

7.1 Netzwerk, SSH und Firewall

Linux wird sehr häufig für Netzwerkdienste, Server, Container, Router, Firewalls, VPN-Systeme und Cloud-Systeme eingesetzt.

Deshalb gehört Netzwerkd Diagnose unter Linux zu den wichtigsten Fähigkeiten in der Systemadministration.

In diesem Kapitel geht es um:

- IP-Adressen anzeigen
- Routing prüfen
- DNS prüfen
- Erreichbarkeit testen
- Ports und Verbindungen prüfen
- SSH verstehen
- Firewall-Grundlagen unter Linux
- typische Netzwerkfehler eingrenzen

Merksatz:

Linux ist im Serverbetrieb stark,
weil Netzwerk,
Dienste,
Logs
und Fernadministration gut über die Shell prüfbar sind.

Lernziele

Nach dieser Seite solltest du erklären und anwenden können:

- wie man IP-Adressen unter Linux prüft
- wie man Routing und Gateway prüft
- wie man DNS prüft
- wie man Erreichbarkeit mit ping testet
- wie man Ports mit ss prüft
- was SSH ist
- warum SSH sicherer als Telnet ist

- wie SSH-Schlüssel grundsätzlich funktionieren
- was eine lokale Firewall macht
- wie man typische Netzwerkfehler systematisch eingrenzt

Grundidee

Netzwerkprobleme unter Linux können viele Ursachen haben.

Typische Fehlerquellen:

- falsche IP-Adresse
- falsche Subnetzmaske
- falsches Gateway
- falscher DNS-Server
- Dienst läuft nicht
- Port ist nicht offen
- Firewall blockiert
- Dienst lauscht nur lokal
- falsche Route
- SSH-Zugang falsch konfiguriert
- Rechte oder Schlüsselproblem

Merksatz:

Bei Netzwerkproblemen immer trennen:
IP,
Route,
DNS,
Port,
Dienst,
Firewall
und Anwendung.

Wichtige Netzwerkbefehle

Befehl	Zweck
ip addr	IP-Adressen anzeigen
ip link	Netzwerkschnittstellen anzeigen

Befehl	Zweck
ip route	Routing-Tabelle anzeigen
ping	Erreichbarkeit per ICMP prüfen
tracert	Weg zum Ziel prüfen, falls installiert
tracertpath	Weg zum Ziel prüfen, häufig einfacher verfügbar
ss	Ports und Verbindungen anzeigen
curl	HTTP/HTTPS testen
dig	DNS prüfen
nslookup	DNS prüfen
hostname	Hostname anzeigen
resolvectl	DNS-Status bei systemd-resolved anzeigen
ssh	sichere Fernadministration

Merksatz:

ip,
ping,
ss,
dig
und ssh gehören zu den wichtigsten Linux-Netzbefehlen.

Netzwerkschnittstellen anzeigen

Netzwerkschnittstellen zeigt man mit:

ip link

Beispielausgabe:

1: lo: <LOOPBACK,UP,LOWER_UP>
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP>

Typische Schnittstellen:

Schnittstelle	Bedeutung
lo	Loopback

Schnittstelle	Bedeutung
eth0	Ethernet-Schnittstelle
ens18	Ethernet-Schnittstelle nach moderner Benennung
enp0s3	Ethernet-Schnittstelle nach moderner Benennung
wlan0	WLAN-Schnittstelle
docker0	Docker-Bridge
br0	Bridge
tun0	VPN-Tunnel
wg0	WireGuard-Schnittstelle

Merksatz:

ip link zeigt,
welche Netzwerkinterfaces existieren und ob sie aktiv sind.

Loopback

Die Loopback-Adresse ist:

127.0.0.1

Der Name dazu ist meist:

localhost

Bedeutung:

Kommunikation mit dem eigenen System.

Beispiele:

curl http://127.0.0.1:8080

ping 127.0.0.1

Wichtig:

Wenn ein Dienst nur auf 127.0.0.1 lauscht,
ist er nur lokal auf dem Server erreichbar,
nicht von anderen Systemen.

Merksatz:

127.0.0.1 bedeutet:
dieses System selbst.

