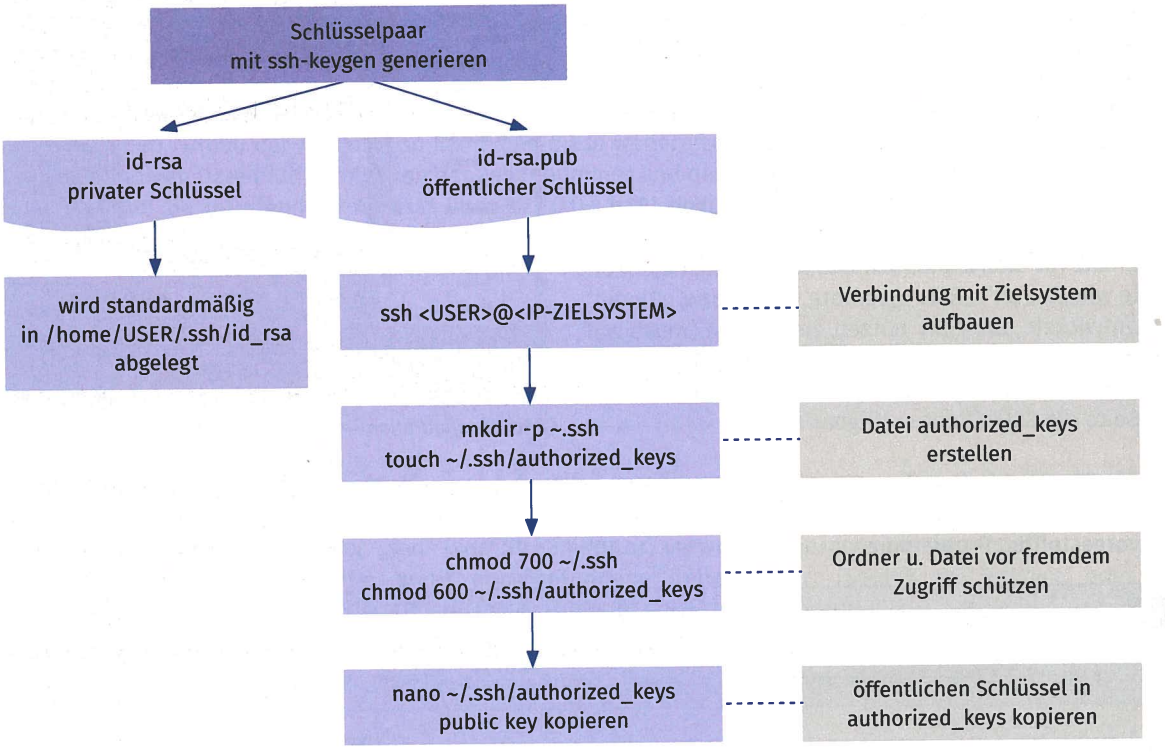
Zu diesem Zweck kann das SSH-Protokoll (SSH = Secure Shell) verwendet werden. Dieses wurde als Ersatz für ältere und nicht verschlüsselte Protokolle wie „Telnet“, „RSH“ („remote shell“) bzw. „Rlogin“ entwickelt. Um SSH nutzen zu können, muss auf dem Zielsystem ein entsprechender SSH-Dienst (Server) konfiguriert werden und auf der lokalen Seite ein SSH-Client verfügbar sein.

Die SSH-Verbindung kann im Linux-System in einem Terminal durch die Eingabe des folgenden Befehls erfolgen. Auf einem Windows-System kann das Tool „putty“ verwendet werden. Dieses umfasst alle hier vorgestellten Programme.

```
ssh [OPTION] [Zielsystem] [COMMAND]
Beispiel:
ssh <USER>@<REMOTE-HOST>
```

Das Zielsystem wird häufig direkt mit <USER>@<REMOTE-HOST> angegeben. Der Benutzer <USER> muss dabei auf dem entfernten System (<REMOTE-HOST> als IP oder FQDN) angelegt sein. Wird erstmalig eine Verbindung zu einem entfernten System aufgebaut, muss der Fingerprint bestätigt werden. Der Fingerprint ist dabei der Hash-Wert des öffentlichen Schlüssels. Bei Unachtsamkeit können hier Sicherheitslücken entstehen, wenn fälschlicherweise ein unbekanntes System als sicher in die lokale Datenbank (unter Linux: „/home/USER/.ssh/known_hosts") aufgenommen wird. Im Anschluss wird das Passwort des Benutzers <USER> abgefragt.
Weitere nützliche Optionen sind in der folgenden Tabelle aufgeführt.

Weitere Optionen für das Programm SSH		
Option	**Bedeutung**	**Einsatzbereich**
-B <remote-IP-Address>	bindet die Verbindung auf eine bestimmte Schnittstelle (IP-Adresse) auf dem Zielsystem	wenn der Server, z. B. über zwei Schnittstellen, erreicht werden kann, aber nur über eine bestimmte konfiguriert wird
-b <local-IP-Address>	bindet wie -B die Verbindung auf eine IP-Adresse, aber auf dem lokalen System	wie bei -B; hier muss allerdings auf der lokalen Seite die Schnittstelle vorgegeben werden.
-i <identity_file>	ein identity_file wird als Private Key verwendet	Wenn man pro Zielsystem ein eigenes Schlüsselpaar verwendet, muss hier die Auswahl getroffen werden.
-o option	gibt eine Option in einer Konfigurationsdatei an	Wenn man häufig auf vielen Zielsystemen mit unterschiedlichen Optionen arbeitet, kann man diese in einer Konfigurationsdatei ablegen.
-v	verbose-Mode, bei dem zusätzliche Debug-Meldungen ausgegeben werden	Wenn man Verbindungs-, Anmelde- oder Konfigurationsprobleme hat, kann man hierüber hilfreiche Informationen erhalten.



Ablauf beim Kopieren auf das Remote-System

Arbeitet man häufig auf denselben Systemen, dann kann man alternativ zur Passworteingabe ein Schlüsselpaar (Private/Public Key) für den Benutzer auf dem entfernten System mit „ssh-keygen“ erstellen und den öffentlichen Schlüssel „id_rsa.pub“ auf das Zielsystem kopieren.

Ist die Verbindung erfolgreich aufgebaut, kann über die SSH-Verbindung so gearbeitet werden, als ob man direkt am entfernten System ein Terminalfenster benutzen würde. Aus Sicherheitsgründen sollte auf dem Zielsystem der direkte Zugriff über den Benutzer „root“ ausgeschaltet werden. Wird ein administrativer Zugriff benötigt, ist dies bei den meisten Systemen über „sudo -s“ erreichbar. Hierzu muss der betreffende Benutzer in die „sudoers“-Gruppe aufgenommen sein.

(6) Sichere Kopiervorgänge

Neben dem eigentlichen Terminalzugang gehört zu einer SSH-Installation die Möglichkeit, mittels der Befehle „scp“ (secure copy protocol) oder „sftp“ (secure file transfer protocol) Dateien von bzw. zu einem entfernten System zu transferieren. Beide nutzen „ssh“ zur Verschlüsselung der Nutzdaten. Daher muss eine SSH-Verbindung wie beschrieben grundlegend eingerichtet sein. Mit dem Programm „scp“ können Dateien direkt kopiert werden. Die folgenden Befehle zeigen die Kopiervorgänge in beide Richtungen. Dabei wird für den Dateitransfer kurzzeitig eine SSH-Verbindung aufgebaut.

```
1 scp [OPTION] [Quelle] [Ziel]
2
3 # Beispiel: Kopie von local nach remote
4 scp datei.txt <USER>@<REMOTE-HOST>:/home/<USER>/Downloads
5 scp datei.txt charlie@192.168.0.2:/home/charlie/Downloads
6
7 # Beispiel: Kopie von remote nach local
8 scp <USER>@<REMOTE-HOST>:/home/<USER>/Downloads/datei.txt
9 scp charlie@192.168.0.2:/home/charlie/Downloads/datei.txt
```

Beim entfernten System müssen jeweils ein berechtigter Benutzer <USER> und das entfernte System <REMOTE-HOST> angegeben werden. Dabei kann entweder eine bekannte IP-Adresse oder der FQDN verwendet werden. Das erste Beispiel (Zeilen 3–5) kopiert die Datei „datei.txt“ in den Ordner „/home/<USER>/Downloads“ (<USER> entspricht dabei einem aktiven Nutzer auf dem entfernten System). Im zweiten Beispiel (Zeilen 7–9) wird die Datei „datei.txt“ vom entfernten System in den lokalen Ordner, aus dem „scp“ aufgerufen wurde, kopiert. Zu beachten ist der Doppelpunkt zur Trennung der Anmeldeinformation von den Dateisysteminformationen.

> **!** Bei „scp“ muss man den Dateipfad auf dem entfernten System ohne Unterstützung exakt angeben. In Shell-Skripten können die Pfade in Variablen abgelegt werden.

Auf Systemen, deren Dateisystem unbekannt ist, ist der Umgang mit „scp“ allerdings schwierig. Hier hilft der Befehl „sftp“. Mit „sftp“ kann man ähnlich wie mit „ssh“ eine interaktive Verbindung zum entfernten System aufbauen und im Dateisystem navigieren sowie Dateien zwischen dem lokalen und entfernten System transferieren.

```
sftp [OPTION] Ziel

Beispiel:
sftp charlie@192.168.0.2

sftp> ?
```

Sobald die Verbindung erfolgreich aufgebaut wurde, kann man mit einem eingeschränkten Befehlssatz interaktiv mit Dateien arbeiten. Mit „?“ erhält man eine Liste der möglichen interaktiven Befehle.

Interaktive Befehle von sftp	
Befehl	**Bedeutung**
ls	zeigt den Inhalt des aktuellen Ordners auf dem entfernten System an (list)
lls	zeigt den Inhalt des aktuellen Ordners auf dem lokalen System an (local list)
cd	wechselt den Ordner auf dem entfernten System (change directory)
lcd	wechselt den Ordner auf dem lokalen System (local change directory)
mkdir	erstellt neuen Ordner auf entferntem System (make directory)
put	kopiert vom lokalen zum entfernten System
get	kopiert vom entfernten zum lokalen System
quit	sftp-Verbindung beenden

Eine weitere Möglichkeit, Dateien sicher auf ein entferntes System zu transferieren, ist mit „FTPS" (FTP mit TLS nach RFC 2228). Grundsätzlich ist „FTPS" allerdings aufwendiger, da zunächst ein eigener FTP-Server zu installieren ist und weiterhin das TLS-System als Verschlüsselung. Ein solcher Server hat einen eigenen Dateiablagebereich, sodass kein Transfer an eine beliebige Stelle im Dateisystem möglich ist. Sofern der Benutzer über die entsprechende Berechtigung verfügt, kann man mittels „ssh" Dateien an jede beliebige Stelle im Dateisystem transferieren. Weiterhin kann man mittels „FTPS" nur Dateien übermitteln.

