3.17 Befehlsübersicht – Netzwerkdiagnose unter Windows, Linux und macOS

Diese Seite dient als schnelle Befehlsreferenz für die systematische Netzwerkdiagnose. Die ausführliche Interpretation, Fehlerbilder und Sicherheitsregeln stehen auf den vorherigen Seiten dieses Kapitels.

---

1. Kennzeichnungen

Kennzeichnung	Bedeutung
[RO]	Rein lesender Befehl
[TEST]	Aktiver Test, der Netzwerkverkehr erzeugt
[PRIV]	Administrator- oder Root-Rechte können erforderlich sein
[FILE]	Ausgabe wird in eine Datei geschrieben
[SENS]	Ausgabe kann sensible Informationen enthalten
[CHANGE]	Befehl verändert eine Konfiguration
[DISRUPT]	Befehl kann Verbindungen oder Dienste beeinträchtigen

Platzhalter:

Platzhalter	Ersetzen durch
INTERFACE	Tatsächlicher Schnittstellenname
HOST	Hostname des Zielsystems
IP	IP-Adresse
PORT	TCP- oder UDP-Port
DNS_SERVER	IP-Adresse des DNS-Servers
CONTAINER	Tatsächlicher Containername
ZONE	Tatsächliche firewalld-Zone
DIENST	Tatsächlicher Dienstname
DATEI	Gewünschter Dateiname beziehungsweise Pfad

Befehle mit Platzhaltern dürfen nicht unverändert ausgeführt werden.

---

2. Empfohlene Kurzreihenfolge

Schritt	Prüfung	Leitfrage
1	Schnittstellenstatus	Ist der Adapter aktiv und verbunden?
2	IP-Konfiguration	Stimmen Adresse, Präfix, Gateway und DNS?
3	Loopback	Funktioniert der lokale TCP/IP-Stack?
4	Eigene Adresse	Ist die lokale Schnittstelle erreichbar?
5	Gateway	Funktioniert das lokale Netz?
6	Entfernte IP	Funktionieren Routing und Rückweg?
7	DNS	Wird der richtige Name aufgelöst?
8	Route	Welcher Pfad und welche Schnittstelle werden verwendet?
9	Port	Ist der benötigte Transportdienst erreichbar?
10	Anwendung	Antwortet das tatsächliche Protokoll?
11	Firewall, NAT, VPN oder Proxy	Wo wird der Datenfluss beeinflusst?
12	Paketmitschnitt	Welches Paketmuster belegt die Ursache?

3. Betriebssystem, Hostname, Zeit und Benutzer

Aufgabe	Windows	Linux	macOS
Betriebssystem	`[RO] Get-ComputerInfo`	`[RO] cat /etc/os-release`	`[RO] sw_vers`
Kernel beziehungsweise Systemversion	`[RO] systeminfo`	`[RO] uname -a`	`[RO] uname -a`
Hostname	`[RO] hostname`	`[RO] hostnamectl`	`[RO] scutil --get ComputerName`
Aktueller Benutzer	`[RO] whoami`	`[RO] id`	`[RO] id`
Lokale Zeit	`[RO] Get-Date -Format o`	`[RO] date --iso-8601=seconds`	`[RO] date "+%Y-%m-%dT%H:%M:%S%z"`
UTC-Zeit	`[RO] (Get-Date).ToUniversalTime().ToString("o")`	`[RO] date -u "+%Y-%m-%dT%H:%M:%SZ"`	`[RO] date -u "+%Y-%m-%dT%H:%M:%SZ"`

Aufgabe	Windows	Linux	macOS
Zeitzone	[RO] Get-TimeZone	[RO] timedatectl status	[RO][PRIV] sudo systemsetup -gettimezone
Zeitstatus	[RO] w32tm /query /status	[RO] timedatectl status	[RO][PRIV] sudo systemsetup -getusingnetworktime