IP-Adressen anzeigen

IP-Adressen zeigt man mit:

```
ip addr
```

oder kurz:

```
ip a
```

Beispiel:

```
ip addr show eth0
```

Typische Informationen:

```
Schnittstellenname
```

```
MAC-Adresse
```

```
IPv4-Adresse
```

```
IPv6-Adresse
```

```
Status
```

```
Netzpräfix
```

Beispiel:

```
inet 192.168.1.50/24
```

Bedeutung:

IPv4-Adresse:

192.168.1.50

Präfix:

/24

Merksatz:

ip addr zeigt IP-Adressen und Präfixe.

IPv4-Adresse und Präfix

Beispiel:

```
192.168.10.25/24
```

Bedeutung:

IP-Adresse:

192.168.10.25

Präfix:

/24

Netz:

192.168.10.0/24

Ein Client mit /24 kann direkt mit anderen Geräten im gleichen Netz kommunizieren, zum Beispiel:

```
192.168.10.1
```

```
192.168.10.50
```

```
192.168.10.200
```

Für andere Netze braucht er ein Gateway.

Merksatz:

```
Prefix bestimmt,  
was lokal erreichbar ist.
```

Routing-Tabelle anzeigen

Die Routing-Tabelle zeigt man mit:

```
ip route
```

Beispielausgabe:

```
default via 192.168.1.1 dev eth0  
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.50
```

Bedeutung:

Eintrag	Bedeutung
default via 192.168.1.1	Standardgateway
dev eth0	über Schnittstelle eth0
192.168.1.0/24	lokales Netz
src 192.168.1.50	eigene Quelladresse

Merksatz:

```
ip route zeigt,  
wohin Pakete gesendet werden.
```

Standardgateway

Das Standardgateway ist der Router in andere Netze.

In der Routing-Tabelle steht es meist als:

```
default via ...
```

Beispiel:

```
default via 192.168.1.1 dev eth0
```

Bedeutung:

Alle Ziele,
für die keine spezifischere Route existiert,
werden über 192.168.1.1 gesendet.

Wichtig:

Das Gateway muss im lokalen Subnetz erreichbar sein.

Merksatz:

Gateway führt aus dem lokalen Netz heraus.

Erreichbarkeit mit ping prüfen

ping prüft Erreichbarkeit per ICMP.

Beispiel:

```
ping 192.168.1.1
```

Nur 4 Pakete senden:

```
ping -c 4 192.168.1.1
```

Beispiele:


```
ping -c 4 127.0.0.1
```

```
ping -c 4 192.168.1.1
```

```
ping -c 4 8.8.8.8
```

```
ping -c 4 example.com
```

Wichtig:

ping prüft nicht,
ob ein TCP- oder UDP-Dienst funktioniert.

Merksatz:

ping prüft Host-Erreichbarkeit,
nicht Dienst-Erreichbarkeit.

ping richtig einordnen

Ergebnis	Bedeutung
ping 127.0.0.1 geht	eigener Netzwerkstack funktioniert grundsätzlich
ping eigene IP geht	lokale IP ist aktiv
ping Gateway geht	lokales Netz zum Gateway funktioniert
ping externe IP geht	Routing ins externe Netz funktioniert
ping Domain geht nicht, IP geht	DNS-Problem wahrscheinlich
ping geht nicht	ICMP blockiert oder Ziel nicht erreichbar

Wichtig:

Manche Systeme blockieren ICMP.
Dann kann ein Dienst trotzdem erreichbar sein.

Merksatz:

ping-Ergebnis immer im Zusammenhang bewerten.

DNS prüfen

DNS löst Namen in IP-Adressen auf.

Typische Befehle:

```
nslookup example.com
```

```
dig example.com
```

```
resolvectl status
```

Beispiel:

```
dig ulrich-wiki.com
```

Wichtig:

```
Wenn IP-Adressen erreichbar sind,  
Namen aber nicht,  
ist DNS wahrscheinlich das Problem.
```

Merksatz:

```
IP geht,  
Name geht nicht:  
DNS prüfen.
```

DNS-Server anzeigen

Je nach Distribution und Konfiguration gibt es verschiedene Wege.

Häufig:

```
cat /etc/resolv.conf
```

Bei systemd-resolved:

```
resolvectl status
```

Wichtig:

`/etc/resolv.conf` kann eine echte Datei
oder ein symbolischer Link auf `systemd-resolved` sein.

Merksatz:

DNS-Konfiguration hängt von Distribution und Resolver ab.

`/etc/hosts`

Die Datei:

```
/etc/hosts
```

enthält lokale Namenszuordnungen.