Gegenüberstellung der Dateitransferprotokolle	
Protokoll	**Eigenschaften**
SCP	• nutzt Port 22 (SSH) zur Übertragung • interaktiver Dateitransfer (cd, ls, put/get) • Zielordner nur durch Rechte des Benutzers eingeschränkt • nutzt eine Verbindung zum entfernten System • gut gegeignet für Shell-Skripte
SFTP	• nutzt Port 22 (SSH) zur Übertragung, hat demnach nichts mit FTP zu tun • interaktiver Dateitransfer (cd, ls, put/get) • Zielordner nur durch Rechte des Benutzers eingeschränkt • nutzt eine Verbindung zum entfernten System • gut geeignet bei unbekannten Systemen
FTPS	• nutzt Port 990 bzw. 989 (bei implizitem Aufbau, sonst 21/20) • interaktiver Dateitransfer (cd, ls, put/get) • Dateiablage nur im FTP-Server • nutzt wie FTP zwei Verbindungen zum entfernten System (command und data channel)
FTP	• nutzt Port 21 und 20 • Dateitransfer ist nicht verschlüsselt • Dateiablage nur im FTP-Server • nutzt zwei Verbindungen zum entfernten System (command und data channel)

(7) Informationsquellen (stackoverflow, github)

Bei der Erstellung von Skripten sollte man auf Informationsquellen zurückgreifen, um schneller zu Lösungen zu gelangen. Hierbei ist allerdings das Urheberrecht zu beachten, sofern das eigene Ergebnis veröffentlicht wird.

Informationsquellen für Skripiting	
Plattform	**Beschreibung**
stackoverflow.com	Stackoverflow stellt eine der größten Plattformen für den Austausch zwischen Entwicklern dar. Es können Fragen zu beliebigen Programmiersprachen gestellt werden. Meist muss keine eigene Frage gestellt werden, da bereits ein Thread zu einer ähnlichen Frage existiert.
ubuntuusers.de	Deutschsprachige Debian/Ubuntu-Plattform mit Anleitungen und Informationen zu vielen Linux-Programmen. Die Plattform enthält sehr viele Anleitungen zur Installation von Software und deren Konfigurationsdateien. Ergänzt wird die Plattform durch ein Forum.
github.com/gitlab.com	Plattformen mit Unterstützung von verteilten Entwicklungen. Beide bieten die Möglichkeit, git-Repositorys zu hosten. Darüber hinaus können für komplexere Projekte CD/CI-Systeme aufgesetzt werden. GitLab kann auch als On-Premise-Installation eingesetzt werden. Es sind bereits tausende Projekte gehostet, die häufige Aufgabenstellungen abdecken. Neben Softwareentwicklung findet man auf beiden Plattformen auch reine Lerninhalte. So befindet sich auf github ein kompletter Kurs zu „Microsoft Azure Administrator" mit mehreren Lektionen.
docs.microsoft.com/ social.technet. microsoft.com	Auf den Webseiten von Microsoft findet man technische Dokumentationen und Foren zu allen Microsoft-Produkten. Die Skriptsprache „PowerShell" ist ebenfalls vertreten.
powershellgallery.com	Zentrales Repository von Microsoft; auf der Plattform findet man PowerShell-Skripte und -Module. Meistens sind diese auf github gehostet und könnten auch über diesen Weg bezogen werden. Diese Seite enthält aber nur PowerShell-Ressourcen, sodass gezielter gesucht werden kann.

Auf den aufgeführten Webseiten finden sich Lösungen für Teilprobleme oder vor allem für Fehlermeldungen, die während der Erstellung von Skripten aufkommen. Wenn nach Fehlermeldungen gesucht wird, sollte der exakte Fehlertext zur Suche verwendet werden. Bei den Lösungsvorschlägen muss allerdings immer abgewogen werden, ob diese zweckmäßig sind und bereits erfolgreich erprobt wurden. Im Zweifelsfall sollten die Lösungen zunächst an Testsystemen angewendet werden.

(8) Versionierung

Im Bereich der Administration von Unix-ähnlichen Servern werden häufig Konfigurationsdateien im Textformat (direkt mit einem Texteditor lesbar) eingesetzt. Bei Windows-Servern werden die Konfigurationsdaten meist entweder in der Registry oder in einem Binärformat (nicht direkt lesbar) gespeichert. Auch die eigenen Skripte werden im Textformat gespeichert. Nach dem Speichern einer solchen Datei ist der vorherige Zustand nicht mehr ersichtlich, somit kann die durchgeführte Änderung nicht mehr nachvollzogen werden. Insbesondere bei Fehlern ist dies ungünstig, da die Stelle des Fehlers nicht mehr ersichtlich ist. Eine gut gepflegte Versionierung, also eine nachvollziehbare Speicherung alter Zustände, kann hier Abhilfe schaffen. Im Bereich der Softwareentwicklung ist dies bereits seit Jahrzehnten Standard (siehe dazu auch Jahrgangsband 2, Lernfeld 7, Kapitel 2.4.2).

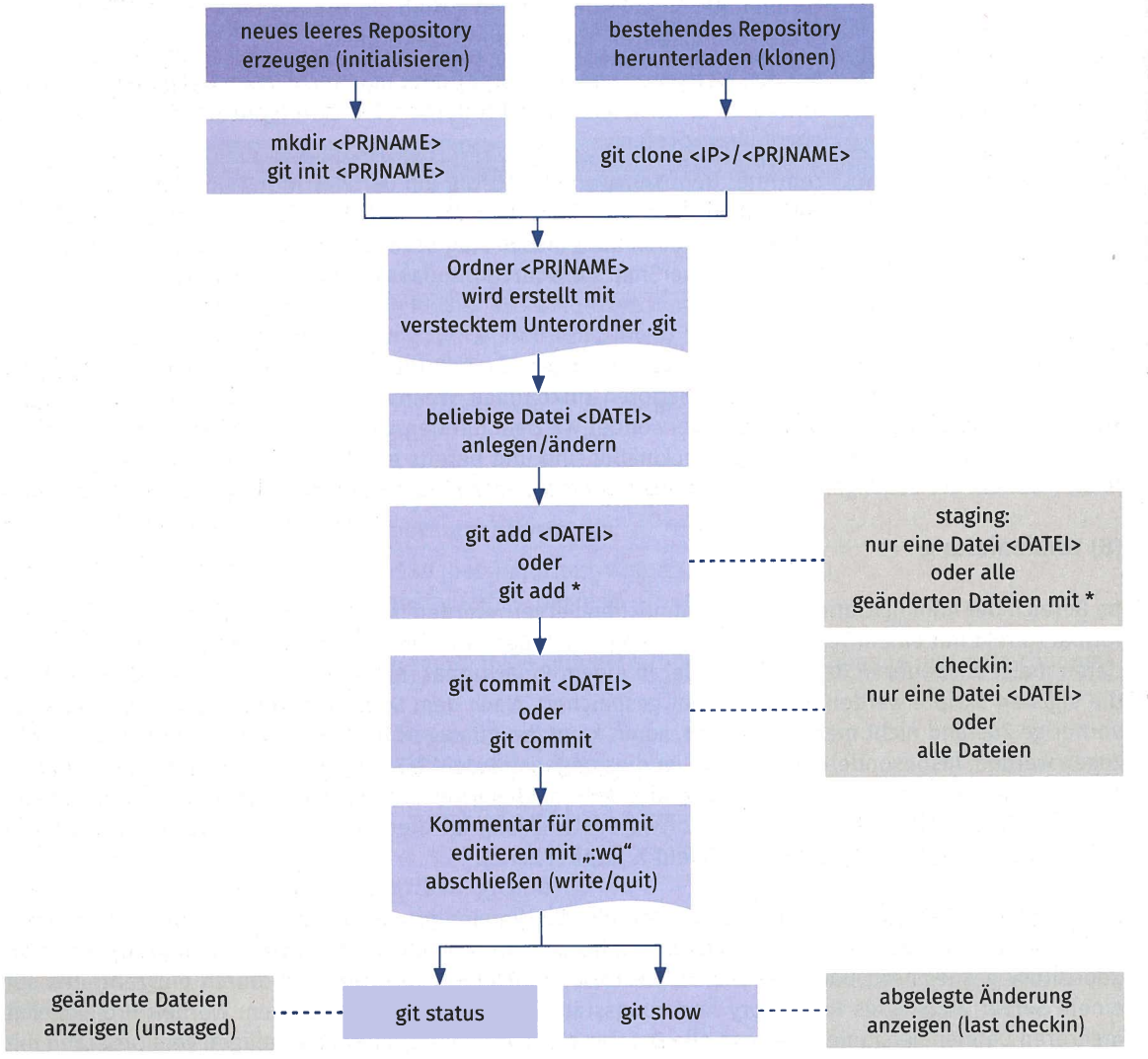
Linus Torvalds hat zur Pflege des Linux-Kernels das Versionierungssystem „git" entwickelt, welches besonders auf verteiltes Arbeiten ausgelegt ist. Bei „git" werden alle Änderungen in einem sogenannten Repository gespeichert. Dabei können Repositories nur lokal existieren oder durch ein zentrales auf einem Server abgelegtes Repository Änderungsstände miteinander austauschen. Werden Projekte mit mehreren Dateien versioniert, können alle Dateien eines Projektes in ihrem jeweiligen Versionsstand mit einem sogenannten „Tag" markiert werden. Diese Funktion ist in der Softwareentwicklung besonders wichtig, wenn ein großes Projekt für die Veröffentlichung freigegeben werden soll.