4. Netzwerkadapter und Schnittstellen

Aufgabe	Windows	Linux	macOS
Adapterübersicht	[RO] Get-NetAdapter	[RO] ip -br link	[RO] ifconfig
Versteckte Adapter	[RO] Get-NetAdapter -IncludeHidden	Nicht direkt vergleichbar	[RO] ifconfig -a
Aktive Adapter	[RO] Get-NetAdapter | Where-Object Status -eq "Up"	[RO] ip link show up	[RO] ifconfig -u
MAC-Adressen	[RO] Get-NetAdapter | Format-Table Name,MacAddress	[RO] ip link show	[RO] ifconfig
Linkgeschwindigkeit	[RO] Get-NetAdapter | Format-Table Name,Status,LinkSpeed	[RO][PRIV] sudo ethtool INTERFACE	[RO] ifconfig INTERFACE
Adapterstatistik	[RO] Get-NetAdapterStatistics	[RO] ip -s link	[RO] netstat -ib
Treiberinformationen	[RO][SENS] Get-CimInstance Win32_PnPSignedDriver | Where-Object DeviceClass -eq "NET"	[RO][SENS] lspci -k	[RO][SENS] system_profiler SPNetworkDataType
USB-Netzwerkadapter	Geräte-Manager beziehungsweise PnP-Geräte prüfen	[RO][SENS] lsusb	[RO][SENS] system_profiler SPUSBDataType
Schnittstellenzuordnung	[RO] Get-NetIPInterface	[RO] ip address show	[RO] networksetup -listallhardwareports

Windows kompakt:

```
[RO] Get-NetAdapter |
    Format-Table Name, InterfaceDescription, Status,
                 LinkSpeed, MacAddress
```

Linux kompakt:

```
[RO] ip -br link
[RO] ip -br address
```

macOS – Hardwareport und Gerätename zuordnen:

```
[RO] networksetup -listallhardwareports
```

---

5. IPv4- und IPv6-Konfiguration

Aufgabe	Windows	Linux	macOS
Gesamte IP-Konfiguration	`[RO][SENS] Get-NetIPConfiguration -All`	`[RO][SENS] ip address show`	`[RO][SENS] ifconfig`
Klassische Ausgabe	`[RO][SENS] ipconfig /all`	Nicht zutreffend	Nicht zutreffend
Kompakte Adressen	`[RO][SENS] Get-NetIPAddress`	`[RO][SENS] ip -br address`	`[RO][SENS] ifconfig`
Nur IPv4	`[RO] Get-NetIPAddress -AddressFamily IPv4`	`[RO] ip -4 address show`	`[RO] ifconfig`
Nur IPv6	`[RO] Get-NetIPAddress -AddressFamily IPv6`	`[RO] ip -6 address show`	`[RO] ifconfig`
Schnittstellen-MTU	`[RO] Get-NetIPInterface`	`[RO] ip link show`	`[RO] ifconfig`
Netzwerkdienstinformationen	Nicht direkt vergleichbar	NetworkManager: `[RO] nmcli device show`	`[RO][SENS] networksetup -getinfo "Wi-Fi"`

Windows nach Schnittstelle sortiert:

```
[RO][SENS] Get-NetIPAddress |
    Sort-Object InterfaceAlias, AddressFamily |
    Format-Table InterfaceAlias, AddressFamily,
                 IPAddress, PrefixLength, AddressState
```

---

6. DHCP prüfen

Aufgabe	Windows	Linux	macOS
DHCP-Konfiguration	`[RO][SENS] ipconfig /all`	`[RO][SENS] nmcli device show`	`[RO][SENS] ipconfig getpacket INTERFACE`