Beispiel:

```
127.0.0.1 localhost  
192.168.1.10 server01.local
```

Wichtig:

Einträge in `/etc/hosts` können DNS-Abfragen übersteuern oder ergänzen.

Typische Nutzung:

lokale Tests

feste interne Namen

kleine Umgebungen

Fehleranalyse

Merksatz:

```
/etc/hosts ist lokale Namensauflösung.
```

Hostname anzeigen

Hostname anzeigen:

```
hostname
```

Mehr Informationen:

```
hostnamectl
```

Hostname setzen, wenn systemd genutzt wird:

```
hostnamectl set-hostname server01
```

Wichtig:

```
Hostname,  
DNS-Name  
und Eintrag in /etc/hosts sollten sinnvoll zusammenpassen.
```

Merksatz:

```
hostname zeigt den Systemnamen.
```

Ports und Verbindungen mit ss prüfen

Offene Ports und Verbindungen zeigt man mit:

```
ss
```

Häufige Variante:

```
ss -tulpen
```

Bedeutung:

Option	Bedeutung
-t	TCP
-u	UDP
-l	listening
-p	Prozess anzeigen
-e	erweiterte Informationen
-n	numerisch, keine Namensauflösung

Beispiel:

```
ss -tulpen
```

Merksatz:

ss zeigt,
welche Dienste auf welchen Ports lauschen.

LISTEN verstehen

Wenn ein Dienst auf einem Port wartet, steht er im Zustand:

```
LISTEN
```

Beispiel:

```
0.0.0.0:22
```

Bedeutung:

Der Dienst lauscht auf TCP 22 auf allen IPv4-Schnittstellen.

Beispiel:

127.0.0.1:8080

Bedeutung:

Der Dienst lauscht nur lokal auf dem System.

Merksatz:

LISTEN heißt:

Dienst wartet auf Verbindungen.

127.0.0.1 und 0.0.0.0

Adresse	Bedeutung bei lauschendem Dienst
127.0.0.1	nur lokal erreichbar
0.0.0.0	auf allen IPv4-Schnittstellen erreichbar
::1	IPv6 localhost
::	auf allen IPv6-Schnittstellen erreichbar

Beispiel:

127.0.0.1:8080

nur lokal.

0.0.0.0:8080

grundsätzlich von außen erreichbar, wenn Firewall, Routing und Netz passen.

Merksatz:

127.0.0.1 ist lokal.

0.0.0.0 ist auf allen IPv4-Interfaces.

HTTP und HTTPS mit curl prüfen

curl kann HTTP- und HTTPS-Verbindungen testen.

Beispiele:

```
curl http://localhost
```

```
curl http://127.0.0.1:8080
```

```
curl https://example.com
```

Nur Header anzeigen:

```
curl -I https://example.com
```

Ausführlichere Verbindungsausgabe:

```
curl -v https://example.com
```

Typische Nutzung:

Webdienst prüfen

Reverse Proxy testen

HTTP-Statuscode sehen

TLS-Probleme erkennen

Merksatz:

curl prüft Webdienste direkt auf Anwendungsebene.

HTTP-Statuscodes kurz

Code	Bedeutung
200	OK
301	dauerhaft weitergeleitet

Code	Bedeutung
302	temporär weitergeleitet
400	fehlerhafte Anfrage
401	nicht authentifiziert
403	verboten
404	nicht gefunden
500	interner Serverfehler
502	Bad Gateway
503	Dienst nicht verfügbar

Merksatz:

ping prüft ICMP.
curl prüft HTTP oder HTTPS.

SSH

SSH steht für:

Secure Shell

SSH dient zur sicheren Fernadministration.

Typischer Port:

TCP 22

Beispiel:

ssh felix@192.168.1.10

Bedeutung:

Verbindung als Benutzer felix zum Server 192.168.1.10 herstellen.

SSH verschlüsselt die Verbindung und ist deshalb sicherer als Telnet.

Merksatz:

SSH = sichere Fernadministration über TCP 22.

SSH und Telnet

Telnet ist unsicher, weil es Daten unverschlüsselt überträgt.

SSH ist sicherer, weil es die Verbindung verschlüsselt.

Protokoll	Sicherheit
Telnet	unverschlüsselt, unsicher
SSH	verschlüsselt, sicherer

Merksatz:

Telnet vermeiden,
SSH nutzen.