Um die Versionierung mittels „git“ zu nutzen, ist zunächst die Software selbst zu installieren. Es stehen Versionen für alle gängigen Betriebssysteme bereit. Unter Linux kann „git“ über „apt“ oder „yum“ installiert werden. Für Windows und macOS kann man die Installationsdatei von der offiziellen Seite (https://git-scm.com) herunterladen und installieren. In der Standardinstallation ist „git“ in der Kommandozeile direkt nutzbar.

Für jeden Versionierungsschritt verlangt „git“ die Angabe des Benutzers (Name und E-Mail), der gerade eine Version ablegt. In „git“ wird dieser Vorgang als „commit“ bezeichnet. Bei anderen Versionierungssystemen wird hierfür der Begriff „checkin“ verwendet. Damit „git“ nicht bei jedem dieser Vorgänge nach einem Benutzer fragt, kann man diese Information zentral im Benutzerordner in der Datei „.gitconfig“ ablegen. Die geschieht mit den folgenden Befehlen:

```
git config --global user.name „Charlie“
git config --global user.email „charlie@abcxyz.com“
```

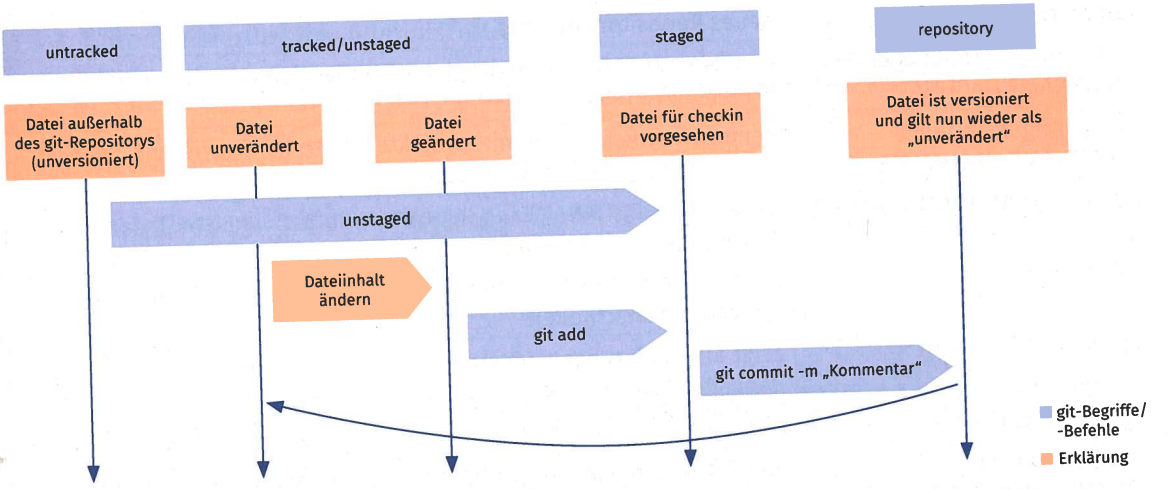
Auf diesem Weg können weitere Optionen für alle Projekte hinterlegt werden.



Ablauf der Repository-Erstellung

Im Projektordner, in dem alle zu versionierenden Dateien abgelegt sind, führt man nun **„git init“** aus. In der Ordnerstruktur wird ein neuer Unterordner „.git“ angelegt, in dem „git“ alle zur Verwaltung notwendigen Daten anlegt. Hier wird die Datei „config“ angelegt, in der ggf. von der globalen Konfiguration (.gitconfig) abweichende Daten für das aktuelle Projekt abgelegt werden. Ein „git init“ kann in einem leeren Ordner ausgeführt werden oder in einem Ordner, der bereits zu versionierende Dateien enthält. Die vorherige Abbildung zeigt den grundsätzlichen Ablauf bei neuen Repositories.

Eine Datei kann in einem Repository verschiedene Stadien durchlaufen. Nachdem das Repository durch den Befehl „git init“ angelegt wurde, müssen ihm zunächst die zu überwachenden Dateien (tracking) bekannt gemacht werden. Dies geschieht durch den Befehl „git add <DATEI>“. Die Datei befindet sich damit im sogenannten Staging-Bereich. In diesem Bereich sind Dateien für das eigentliche Ablegen im Repository („checkin“ mittels „commit“) notiert. Durch den Befehl „git commit -m 'Mein Kommentar'“ wird für die Datei eine Prüfsumme ermittelt und die Änderung zum letzten „commit“ im Repository abgespeichert. Die Datei gilt nach einem erfolgreichen „commit“ als „unverändert“ und befindet sich wieder im „unstaged“-Bereich.



Workflow für Änderungen einer Datei

Mit dem Befehl „git status“ werden alle Dateien, die geändert wurden und deren Änderungen somit noch nicht im Repository gesichert wurden, aufgelistet. Im folgenden Beispiel wird die Datei „test.txt“ als geändert angezeigt.

```
~/gittest (git)-[master] % git status
Auf Branch master
Änderungen, die nicht zum Commit vorgemerkt sind:
 (benutzen Sie "git add <Datei>...", um die Änderungen zum Commit vorzumerken)
 (benutzen Sie "git restore <Datei>...", um die Änderungen im Arbeitsverzeichnis zu verwerfen)
 geändert: test.txt

keine Änderungen zum Commit vorgemerkt (benutzen Sie "git add" und/oder "git commit -a")

ODER nach git add
~/gittest (git)-[master] % git add test.txt
~/gittest (git)-[master] % git status
Auf Branch master
Zum Commit vorgemerkte Änderungen:
 (benutzen Sie "git restore --staged <Datei>..." zum Entfernen aus der Staging-Area)
 geändert: test.txt
```

Eine der Stärken eines Versionierungssystems besteht in der Möglichkeit, die Versionen einer Datei miteinander zu vergleichen. Wurde eine Datei mittels „git add“ gerade in den Staging-Bereich aufgenommen, so kann man sich mit „git diff --cached“ die zuletzt gemachte Änderung anzeigen lassen. Dabei werden

hinzugefügte Zeilen mit + und entfernte Zeilen mit - gekennzeichnet. Im folgenden Beispiel wurde der Text „Weitere Änderung" neu in die Datei „test.txt" geschrieben.

```
/gittest (git)-[master] % git diff --cached
diff --git a/test.txt b/test.txt
index ffdfca3..74cb23f 100644
--- a/test.txt
+++ b/test.txt
@@ -1,2 +1,3 @@
+Weitere Änderung
 Mein neuer Text
```

git-Befehlsübersicht	
Befehl	**Bemerkung**
git init	neues Repository anlegen
git config --global user.name "VORNAME NACHNAME"	Benutzername für „commit" global konfigurieren („user.email" analog für E-Mail-Adresse)
git config user.name "VORNAME NACHNAME"	Benutzername für „commit" projektbezogen konfigurieren („user.email" analog für E-Mail-Adresse)
git add <DATEINAME>	neue Datei <DATEINAME> in Repository aufnehmen bzw. Datei in Zustand „staged" bringen
git status	zeigt alle geänderten Dateien an; dabei wird zwischen „unstaged" und „staged" unterschieden
git diff <DATEINAME>	zeigt alle Änderungen der Datei <DATEINAME> zum letzten Stand im Repository an
git diff --cached	zeigt alle Änderungen der Dateien im Zustand „staged" an
git commit -m "MEIN KOMMENTAR"	alle Dateien im Zustand „staged" werden in der aktuellen Version mit dem Kommentar (-m) ins Repository geschrieben
git status	zeigt alle geänderten Dateien unterschieden nach „staged" (für „commit" vorgesehen), „unstaged" (nicht für „commit" vorgesehen) und „untracked" (nicht im Repository aufgenommen) an

Durch die Versionierung ist es möglich, auch nach einer längeren Zeit alle Veränderungen an einem System nachzuvollziehen. Hier ist auf Datensicherung des zentralen Repository zu achten. Durch einfache Skripte lässt sich die Versionierung mit dem Ausrollen der Konfiguration auf die eigentlichen Server verknüpfen. Dies reduziert Fehlerquellen in der Bedienung, da keine Schritte vergessen werden können. Insbesondere in großen Installationen mit vielen Servern kann man so den Überblick über die gesamte Anlage behalten.

Neben dem hier vorgestellten „git" können auch andere Versionierungssysteme zum Einsatz kommen. Zu den bekanntesten gehören Subversion (SVN), Mercuial oder Concurrent Versions System (CSV/OpenCSV, Vorgänger von SVN). Zu beachten ist hier, dass einige Systeme anders als „git" einen zentralen Server voraussetzen.

Kompetenzcheck

1. Geben Sie an, welche Aussagen richtig sind.
   a) Compiler übersetzen zur Laufzeit den Quellcode in ausführbaren Binärcode.
   b) Python und Perl sind Skriptsprachen.

c) Die „Shell“ kommuniziert über „stdin“, „stdout“ und „stderr“ mit dem Benutzer.
d) Der Inhalt der Variable „$filename“ kann in der „Shell“ mit „print $filename“ ausgegeben werden.
e) Mit regulären Ausdrücken (RegEx) können Texte nach vorgegebenen Mustern gefiltert werden.
f) Der reguläre Ausdruck „/Haus[tier|schuh|halt]/“ findet die folgenden Begriffe in einem Text: Haustier ODER Hausschuh ODER Haushalt.
g) Für den Remote-Zugriff sollte „telnet“ verwendet werden.
h) Die Protokolle „sftp“ und „FTPS“ unterscheiden sich nur darin, ob vor- bzw. nachher verschlüsselt wird.
i) Bei der Entwicklung sollte man unter Berücksichtigung des Urheberrechts auf externe Informationsquellen zurückgreifen, um Teilprobleme schneller lösen zu können.
j) Zur Versionierung von Quellcode können Kopien der Projektordner angelegt werden, damit Vergleiche mit alten Versionsständen ermöglicht werden.

2 Bearbeiten Sie die Aufgaben 1, 2 und 3 der Lernsituation 5 im Arbeitsbuch. A1 A2 A3

1.5.2 Automatisierung der Administration

S Sie sollen Möglichkeiten zur Automatisierung der Administration kennenlernen.

(1) Konfigurationsmanagement

In Rechenzentren wird häufig mit einer großen Anzahl von Servern gearbeitet, die sehr ähnliche Grundkonfigurationen haben. Nach der ersten Inbetriebnahme sind regelmäßige Arbeiten notwendig. Eine weitere Aufgabe ist die nachträgliche Änderung von Konfigurationen. Bei all diesen Tätigkeiten muss die Integrität der Funktionalität stets gewahrt bleiben, da es ansonsten zu Ausfällen oder Fehlern kommen kann.