Aufgabe	Windows	Linux	macOS
IPv4-Adresse	`[RO] Get-NetIPAddress -AddressFamily IPv4`	`[RO] ip -4 address show`	`[RO] ipconfig getifaddr INTERFACE`
DHCP-Paketmitschnitt	Dumpcap: `udp port 67 or udp port 68`	tcpdump: `udp port 67 or udp port 68`	tcpdump: `udp port 67 or udp port 68`
DHCPv6-Mitschnitt	Dumpcap: `udp port 546 or udp port 547`	tcpdump: `udp port 546 or udp port 547`	tcpdump: `udp port 546 or udp port 547`

Windows – Lease erneuern:

```
[CHANGE][DISRUPT] ipconfig /renew
```

Windows – Lease freigeben und neu beziehen:

```
[CHANGE][DISRUPT] ipconfig /release
[CHANGE][DISRUPT] ipconfig /renew
```

macOS – DHCP für einen Netzwerkdienst erneuern:

```
[CHANGE][DISRUPT][PRIV] sudo ipconfig set INTERFACE DHCP
```

Unter Linux hängt das sichere Erneuern vom verwendeten Netzwerkmanager ab. Zuerst prüfen:

```
[RO] nmcli device status
[RO] systemctl is-active NetworkManager
[RO] systemctl is-active systemd-networkd
```

DHCP-Erneuerungen können bestehende Verbindungen unterbrechen und dürfen nicht unkontrolliert auf Remote-Systemen ausgeführt werden.

---

7. ARP und IPv6 Neighbor Discovery

Aufgabe	Windows	Linux	macOS
IPv4-Nachbarn	`[RO][SENS] Get-NetNeighbor -AddressFamily IPv4`	`[RO][SENS] ip -4 neigh show`	`[RO][SENS] arp -an`
IPv6-Nachbarn	`[RO][SENS] Get-NetNeighbor -AddressFamily IPv6`	`[RO][SENS] ip -6 neigh show`	`[RO][SENS] ndp -an`

Aufgabe	Windows	Linux	macOS
Klassische ARP-Tabelle	[RO][SENS] arp -a	[RO][SENS] ip neigh show	[RO][SENS] arp -a
Bestimmte Adresse	[RO] Get-NetNeighbor -IPAddress IP	[RO] ip neigh show IP	[RO] arp -n IP
ARP-Mitschnitt	Capture Filter arp	[TEST][PRIV][SENS] sudo tcpdump -i INTERFACE -nn arp	[TEST][PRIV][SENS] sudo tcpdump -i INTERFACE -nn arp
NDP-Mitschnitt	Capture Filter icmp6	[TEST][PRIV][SENS] sudo tcpdump -i INTERFACE -nn icmp6	[TEST][PRIV][SENS] sudo tcpdump -i INTERFACE -nn icmp6

Nachbartabellen dürfen nicht unkontrolliert geleert werden. Dadurch können bestehende Verbindungen kurzfristig beeinflusst werden.

---

8. VLAN und Layer 2

Aufgabe	Windows	Linux	macOS
Adapter und MAC	[RO] Get-NetAdapter	[RO] ip link show	[RO] ifconfig
VLAN-Schnittstellen	Abhängig vom Adaptertreiber und Hyper-V	[RO] ip -d link show type vlan	[RO] networksetup -listVLANs
Bridge	[RO] Get-NetAdapterBinding beziehungsweise Hyper-V prüfen	[RO] bridge link show	[RO] ifconfig bridge0
Linkstatistik	[RO] Get-NetAdapterStatistics	[RO] ip -s link	[RO] netstat -ib
Ethernetdetails	Adaptereigenschaften und Switch prüfen	[RO][PRIV] sudo ethtool INTERFACE	[RO] ifconfig INTERFACE
LLDP-Nachbarn	Abhängig von installiertem Werkzeug	Falls installiert: [RO] lldpcli show neighbors	Abhängig von installiertem Werkzeug

VLAN-Zuordnung wird häufig auf Switch, Access Point, Hypervisor oder Controller festgelegt und ist am Client nicht vollständig sichtbar.