SSH-Server und SSH-Client

SSH besteht aus Client und Server.

SSH-Client:

baut Verbindung auf

Beispiel:

ssh felix@server

SSH-Server:

nimmt Verbindungen an

Dienstname häufig:

```
ssh
```

oder:

```
sshd
```

Prüfen:

```
systemctl status ssh
```

oder je nach Distribution:

```
systemctl status sshd
```

Merksatz:

```
ssh ist der Client.  
sshd ist der SSH-Dienst.
```

SSH-Konfiguration

Die wichtige SSH-Server-Konfiguration liegt häufig unter:

```
/etc/ssh/sshd_config
```

Typische Einstellungen:

```
Port
```

```
PermitRootLogin
```

```
PasswordAuthentication
```

```
PubkeyAuthentication
```

```
AllowUsers
```

ListenAddress

Wichtig:

Nach Änderungen SSH-Konfiguration prüfen,
Dienst neu laden oder neu starten
und bestehende Sitzung nicht sofort schließen.

Merksatz:

SSH-Änderungen vorsichtig durchführen,
damit man sich nicht aussperrt.

SSH-Root-Login

Direkter Root-Login per SSH ist sicherheitskritisch.

Risiken:

Angreifer kennen den Benutzernamen root

Brute-Force-Angriffe einfacher

Aktionen weniger personenbezogen nachvollziehbar

Besser:

normaler Benutzer meldet sich an

administrative Befehle gezielt mit sudo

Merksatz:

Kein direkter Root-Login per SSH,
wenn nicht zwingend nötig.

SSH-Passwort und SSH-Schlüssel

SSH kann verschiedene Authentifizierungsarten nutzen.

Methode	Bedeutung
Passwort	Benutzer meldet sich mit Passwort an
Schlüssel	Anmeldung über privates und öffentliches Schlüsselpaar

SSH-Schlüssel bestehen aus:

privatem Schlüssel
öffentlichem Schlüssel

Der öffentliche Schlüssel liegt auf dem Server. Der private Schlüssel bleibt beim Benutzer.

Merksatz:

Öffentlicher Schlüssel auf Server.
Privater Schlüssel bleibt geheim.

SSH-Schlüssel-Grundprinzip

Vereinfachtes Prinzip:

1. Benutzer besitzt privaten Schlüssel.
2. Server kennt passenden öffentlichen Schlüssel.
3. SSH prüft,
ob der private Schlüssel zum öffentlichen Schlüssel passt.
4. Der private Schlüssel wird nicht an den Server übertragen.

Wichtig:

Der private Schlüssel sollte mit einer Passphrase geschützt sein.

Merksatz:

Der private SSH-Schlüssel darf nicht weitergegeben werden.

authorized_keys

Öffentliche SSH-Schlüssel eines Benutzers liegen häufig in:

```
~/.ssh/authorized_keys
```

Beispiel:

```
/home/felix/.ssh/authorized_keys
```

Wichtige Rechte:

```
~/.ssh sollte nicht für andere beschreibbar sein.
```

```
authorized_keys sollte nicht für andere beschreibbar sein.
```

Typische Rechte:

```
chmod 700 ~/.ssh
```

```
chmod 600 ~/.ssh/authorized_keys
```

Merksatz:

SSH-Schlüsselzugriff hängt auch von Dateirechten ab.

SSH-Fehlerbilder

Fehlerbild	Mögliche Ursache
Connection refused	SSH-Dienst läuft nicht oder Port falsch
Connection timed out	Firewall, Routing oder Ziel nicht erreichbar
Permission denied	Benutzer, Passwort oder Schlüssel falsch
Host key verification failed	Host-Key hat sich geändert

Fehlerbild	Mögliche Ursache
No route to host	Routingproblem oder Zielnetz nicht erreichbar
Port 22 nicht erreichbar	Firewall oder Dienstproblem
Login als root nicht möglich	PermitRootLogin deaktiviert

Merksatz:

SSH-Fehler unterscheiden:

Netzwerk,
Port,
Dienst,
Schlüssel
oder Benutzer.

Host-Key

Beim ersten Verbinden speichert SSH den Host-Key des Servers.

Datei auf dem Client:

```
~/.ssh/known_hosts
```

Zweck:

prüfen,
ob man wieder mit demselben Server spricht.