Regelmäßige Arbeiten an der Serverinfrastruktur	
Aktion	**Bemerkung**
Softwarepakete aktualisieren	Fehler oder Sicherheitslücken können nur durch aktuelle Softwarestände verhindert werden.
Backup	Sicherungen können Datenbanken, Konfigurationsdateien oder sonstige Nutzdaten umfassen.
Auswertung von log-Dateien	Fehlverhalten oder ungewöhnliche Zugriffe können über die Auswertung von log-Dateien festgestellt werden.
Änderungen der Konfiguration	Erweiterungen oder Anpassungen der Funktionalität eines Server, z. B. das Bereitstellen einer neuen Domäne

Konfigurationsmanagement

Konfigurationsmanagement konzentriert sich auf das Einrichten und Bewahren der Integrität der Funktionalität von IT-Systemen. Es umfasst die Ersteinrichtung, Änderung und Überwachung der Konfiguration von IT-Systemen während ihrer gesamten Lebensdauer.

Theoretisch kann man zu jedem Server eine eigene Remote-Verbindung aufbauen und diese Arbeiten händisch ausführen. Dies ist fehleranfällig und sehr zeitaufwendig. Darüber hinaus werden die Tätigkeiten nicht protokolliert. Diese Vorgehensweise bei der Administration von Servern ist somit nur für kleinere Systeme sinnvoll. Bei manueller Konfiguration, z. B. über Weboberflächen oder in Anwendungsoberflächen, ist so kaum sicherzustellen, dass viele identische Server in Betrieb genommen werden. Dies kann in der Folge zu sogenannten Snowflake-Servern führen.

Snowflake-Server

Als Snowflake-Server bezeichnet man einen sehr individuellen Server (einzigartig wie eine Schneeflocke). Ein Snowflake-Server könnte aufgrund der meist undokumentierten Änderungen (z. B. in einer GUI) selten mit identischer Funktionalität auf einer neuen Hardware in Betrieb genommen werden.

Eine Möglichkeit, Konfigurationsarbeiten zu automatisieren, besteht in der Nutzung von Automatisierungs-Frameworks, welche die IT-Systeme automatisiert in Betrieb nehmen und weiterführend konfigurieren. Dabei werden keine individuellen Skripte verwendet, sondern allgemein formulierte und nachvollziehbar dokumentierte Konfigurationsdateien. Die folgende Tabelle liefert einen Überblick über verfügbare Frameworks.

Gegenüberstellung möglicher Automatisierungssysteme					
	Ansible	**Chef**	**Puppet**	**SaltStack**	**Terraform**
Konfigurationsmanagement	ja	ja	ja	ja	nein (Orchestrierung)
veränderliche Infrastruktur	ja	ja	ja	ja	nein
Sprache	deklarativ	imperativ/ prozedural	deklarativ	deklarativ	nein (Orchestrierung)
Architektur	nur Client (agentless)	Client/ Server	Client/ Server	nur Client (agentless)	nur Client (agentless)

Terraform hat einen grundlegend anderen Ansatz als die übrigen Systeme, da es von Systemen ausgeht, die nach der erstmaligen Inbetriebnahme unverändert bleiben. Werden hier neue Funktionalitäten benötigt, wird eine neue Instanz definiert, die die bisherige ersetzt. Ein weiterer großer Unterschied liegt in der Architektur.

Mit „Client/Server" ist hier gemeint, dass auf dem Zielsystem ein zusätzlicher Server (meist Agent genannt) vorhanden sein muss, damit die Konfiguration zentral vorgenommen werden kann. Bei den Systemen „nur Client" reicht hingegen eine Remote-Verbindung auf dem Administrationsrechner, meist in Form einer SSH-Verbindung. Somit ist die Inbetriebnahme dieser Systeme einfacher. Als ein Vertreter dieser Systeme wird im Folgenden Ansible exemplarisch vorgestellt.

(2) Ansible

Ansible im Überblick

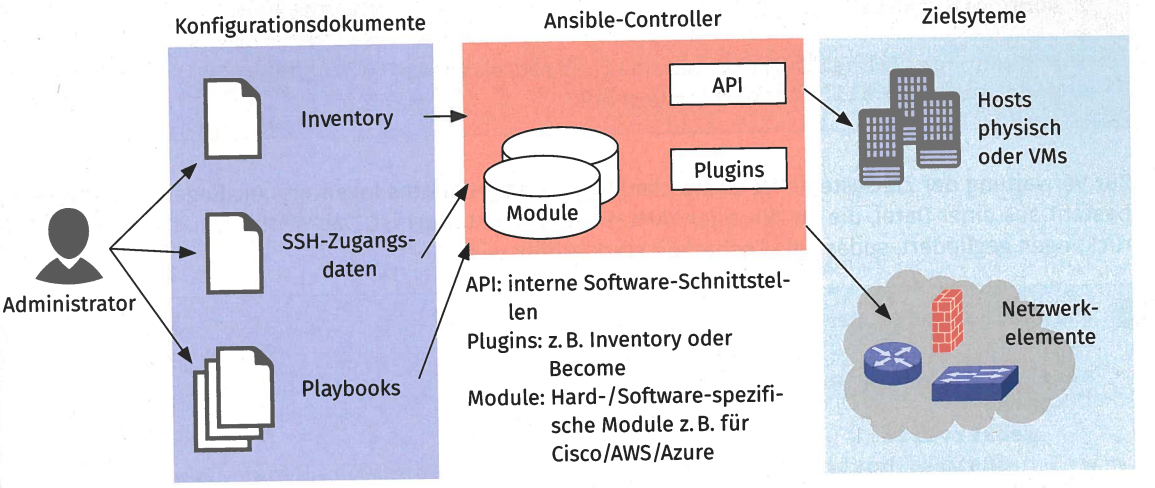
- Erscheinungsjahr: 2012; 2015 von Red Hat übernommen
- Unterstützt prinzipiell alle Unix-basierenden Betriebssysteme; ab 1.7 wird auch Windows mit PowerShell-Befehlen unterstützt.
- wird sehr aktiv weiterentwickelt
- auch gut geeignet für Cloud-Installationen, da eine große Anzahl von Modulen explizit für Cloud-Anbieter existieren (allein über 40 Module für AWS und über 100 für Azure)

Vorteile von Ansible

- benötigt keinen Agent auf dem zu konfigurierenden System (agentless)
- benötigt nur einen SSH-Zugriff

- Aufgaben sind selbsterklärend abgelegt, sodass sowohl Mensch als auch ausführende Maschine diese lesen können
- niedrige Einstiegshürde bzw. leicht zu erlernen

Das Inventory umfasst alle Zielsysteme, die von Ansible verwaltet werden sollen. Dabei können sogenannte Gruppen gebildet werden. Dies ist hilfreich, wenn eine große Anzahl von gleichartigen Servern oder alle Geräte eines bestimmten Standorts angesprochen werden sollen.



Prinzipieller Aufbau des Ansible-Konfigurationssystems

Für den Zugang zu den Zielsystemen benötigt Ansible weiterhin die entsprechenden SSH-Zugangsdaten auf den einzelnen Zielsystemen. Die eigentlichen Arbeitsschritte können direkt an Ansible über Parameter übergeben werden. Wesentlich komfortabler und nachhaltiger ist allerdings die Verwendung der Playbooks. In einem Playbook werden die gewünschten Installationsvorgaben aufgeführt.

Das eigentliche Ansible-Framework besteht aus einer Application Programming Interface (API), die von einem Open-Source-Projektteam weiterentwickelt wird. Der Benutzer verwendet Ansible in Form von Modulen und Plugins. Module stellen die eigentlichen Kernfunktionalitäten von Ansible dar. Die Liste umfasst knapp 20 Module, die zum Kernsystem gehören. Hierzu gehören das Inventory-Modul (Verwaltung der Zielsysteme) und das Become-Modul (um administrative Rechte auf den Zielsystemen zu erhalten). Plugins werden dann benötigt, wenn spezielle Hard-/Software mit Ansible adminstriert werden soll. Ein Beispiel für Hardware-Plugins sind Cisco-IOS-Netzelemente wie Switch oder Router. Ebenso werden beispielsweise für AWS oder Azure eine Reihe von Plugins bereitgestellt. Die Liste der Plugins umfasst über 3000 Einträge.

Vorbereitung der Zielsysteme

Damit Ansible auf die Zielsysteme zugreifen kann, wird ein SSH-Zugang benötigt. Dieser kann mit den folgenden Befehlen auf einem Linux-System eingerichtet werden. Es wird hier von einem Benutzer „devopXX" auf einem System mit der IP-Adresse ausgegangen. Diesem Benutzer werden durch Hinzufügen in die sudo-Gruppe administrative Rechte gegeben (siehe auch Lernfeld 10b, Kapitel 1.5.1).

```
# zu administrierendes Zielsystem mit IP-Adresse <remote-host>
sudo adduser devopXX
sudo usermod -aG sudo devopXX

# Ansible-Controller
ssh-keygen
ssh-copy-id -i id_rsa.pub devopXX@<remote-host>
```

Installation von Ansible
Das Ansible-Framework ist in der Programmiersprache „Python" geschrieben, daher kann die Installation der aktuellsten Version mittels „pip" (package installer for Python) erfolgen. Die folgenden Befehle installieren „pip" (Zeile 1) und im Anschluss Ansible (Zeile 2 und 5). Wenn Ansible nicht systemweit installiert werden soll, kann jeweils die Option „sudo -H" verwendet werden, um die Installation im aktuellen Benutzerordner vorzunehmen.

```
1 sudo apt install python3-pip
2 sudo pip install ansible
3
4 # Nach der Installation, aktualisieren mittels
5 sudo pip install ansible --upgrade
```

Zur Verwaltung der Zielsysteme ist es zweckmäßig, ein sogenanntes Inventory anzulegen. Das Inventory besteht aus einer Datei, die im INI- oder YAML-Format strukturiert ist. Das YAML-Format wird durch Einrückungen gegliedert, sodass die Lesbarkeit stark erhöht wird.

```
# Inhalt: hosts.yaml
---
all:
  hosts:
    webserver25:
      ansible_host: 192.168.0.25
      ansible_user: devop25
    webserver46:
      ansible_host: 192.168.0.46
      ansible_user: devop46

  children:
    webservers:
      hosts:
        webserver25:
        webserver46:
```

Dafür ist das INI-Format kompakter. Im INI-Format sieht das Inventory so aus:

```
# Inhalt: hosts.ini
webserver25 ansible_host=192.168.0.25 ansible_user=devop25
webserver46 ansible_host=192.168.0.46 ansible_user=devop46

[webservers]
webserver25
webserver46
```

Hinweis: Alle weiteren Dateien werden im YAML-Format dargestellt.