Warnung:

```
Host key verification failed
```

kann bedeuten:

Server wurde neu installiert

IP zeigt auf anderes System

Host-Key wurde geändert

möglicher Man-in-the-Middle-Angriff

Merksatz:

Host-Key schützt vor unbemerktem Serverwechsel.

SCP und SFTP

SSH kann auch für Dateiübertragung genutzt werden.

SCP:

Dateiübertragung über SSH

SFTP:

SSH File Transfer Protocol

Beispiele:

```
scp datei.txt felix@server:/tmp/
```

```
sftp felix@server
```

Wichtig:

SFTP ist nicht dasselbe wie FTPS.

Merksatz:

SFTP nutzt SSH.

FTPS ist FTP mit TLS.

Firewall unter Linux

Eine Firewall filtert Netzwerkverkehr.

Sie kann entscheiden:

- welche Verbindungen erlaubt sind
- welche Ports erreichbar sind
- welche Quellen zugreifen dürfen
- welcher Verkehr blockiert wird

Linux-Firewall-Techniken:

iptables

nftables

firewalld

ufw

Wichtig:

Je nach Distribution wird ein anderes Werkzeug verwendet.

Merksatz:

Linux-Firewall kann mit unterschiedlichen Werkzeugen verwaltet werden.

ufw

ufw steht für:

Uncomplicated Firewall

ufw wird häufig bei Ubuntu verwendet und ist eine vereinfachte Oberfläche für Firewall-Regeln.

Typische Befehle:


```
ufw status
```

```
ufw allow 22/tcp
```

```
ufw allow 443/tcp
```

```
ufw deny 445/tcp
```

```
ufw enable
```

```
ufw disable
```

Wichtig:

Vor dem Aktivieren einer Firewall per SSH sicherstellen,
dass SSH erlaubt ist.

Merksatz:

Firewall per SSH nie aktivieren,
ohne SSH-Regel zu erlauben.

firewalld

firewalld wird häufig bei Fedora, Rocky Linux und ähnlichen Distributionen genutzt.

Typische Konzepte:

Zonen

Dienste

Ports

permanente Regeln

Beispielbefehle:

```
firewall-cmd --state
```

```
firewall-cmd --list-all
```

```
firewall-cmd --add-service=ssh --permanent
```

```
firewall-cmd --reload
```

Merksatz:

```
firewalld arbeitet stark mit Zonen und Diensten.
```

nftables

nftables ist ein moderner Linux-Firewall-Unterbau.

Es ersetzt in vielen Umgebungen ältere iptables-Strukturen.

Prüfen:

```
nft list ruleset
```

Wichtig:

```
nftables ist leistungsfähig,  
aber weniger einsteigerfreundlich als ufw.
```

Merksatz:

```
nftables ist moderner Firewall-Unterbau unter Linux.
```

iptables

iptables ist ein älteres, aber noch häufig anzutreffendes Firewall-Werkzeug.

Prüfen:

```
iptables -L -n -v
```

Wichtig:

Auf modernen Systemen kann iptables intern auf nftables aufsetzen.

Merksatz:

iptables ist klassisch,
nftables ist moderner.

Firewall-Regeln vollständig denken

Eine Firewall-Regel sollte nicht nur aus einem Port bestehen.

Wichtige Angaben:

Quelle

Ziel

Protokoll

Port

Richtung

Aktion

Zweck

Beispiel:

Quelle:
Adminnetz

Ziel:
Linux-Server

Protokoll:

TCP

Port:

22

Aktion:

erlauben

Zweck:

SSH-Administration

Merksatz:

Port allein ist keine vollständige Firewall-Regel.

Lokale Firewall und Netzwerkfirewall

Ein Zugriff kann an mehreren Stellen blockiert werden.

Beispiele:

lokale Firewall auf dem Linux-Server

Firewall auf Router

Firewall zwischen VLANs

Cloud Security Group

Container-Firewall oder Docker-Regeln

Provider-Firewall

Merksatz:

Bei Portproblemen alle Filterstellen prüfen.

Dienst läuft, aber Zugriff geht nicht

Mögliche Ursachen:

Dienst lauscht nur auf 127.0.0.1

Dienst lauscht auf anderem Port

lokale Firewall blockiert

Netzwerkfirewall blockiert

Routingproblem

DNS zeigt auf falsche IP

falsches Protokoll TCP/UDP

Dienst erwartet TLS oder Hostname

Prüfen:

`systemctl status dienst`

`ss -tulpen`

`ip addr`

`ip route`

Firewall-Regeln

Logs

Merksatz:

Dienststatus allein reicht nicht.