Anzeige des Inventorys
Mit „ansible-inventory -i hosts.yaml „--list" bzw. „--graph" kann man sich das Ansible Inventory anzeigen lassen. Mit dem Parameter „--list" werden alle Hosts inklusive der abgelegten Variablen (hier: „ansible_host" und „ansible_user") angezeigt. Der Parameter „--graph" zeigt nur die Gruppierung der Hosts.

Erster Test

Mit dem direkten Aufruf „ansible -i hosts.yaml all -m ping“ kann man die Verbindung zu allen Hosts des Inventorys testen. Es wird dabei nicht nur ein ICMP-Test („ping“) durchgeführt, sondern der erfolgreiche administrative Zugriff auf das Zielsystem geprüft.

```
ansible -i hosts.yaml all -m ping

# Ausgabe:
webserver25 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
webserver46 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Hier wird das Modul „ping“ für die Host-Gruppe „all“ aus dem Inventory „hosts.yaml“ ausgeführt. Es wäre auch möglich, das Modul „ping“ nur für einen Host auszuführen (z. B. „ansible -i hosts.yaml webserver25 -m ping“). Eine häufige Aufgabe ist es, bestimmte Softwarepakete zu installieren. Dies kann man mit dem Module „package“ und den Modulparametern „-a “name=vim state=present“ " durchführen. Der Parameter „-b“ (steht für „become“) wird zur sogenannten „Privilege Escalation“ genutzt, also, um administrative Rechte zu erhalten. In Debian-basierten System wird dies mit dem Befehl „sudo“ erreicht.

Der folgende Ansible-Befehl prüft, ob das Paket „vim“ installiert ist. Fehlt das Paket, so wird es nachinstalliert. Hierzu sind die zusätzlichen Rechte „-b“ notwendig.

```
ansible -i hosts.yaml all -m package -a "name=vim state=present" -b

# Ausgabe
webserver25 | FAILED! => {
    "msg": "Missing sudo password"
}
webserver46 | FAILED! => {
    "msg": "Missing sudo password"
}
```

Der Befehl schlägt fehl, da Ansible das Passwort des SSH-Benutzers benötigt, um innerhalb der SSH-Verbindung zusätzliche sudo-Rechte zu erhalten. Dieses Passwort kann man als zusätzliche Variable im Inventory ablegen. Allerdings stellt dies ein sehr hohes Sicherheitsrisiko dar.

```
# Inhalt: hosts.yaml UNSICHER mit Passwörtern
---
all:
  hosts:
    webserver25:
      ansible_host: 192.168.0.25
      ansible_user: devop25
      ansible_become_password: test123
    webserver46:
      ansible_host: 192.168.0.46
      ansible_user: devop46
      ansible_become_password: test123
```

Die Datei „hosts.yaml" wird auf diese Weise besonders sensibel für Angriffe. Die bessere Variante ist die Nutzung von „ansible-vault". Dabei werden die Passwörter nicht direkt im Inventory gespeichert, sondern in einer gesonderten, verschlüsselten Vault-Datei.

```
# Inhalt: hosts.yaml SICHER mit Variablen aus Vault-Datei
---
all:
  hosts:
    webserver25:
      ansible_host: 192.168.0.25
      ansible_user: devop25
      ansible_become_password: "{{ webserver25_become_pass }}"
    webserver46:
      ansible_host: 192.168.0.46
      ansible_user: devop46
      ansible_become_password: "{{ webserver46_become_pass }}"
```

Statt der tatsächlichen Passwörter werden hier Variablen benutzt. In Ansible werden diese mit doppelten geschweiften Klammern markiert, z. B.: „{{ VARIABLE }}". Die eigentlichen Passwörter werden mittels des Befehls „ansible-vault" in eine eigene Datei (z. B. „vaulted-vars.yaml") abgelegt. Diese Datei wird verschlüsselt und mit einem eigenen Passwort (Vault-Password) gesichert.

```
ansible-vault create vaulted_vars.yaml
# Eingabe des Passworts für die Datei vaulted_vars.yaml
New Vault password: ****
Confirm New Vault password: ****

# Editieren der vaulted_vars.yaml Datei
webserver25_become_pass: test123
webserver46_become_pass: test123
# Speichern und schließen nach vi-Syntax
:wq
```

Der Aufruf für die Installation des „vim"-Pakets sieht nun wie folgt aus:

```
ansible -i hosts.yaml all -m package -a "name=vim state=present" -b
--ask-vault-pass -e@vaulted_vars.yaml
Vault password: ****

# Ausgabe:
webserver25 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1659283110,
    "cache_updated": false,
    "changed": false
}
webserver46 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1659260575,
    "cache_updated": false,
    "changed": false
}
```

Mit der zusätzlichen Variablen „-e@vaulted_vars.yaml" wird die Datei mit den verschlüsselten Passwörtern übergeben. Damit Ansible nach dem Vault-Password fragt, wird der Parameter „--ask-vault-pass" angegeben. Den Inhalt der „vaulted_vars.yaml" kann man mit folgendem Befehl anzeigen bzw. editieren:

```
ansible-vault view vaulted_vars.yaml
# ODER
ansible-vault edit vaulted_vars.yaml
Vault password: ****
webserver25_become_pass: test123
webserver46_become_pass: test123
```

Die Stärke von Ansible kommt erst beim Einsatz der sogenannten Playbooks oder Rollen (vgl. Lernfeld 12b, Kapitel 5.4.1) zum Tragen, mit deren Hilfe man komplexere Aufgaben erledigen kann. Bei direktem Aufruf über die Kommandozeile werden die einzelnen Parameter nicht nachhaltig gespeichert, sodass keine reproduzierbaren Ergebnisse erzielt werden können.

Im folgenden Beispiel werden die log-Dateien „access.log" und „error.log" des Webservers „Nginx" auf den Ansible-Controller in den Unterordner „fetched" heruntergeladen. Zunächst werden aus dem Inventory alle Zielsysteme ausgewählt (hier: „hosts: all" Zeile 2). Für das Herunterladen werden wieder sudo-Rechte benötigt, die mittels „become: yes" (Zeile 3) angefordert werden.

Die eigentliche Aufgabe bekommt zunächst einen nachvollziehbaren Namen, der während der Ausführung des Playbooks angezeigt wird (hier: „fetch *.log" Zeile 6). Für die Aufgabe wird das fetch-Modul verwendet. Das Modul verlangt eine Quelle (src) auf dem Zielsystem und einen Ablageort (dest) auf dem Ansible-Controller. Um nicht für jede log-Datei einen eigenen Task anlegen zu müssen, wird eine Schleife (loop), die die Items „access" und „error" (Zeile 11–13) enthält, benutzt. Die Werte werden als Variable „{{ item }}" (Zeile 8) in der Quelle verwendet.

```
# Inhalt: playbook-loop.yaml
- hosts: all
  become: yes

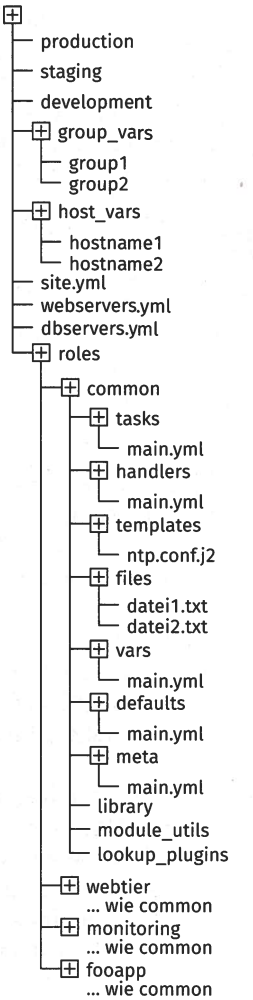
  tasks:
    - name: fetch *.log
      fetch:
        src: /var/log/nginx/{{ item }}.log
        dest: fetched
      register: fetch_output
      loop:
        - access
        - error

# ENDE playbook-loop.yaml

# Aufruf Playbook playbook-loop.yaml
ansible-playbook -i hosts2.yaml playbook-loop.yaml --ask-vault-pass -e@vaulted_vars.yml
```

Um in größeren Umgebungen eine nachvollziehbare Struktur der einzelnen Dokumente pflegen zu können, empfiehlt das Ansible-Projekt, die Dateien in einer vorgegebenen Ordnerstruktur abzulegen und mit sogenannten Rollen (roles) zu arbeiten. Die grundlegenden Ordner der empfohlenen Ordnerstruktur zeigt die folgende Abbildung rechts neben der Tabelle.

Bereich	Bedeutung
production/ staging/ development	Playbooks für die beispielhaften Bereiche
group_vars	Variablen, die innerhalb von Gruppen eingesetzt werden
host_vars	Variablen, die in Verbindung mit Hosts genutzt werden
site.yml/ webservers.yml/ dbservers.yml	Haupt-Playbook (site) und spezielle Playbooks für z. B. Web-/Datenbankserver
roles	Ordner umfasst alle Rollen; hier: common, webtier, monitoring
common	Ordner umfasst exemplarisch alle Ordner einer Rolle
tasks	Ordner enthält die Hauptaufgaben für die Rolle
handlers	Handler können für Aktionen wie das Neustarten eines Servers genutzt werden, nachdem ein Task erfolgreich ausgeführt wurde
templates	Ordner enthält die variablenbasierten Jinja-Vorlagen, die während der Ausführung in statischen Dateien gespeichert werden
files	Ordner enthält statische Dateien, die für die Ausführung der Rolle benötigt werden
vars	Variablen mit höherer Priorität als die Standard-variabeln (defaults); diese Variablen können nur über die Kommandozeile überschrieben werden
defaults	Standardvariabeln für die Rolle mit niedrigster Priorität
meta	Ordner enthält Metainformationen über Abhängigkeiten, die zur Ausführung der Rolle benötigt werden



Im Downloadbereich ist ein Beispiel für eine Rolle „apache2-test“ hinterlegt.

Die Rolle nutzt einen der bekannten Server als Inventory. Diese sind als „inventory.yml“ abgelegt. Als Startpunkt wird das Playbook „playbook.yml“ genutzt. Dieses wird mittels „ansible-playbook -i inventory.yml playbook.yml“ aufgerufen. Im Playbook werden zunächst die Zielsysteme aus der Datei „inventory.yml“ entnommen und die Rolle „apache2-test“ aufgerufen, die im roles-Ordner mit der oben aufgeführten Struktur ablegt ist. Innerhalb der Rolle werden nun automatisch die Tasks in der „main.yml“ im Tasks-Ordner aufgerufen. Dort sind insgesamt fünf Aufgaben hinterlegt.

```
---
- name: Installiere Pakete (1)
  become: true
  apt:
    name: "{{ item }}"
    state: present
  loop: "{{ apps_to_install }}"

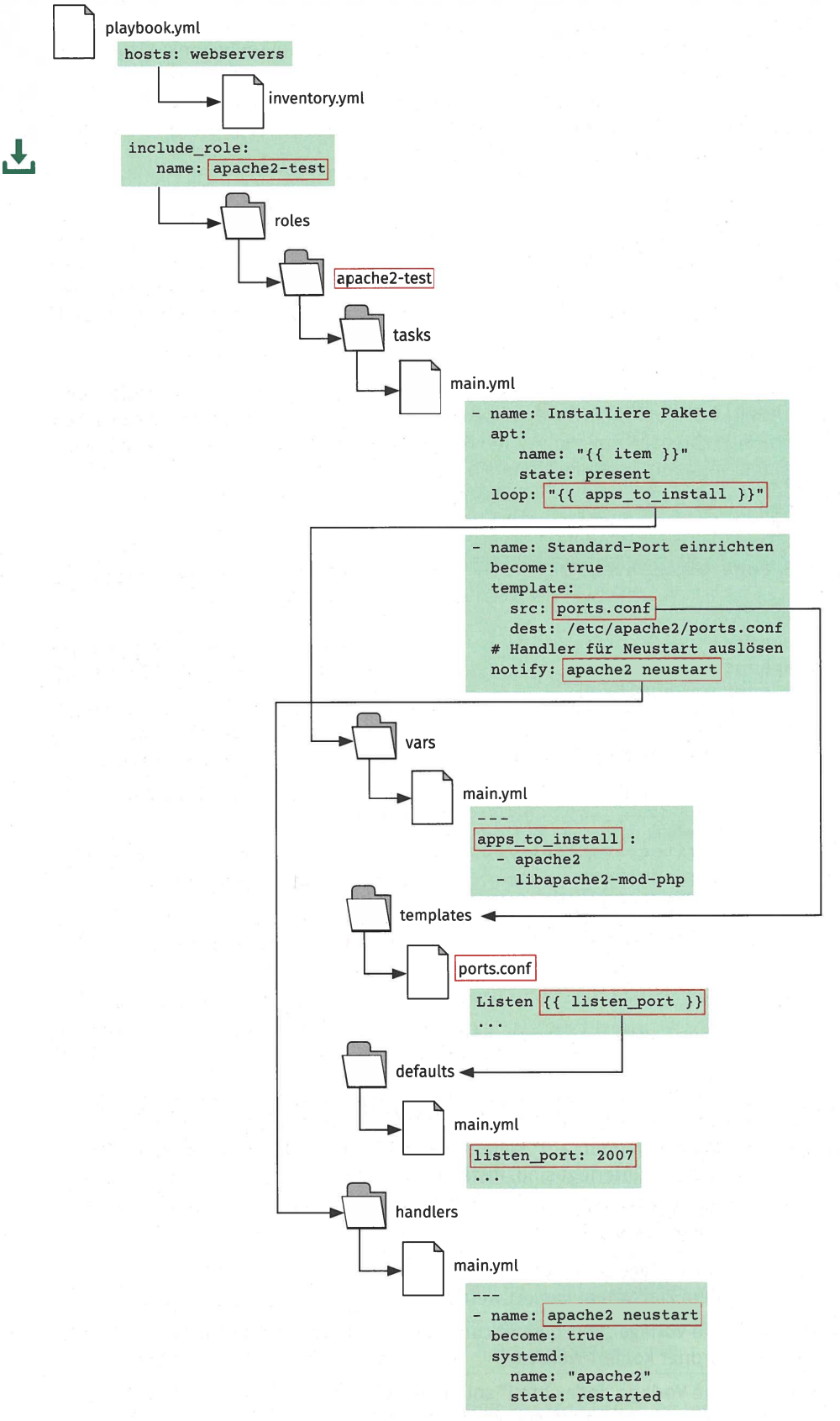
- name: Apache2 enable und starten (2)
  become: true
  systemd:
    name: apache2
    state: started
    enabled: true

- name: Standard-Port einrichten (3)
  become: true
  template:
    src: ports.conf
    dest: /etc/apache2/ports.conf
  # Handler für Neustart auslösen
  notify: apache2 neustart

- name: Standard-Site einrichten (4)
  become: true
  template:
    src: 000-default.conf
    dest: /etc/apache2/sites-available/000-default.conf
  # Handler für Neustart auslösen
  notify: apache2 neustart