Netzwerkdiagnose: sinnvolle Reihenfolge

Bei Linux-Netzwerkproblemen:

1. Schnittstelle aktiv?
2. IP-Adresse korrekt?
3. Route und Gateway korrekt?
4. DNS korrekt?
5. Ziel per IP erreichbar?
6. Ziel per Name erreichbar?
7. Port offen?
8. Dienst läuft?
9. Firewall-Regeln korrekt?
10. Logs prüfen

Merksatz:

Erst IP,
dann Route,
dann DNS,
dann Port,
dann Dienst.

Beispiel: Kein Internet

Fehlerbild:

Server kommt nicht ins Internet.

Prüfen:

```
ip addr
```

```
ip route
```

```
ping -c 4 gateway-ip
```

```
ping -c 4 8.8.8.8
```

```
ping -c 4 example.com
```

Mögliche Einordnung:

Ergebnis	Ursache
keine IP	DHCP oder Interface
Gateway nicht erreichbar	lokales Netz oder Gateway
externe IP nicht erreichbar	Routing, NAT oder Firewall
Domain nicht erreichbar, IP geht	DNS

Merksatz:

Kein Internet ist kein genauer Fehler.
Schrittweise eingrenzen.

Beispiel: Webseite geht nicht

Fehlerbild:

Webseite auf Linux-Server ist nicht erreichbar.

Prüfen:

```
systemctl status nginx
```

```
ss -tulpen
```

```
curl http://127.0.0.1
```

```
curl http://server-ip
```

```
ip addr
```

```
ip route
```

```
Firewall
```

```
journalctl -u nginx
```

Mögliche Ursachen:

Webserver läuft nicht

Port 80 oder 443 nicht offen

Dienst lauscht nur lokal

Firewall blockiert

falsche IP

Reverse Proxy falsch

Zertifikatproblem

Merksatz:

Webseite prüfen:

Dienst,

Port,

Bind-Adresse,

Firewall,

Logs.

Beispiel: SSH geht nicht

Fehlerbild:

SSH-Verbindung schlägt fehl.

Prüfen auf Clientseite:

```
ssh -v benutzer@server
```

Prüfen auf Serverseite:

```
systemctl status ssh
```

```
ss -tulpen
```

```
journalctl -u ssh
```

Firewall-Regeln

Benutzer und Rechte

```
~/.ssh/authorized_keys
```

Mögliche Ursachen:

Dienst läuft nicht

TCP 22 blockiert

falscher Benutzer

Schlüssel fehlt

Rechte auf .ssh falsch

Root-Login deaktiviert

Merksatz:

SSH-Fehler mit -v,
Dienststatus,

Port
und Logs eingrenzen.

Beispiel: DNS geht nicht

Fehlerbild:

IP funktioniert,
Name funktioniert nicht.

Prüfen:

```
cat /etc/resolv.conf

resolvectl status

dig name

nslookup name

ping -c 4 dns-server-ip

Firewall UDP/TCP 53
```

Mögliche Ursachen:

falscher DNS-Server

DNS-Server nicht erreichbar

falscher Suchsuffix

Firewall blockiert DNS

/etc/hosts überschreibt Namen

Merksatz:

IP ja,
Name nein:
DNS.

Typische Fehlerbilder

Fehlerbild	Wahrscheinliches Thema
keine IP-Adresse	DHCP oder Interface
Gateway nicht erreichbar	lokales Netz, VLAN, Gateway
externe IP nicht erreichbar	Routing, NAT oder Firewall
IP erreichbar, Name nicht	DNS
Dienst läuft, Port nicht erreichbar	Firewall, Bind-Adresse, Port
Port lauscht auf 127.0.0.1	nur lokal erreichbar
SSH Connection refused	Dienst läuft nicht oder Port falsch
SSH Connection timed out	Firewall oder Routing
SSH Permission denied	Benutzer, Passwort oder Schlüssel
curl zeigt 404	Ressource oder Pfad nicht gefunden
curl zeigt 502	Proxy oder Backendproblem