- name: index.html einrichten (5)
  become: true
  template:
    src: index.html
    dest: /var/www/{{ html_dir }}/index.html
```

Nr.	Aufgaben	Beschreibung
1	**Installiere Pakete**	Mit dem Modul „apt“ sollen alle Pakete, die in der Variable „apps_to_install“ hinterlegt sind, installiert werden.
2	**Apache2 enable und starten**	Der Apache-Server soll gestartet werden. Durch „enable“ auch nach einem Neustart.
3	**Standard-Port einrichten**	Die Vorlage „ports.conf“ soll in den Apache-Konfigurationsordner auf dem Zielsystem kopiert werden.
4	**Standard-Site einrichten**	Die Vorlage „000-default.conf“ soll in den Apache-Konfigurationsordner kopiert werden.
5	**index.html einrichten**	Die Vorlage „index.html“ soll in den vorgegebenen Ordner auf dem Zielsystem kopiert werden.



Aufrufstruktur innerhalb der Rolle

In der vorliegenden Rolle werden an einigen Stellen Variablen verwendet. In der vorangegangenen Abbildung wird der Ablauf während der Ausführung des Playbooks aufgezeigt. Exemplarisch werden hier die beiden Tasks „Installiere Pakete“ und „Standard-Port einrichten“ genauer betrachtet.

Die Liste der zu installierenden Pakete wird durch die Variable „{{ apps_to_install }}“ vorgegeben. Der Inhalt dieser Variable ist im Ordner „vars“ in „main.yml“ abgelegt. Als Pakete sind „apache2“ und „libapache2-mod-php“ hinterlegt.

Für den Task „Standard-Port einrichten“ wird das Modul „template“ verwendet. In der Vorlage „ports.conf“ wird wiederum eine Variable „{{ listen_port }}“ genutzt. Diese Variable wird aus „main.yml“ im Ordner „defaults“ hinterlegt. Damit die neue Konfiguration für den Port wirksam wird, ist es notwendig, den Server neuzustarten. Dies wird durch einen Handler „apache2 neustart“ realisiert, welcher in „main.yml“ im Ordner „handlers“ definiert ist.

In Verbindung mit einer Versionierung wie „git“ kann auf diese Weise eine nachhaltige Konfigurationsumgebung geschaffen werden. In großen Umgebungen mit vielen Kunden können die kundenspezifischen Daten unabhängig von den eigentlichen Konfigurationsschritten abgelegt werden (siehe auch Lernfeld 10b, Kapitel 1.5.1).

(3) Herstellerspezifische Konfigurationssysteme

Neben den vorgestellten offenen Systemen gibt es im Cloud-Bereich herstellerspezifische Systeme zur Verwaltung der Infrastruktur. Die grundsätzliche Vorgehensweise hier ist nahezu identisch.

Weitere Konfigurationsumgebungen	
System	**Erklärung**
Microsoft Azure	Microsoft bietet eine Reihe von Tools wie Azure Boards, Pipelines, Repos, Test Plans und Artifacts zur Automatisierung des Betriebs an.
VMware	VMware unterstützt ebenfalls die Verwaltung von Cloud-Systemen genauso wie von On-Premise-Umgebungen.
IBM Engineering Lifecycle Management (ELM)	IBM bietet im Rahmen des ELM eine umfangreiche Lösung zur Verwaltung aller Teilaspekte einer Entwicklung bis hin zur Steuerung des Quality und Requirements Management.
Amazon AWS Systems Manager	Der AWS Systems Manager umfasst u. a. Operations-, Applications-, Change- und Node-Management.
Google Cloud Composer	Die Google Cloud bietet zur Automatisierung den sogenannten Cloud Composer mit einer Python-API zur Automatisierung von Aufgaben an.

(4) LUN und SAN

Die Logical Unit Number (LUN) stammt ursprünglich aus dem Bereich der SCSI-Speichersysteme (Small Computer System Interface) und bezeichnet eine eindeutige Kennung für einen Speicherbereich. LUN kommen bei der Verwaltung von Block-Storage-Arrays in Storage Area Networks (SAN) zum Einsatz und werden vom sogenannten Logical Volume Manager (LVM) verwaltet. Eine LUN kann unterschiedliche (hybride) Speichertechnologien zusammenfassen.

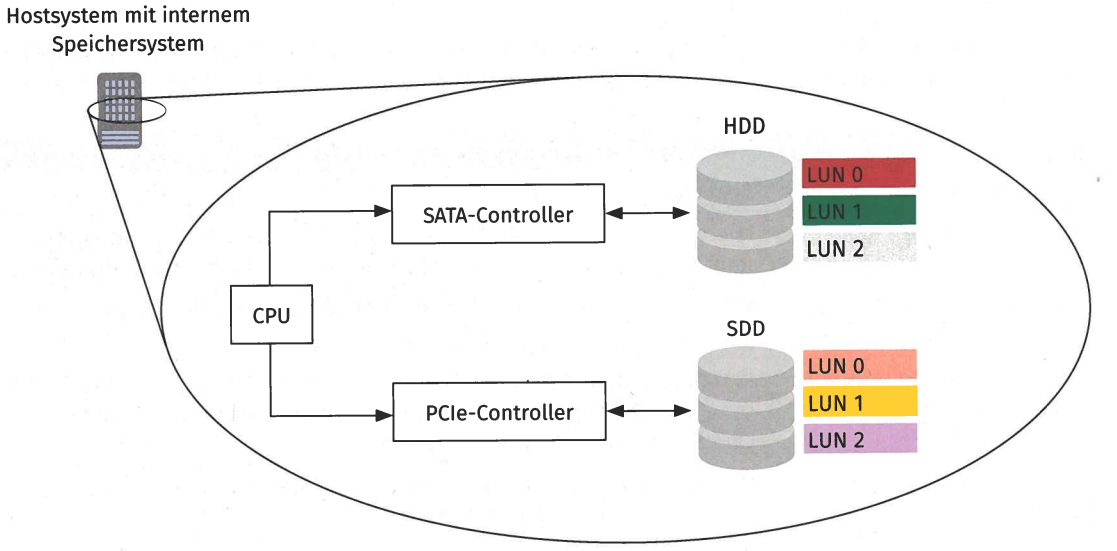
In einem Windows-System kann die Laufwerkskennung „C:“ bzw. „D:“ als LUN verstanden werden. In einem Linux-System ist dies noch flexibler, da jeder Ordner innerhalb des Dateisystems als Einhängepunkt für einen logischen Speicherbereich dienen kann. Die Wurzel des Dateisystems „/“ (sogenannte „root“) kann dabei auf einen anderen logischen Speicherbereich verweisen, als der darunterliegende Benutzerordner „/home“. Diese Bereiche können auf einer physischen Festplatte wie einer HDD in Form von zwei

logischen Partitionen oder getrennt auf einer SSD und einer HDD verteilt liegen. Die Verwaltung dieser LUN übernimmt der LVM in Form der Datenträgerverwaltung von Windows.

Logical Unit Number (LUN)

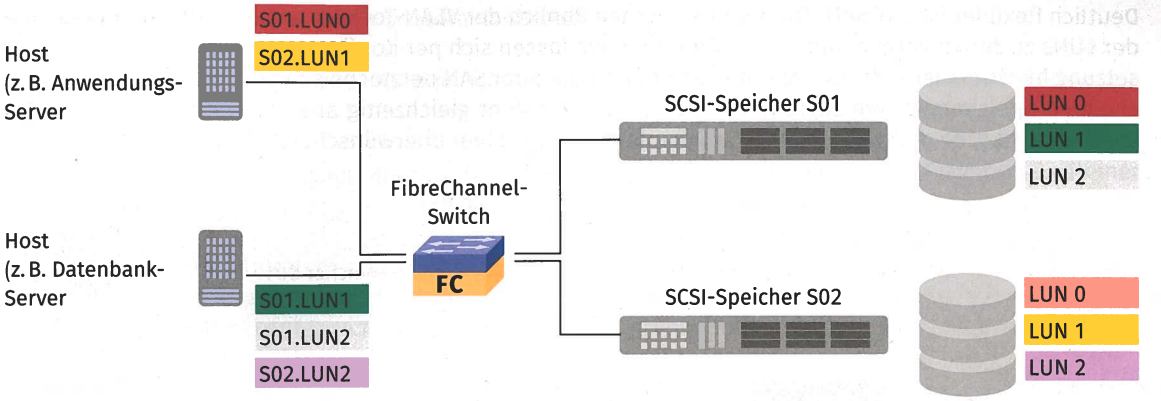
Eine Logical Unit Number stellt die Funktionalitäten eines Speichersystems in allgemeiner Form zur Verfügung. Die direkten physischen Zugriffe, die sich für jede Speichertechnologie unterscheiden, werden dem nutzenden System (meist das Betriebssystem) auf diese Weise abstrahiert zugänglich gemacht.

In einem PC wird eine LUN in der Regel direkt über eine SATA-Schnittstelle (Serial Advanced Technology Attachement) oder bei SSD über eine PCIe (Peripheral Component Interconnect Express) an die CPU angebunden. Die Festplatten sind hierdurch nur genau einem Rechnersystem zugänglich. In einem Rechenzentrum werden die Rechnersysteme (Compute Unit, bestehend aus CPU und RAM) getrennt von den Speichersystemen (SAN) aufgebaut und über Netzelemente miteinander verbunden. Als Schnittstelle zwischen Rechnersystem und Speichersystem kommt meist FibreChannel zum Einsatz, um bestmöglichen Zugriff (Zugriffszeiten und Durchsatz) zu gewährleisten.



Gegenüberstellung innerhalb des PC mit SATA-Anbindung direkt an CPU

Durch das Entkoppeln von Rechner- und Speichersystem ergeben sich diverse Vorteile. So können Server durch Standby-Geräte abgesichert werden. Wird der Hauptserver zu Wartungszwecken heruntergefahren, kann der Standby-Server direkt mit dem Speichersystem des Hauptservers hochgefahren werden. Wäre das Speichermedium direkt im Hauptserver verbaut, müssten hierfür die Festplatten zunächst in den Standby-Server eingebaut werden.

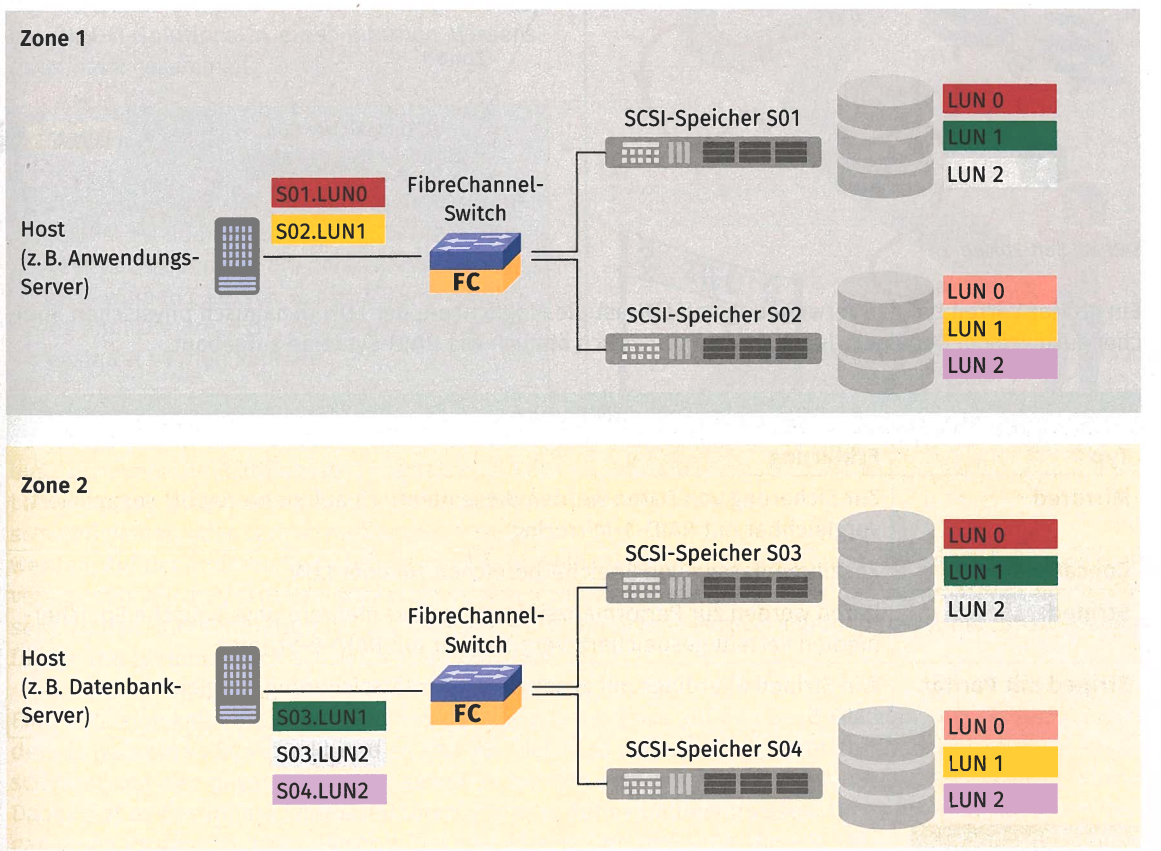


Storage Area Network (SAN) mit FibreChannel-Switch

Zoning

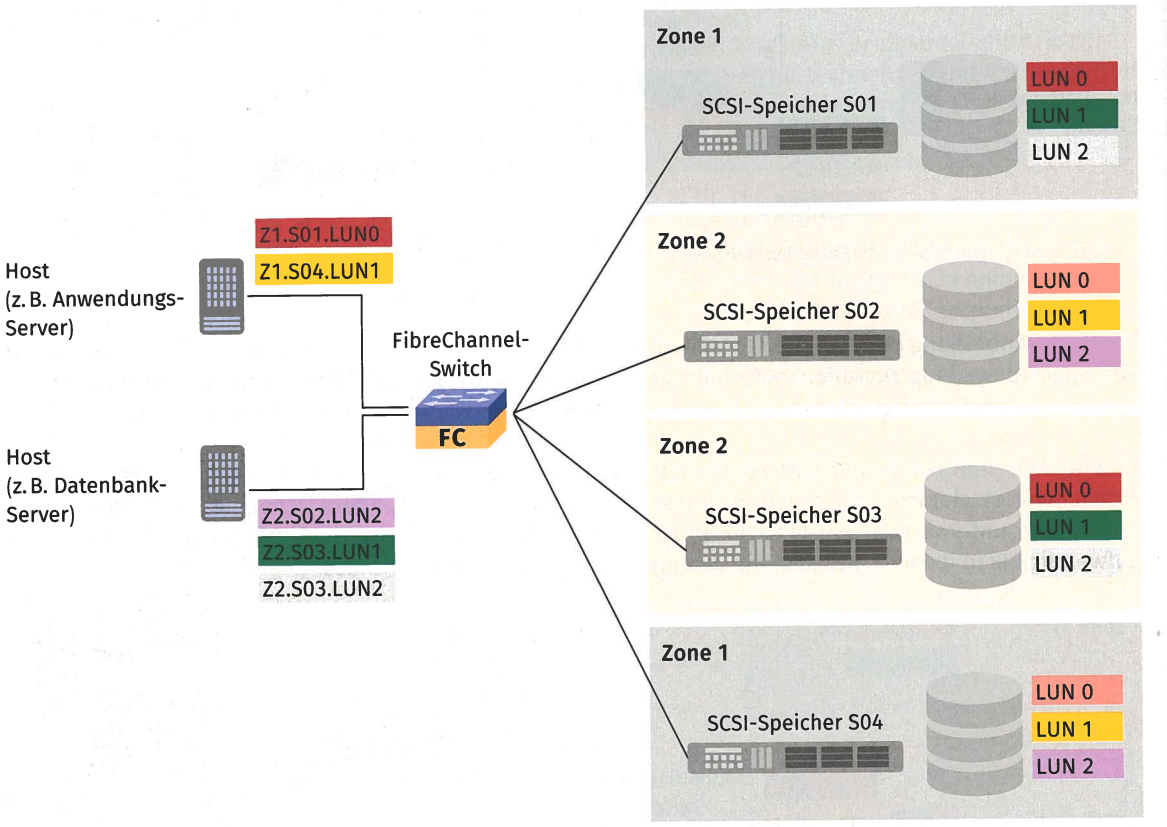
Durch die Anbindung der Speichersysteme über ein Netzwerk können zunächst alle Rechnersysteme auf alle Speichersysteme zugreifen, sofern diese über das Netzwerk erreichbar sind. Dies stellt eine grundsätzliche Sicherheitslücke dar. Durch SAN-Zoning kann man die Zugriffsbereiche einschränken.

Es wird zwischen Hard- und Soft-Zoning unterschieden. Bei **Hard-Zoning** wird die Trennung durch die möglichen Netzelemente durchgeführt. Diese Art des Zonings ist besonders sicher, da ungewollte Zugriffe physisch unterbunden werden. Allerdings ist diese Art des Zonings hinsichtlich Änderungen relativ aufwendig, da in diesem Fall auch die Verkabelung angepasst werden muss.



SAN mit Hard-Zoning

Deutlich flexibler ist das **Soft-Zoning**. Hier werden ähnlich der VLAN-Technik per Software die Zuordnung der LUNs zu Zonen vorgenommen. Die Zuordnungen lassen sich per Konfiguration leicht ändern. Voraussetzung hierfür ist allerdings, dass alle Rechnersysteme per SAN netztechnisch physisch auf die entsprechenden Speichersysteme zugreifen können. Hierin besteht gleichzeitig aber auch ein potenzielles Sicherheitsrisiko, da durch Fehlkonfiguration oder Softwarefehler unerwünschte Zugriffe erfolgen könnten.



SAN mit Soft-Zoning

Ein großer Vorteil bei der Verwendung von LUN ist die Möglichkeit, der LUN dynamisch physischen Speicher zuzuweisen. Dabei werden LUN-Typen physisch ähnlich wie RAID-Systeme aufgebaut.

LUN-Typen	
Typ	**Erklärung**
Mirrored	Zur Sicherung von Daten werden diese identisch auf mehrere LUN gespeichert; vergleichbar mit RAID-1-Mirroring
Concatenated	Zusammenfassen von Speicherbereichen zu einer LUN
Striped	Daten werden zur Performanzsteigerung über mehrere physikalische Speichermedien verteilt gespeichert; vergleichbar mit RAID-0-Striping
Striped mit Parität	Wie Striped allerdings mit zusätzlicher Paritätssicherung; vergleichbar mit RAID-5

Kompetenzcheck

1. Beschreiben Sie in eigenen Worten, was unter dem Begriff „Konfigurationsmanagement" zu verstehen ist.

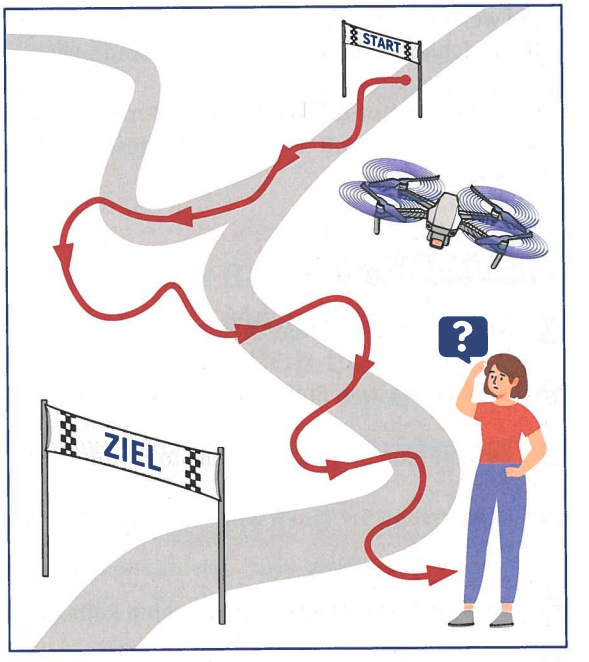
2. Geben Sie zwei Nachteile von Snowflake-Servern an.
3. Geben Sie mindestens zwei Automatisierungssysteme an, die ohne Agenten auf dem Zielsystem auskommen.
4. Beschreiben Sie in eigenen Worten den Unterschied zwischen einem Ansible-Playbook und einer Ansible-Rolle.
5. Erklären Sie den Unterschied zwischen Hard- und Soft-Zoning anhand eines Beispiels.
6. Bearbeiten Sie die Aufgabe 4 der Lernsituation 5 im Arbeitsbuch.

A4

1.6 Reflexion der erzielten Umsetzung

S **Sie sollen Lösungen reflektieren und hinsichtlich der Kundenanforderungen beurteilen.**

Bei jedem Abschluss eines Projektes räumt die JIKU IT-Solutions ihren Mitarbeitern genügend Zeit ein, das Projekt und die dafür durchgeführten Prozesse zu reflektieren. Die Reflexion wird im Projektteam und alleine durchgeführt. Die aus der Reflexion erworbenen Erkenntnisse werden bei JIKU IT-Solutions in einer internen Wissensdatenbank gesammelt.



Der Weg ist das Ziel.

Reflexion

Ziel einer Reflexion im Unternehmenskontext ist es, über einen abgeschlossenen Prozess nachzudenken, um daraus Erkenntnisse zu gewinnen („Lesson learned"), wie zukünftig ähnliche Prozesse besser oder effizienter gestaltet werden können.

Für Projekte soll die Reflexion helfen, zukünftig eine genauere Projektplanung zu erreichen. Für Lernprozesse sollen auf der einen Seite Lernschwierigkeiten oder Lernvermeidungen beseitigt werden. Auf der anderen Seite soll eine kritische Selbstüberprüfung stattfinden, ob das Gelernte wirklich verstanden wurde. Lernschwierigkeiten beruhen in der Regel darauf, dass das Lernmaterial nicht passend zu den Vorkenntnissen ist, um das Lernziel zu erreichen. Lernvermeidung findet bei Inhalten statt, für die der Lernende sich schwer motivieren kann.

Eine Reflexion kann mit einer Sicht aus der Vogelperspektive auf den beschrittenen Weg verglichen werden: Es gab einen Startpunkt und es gab ein Ziel. Nun soll ein Blick, nicht zu hoch und nicht zu eingeschränkt, auf den gegangenen Weg geworfen werden und geschaut werden, ob das Ziel erreicht wurde. Dabei gibt es bestimmte Einflussfaktoren, die eine Reflexion beeinflussen. Da es ein Blick auf einen vergangenen Prozess ist, kann dieser durch eine eigene Grundhaltung, die z. B. sehr pessimistisch oder sehr optimistisch ist, verfälscht werden. Beim Lernprozess kann das neue gelernte Wissen neue Fragen aufwerfen und man kann das Gefühl bekommen, weniger zu wissen als zuvor. Dabei wird aber der Blick auf die aufgeworfenen Fragen gelenkt und der eigentliche Wissenszuwachs ignoriert.

Hilfreich, um diese Beeinflussungen der Reflexion zu vermeiden, ist es, bereits während des Prozesses Notizen zu machen und den Prozess mit diesen Aufzeichnungen zu begleiten. Für die abschließende Reflexion sind diese Notizen dann Informationspunkte, die nicht durch die zuvor beschriebenen Beeinflussungen geprägt sind.

Die Reflexion kann durch Leitfragen unterstützt werden. Grundlegende Leitfragen sind:
- Was war die Ausgangssituation?
- Was war das Ziel?
- Wurde das Ziel erreicht?
- Welche Probleme sind aufgetreten?
- Was ist besonders gut gelungen?
- Was hat Schwierigkeiten bereitet?
- Was hätte im Nachhinein geholfen, die Schwierigkeiten besser zu lösen?

In Bezug auf den Lernprozess geht es weniger darum, welche Lerninhalte besonders Spaß gemacht haben oder welche langweilig zu lernen waren, sondern ob die nötigen Lerninhalte gelernt und auch verstanden wurden. Für den Lernprozess können folgende zusätzliche Fragen bei der Reflexion helfen:
- Welche Verknüpfungen konnte ich zu bisherigem Wissen herstellen?
- Habe ich etwas gelernt, das im völligen Gegensatz zu meinem bisherigen Wissen stand?
- Bei welchen Inhalten hatte ich Schwierigkeiten sie zu verstehen? Was war der Grund dafür?
- Bei welchen Inhalten ist mir das Verständnis leicht gefallen? Was war der Grund dafür?

Die genannten Notizen während des Prozesses sollten mit Blick auf diese Fragen erstellt werden. Daher ist es sinnvoll, sich am Anfang des Prozesses die Fragen aufzuschreiben und weitere passende Fragen zu sammeln.

Kompetenzcheck

1. Nennen Sie zwei Ziele, die durch eine Reflexion erreicht werden sollen.
2. Nennen Sie zwei Einflussfaktoren, die eine Reflexion verfälschen können.
3. Beschreiben Sie, wann der beste Zeitpunkt ist, sich über eine Reflexion Gedanken zu machen.
4. Beschreiben Sie einen Weg, um Verfälschungen einer Reflexion am Ende des Prozesses zu reduzieren.
5. Nennen Sie vier Leitfragen, die eine Reflexion unterstützen können.
6. Bearbeiten Sie den Abschlusstest zum Lernfeld 10b für Systemintegration im Arbeitsbuch.

Test