Typische Prüfungsfragen

Frage	Kurzantwort
Wofür ist ip addr?	IP-Adressen anzeigen
Wofür ist ip route?	Routing-Tabelle anzeigen
Was ist default via?	Standardgateway
Wofür ist ping?	ICMP-Erreichbarkeit prüfen
Warum ersetzt ping keinen Porttest?	ping prüft nicht TCP/UDP-Dienste
Wofür ist ss?	Ports und Verbindungen anzeigen
Was bedeutet LISTEN?	Dienst wartet auf Verbindungen
Was bedeutet 127.0.0.1?	localhost, nur eigenes System
Was bedeutet 0.0.0.0 bei Diensten?	alle IPv4-Schnittstellen
Was ist SSH?	sichere Fernadministration
Welchen Port nutzt SSH?	TCP 22

Frage	Kurzantwort
Warum ist Telnet unsicher?	unverschlüsselte Übertragung
Was ist /etc/hosts?	lokale Namensauflösung
Was ist ufw?	einfache Firewall-Verwaltung
Was ist firewalld?	Firewall-Verwaltung mit Zonen

Typische Prüfungsfallen

Falle	Richtig denken
ping geht, also Dienst geht	falsch, ping prüft nur ICMP
127.0.0.1 als extern erreichbar verstehen	nur lokal
0.0.0.0 als Zieladresse verwenden	bei LISTEN bedeutet alle Interfaces
DNS und Internet verwechseln	IP kann gehen, DNS trotzdem nicht
SSH-Problem nur als Passwortproblem sehen	Netzwerk, Port, Dienst, Schlüssel prüfen
Firewall nur lokal prüfen	auch Netzwerkfirewall, Cloud, Router prüfen
Portnummer ohne TCP/UDP nennen	unvollständig
SSH-Root-Login erlauben	Sicherheitsrisiko
private SSH-Schlüssel weitergeben	niemals weitergeben
/etc/hosts vergessen	kann Namensauflösung beeinflussen

IHK-sichere Kurzformulierung

Unter Linux werden Netzwerkprobleme systematisch mit Werkzeugen wie `ip addr`, `ip route`, `ping`, `dig`, `nslookup`, `ss`, `curl` und `journalctl` geprüft. `ip addr` zeigt IP-Adressen und Schnittstellen, `ip route` zeigt Routing-Tabelle und Standardgateway. `ping` prüft die ICMP-Erreichbarkeit, ersetzt aber keinen Port- oder Diensttest. `ss` zeigt, welche Dienste auf welchen TCP- oder UDP-Ports lauschen. DNS-Probleme erkennt man häufig daran, dass eine IP-Adresse erreichbar ist, ein Name aber nicht aufgelöst wird. SSH dient der sicheren Fernadministration über TCP 22 und ist Telnet vorzuziehen, weil die Verbindung verschlüsselt ist. Bei Verbindungsproblemen müssen Dienststatus, Port, Bind-Adresse, Firewall, Routing, DNS und Logs gemeinsam geprüft werden.

Merksätze

Netzwerkfehler systematisch eingrenzen.

`ip addr` zeigt IP-Adressen.

ip link zeigt Schnittstellen.

ip route zeigt Routing.

default via ist das Standardgateway.

ping prüft ICMP.

ping ist kein Porttest.

DNS löst Namen auf.

IP ja,

Name nein:

DNS.

ss zeigt Ports und Verbindungen.

LISTEN heißt:

Dienst wartet.

127.0.0.1 heißt:

nur lokal.

0.0.0.0 heißt:

alle IPv4-Interfaces.

curl prüft HTTP und HTTPS.

SSH nutzt TCP 22.

SSH ist verschlüsselt.

Telnet ist unverschlüsselt.

ssh ist der Client.

sshd ist der Dienst.

Privater SSH-Schlüssel bleibt geheim.

authorized_keys enthält öffentliche Schlüssel.

Host-Key schützt vor unbemerktem Serverwechsel.

ufw ist eine einfache Firewall-Oberfläche.

firewalld arbeitet mit Zonen.

nftables ist moderner Firewall-Unterbau.

iptables ist klassisch.

Firewall-Regeln brauchen Quelle,
Ziel,
Protokoll,
Port,
Aktion
und Zweck.

Dienst läuft heißt nicht automatisch:
Dienst ist erreichbar.

Erst IP,
dann Route,
dann DNS,
dann Port,
dann Dienst,
dann Logs.