---

9. Routing und Standardgateway

Aufgabe	Windows	Linux	macOS
Routingtabelle	[RO] Get-NetRoute	[RO] ip route show	[RO] netstat -rn
IPv4-Routen	[RO] route print -4	[RO] ip -4 route show	[RO] netstat -rn -f inet

Aufgabe	Windows	Linux	macOS
IPv6-Routen	`[RO] route print -6`	`[RO] ip -6 route show`	`[RO] netstat -rn -f inet6`
IPv4-Standardroute	`[RO] Get-NetRoute -DestinationPrefix "0.0.0.0/0"`	`[RO] ip route show default`	`[RO] route -n get default`
IPv6-Standardroute	`[RO] Get-NetRoute -DestinationPrefix "::/0"`	`[RO] ip -6 route show default`	`[RO] route -n get -inet6 default`
Route zu einem Ziel	`[RO] Find-NetRoute -RemoteIPAddress IP`	`[RO] ip route get IP`	`[RO] route -n get IP`
Policy Routing	Abhängig von Windows-Richtlinien	`[RO] ip rule show`	`[RO] netstat -rn`
Alle Linux-Tabellen	Nicht zutreffend	`[RO][SENS] ip route show table all`	Nicht zutreffend

Windows übersichtlich:

```
[RO] Get-NetRoute |
    Sort-Object AddressFamily, DestinationPrefix, RouteMetric |
    Format-Table AddressFamily, DestinationPrefix,
                 NextHop, InterfaceAlias, RouteMetric
```

---

10. Ping und grundlegende Erreichbarkeit

Aufgabe	Windows	Linux	macOS
IPv4-Ping	`[TEST] ping -4 -n 4 IP`	`[TEST] ping -4 -c 4 IP`	`[TEST] ping -c 4 IPV4`
IPv6-Ping	`[TEST] ping -6 -n 4 IP`	`[TEST] ping -6 -c 4 IP`	`[TEST] ping6 -c 4 IPV6`
20 Messungen	`[TEST] ping -n 20 IP`	`[TEST] ping -c 20 IP`	`[TEST] ping -c 20 IP`
Fortlaufender Ping	`[TEST] ping -t IP`	`[TEST] ping IP`	`[TEST] ping IP`
PowerShell-Test	`[TEST] Test-Connection IP -Count 4`	Nicht zutreffend	Nicht zutreffend
Größere Nutzlast	`[TEST] ping -n 4 -l 1400 IP`	`[TEST] ping -c 4 -s 1400 IP`	`[TEST] ping -c 4 -s 1400 IP`
Don't Fragment	`[TEST] ping -4 -f -l 1400 IP`	`[TEST] ping -4 -M do -s 1400 IP`	`[TEST] ping -D -s 1400 IP`

Empfohlene Reihenfolge:

```
Loopback
→ eigene Adresse
```

→ Standardgateway

→ internes Ziel

→ externes Ziel

→ Hostname

11. Netzwerkpfad untersuchen

Aufgabe	Windows	Linux	macOS
Pfad mit DNS	`[TEST] tracert HOST`	`[TEST] traceroute HOST`	`[TEST] traceroute HOST`
Pfad ohne DNS	`[TEST] tracert -d IP`	`[TEST] traceroute -n IP`	`[TEST] traceroute -n IP`
IPv4-Pfad	`[TEST] tracert -4 -d HOST`	`[TEST] traceroute -4 -n HOST`	`[TEST] traceroute -4 -n HOST`
IPv6-Pfad	`[TEST] tracert -6 -d HOST`	`[TEST] traceroute -6 -n HOST`	`[TEST] traceroute -6 -n HOST`
Pfad und Verlust	`[TEST] pathping /n IP`	Falls installiert: `[TEST] mtr -n -r -c 100 IP`	Falls installiert: `[TEST] mtr -n -r -c 100 IP`
Path-MTU-Hinweis	Nicht direkt vergleichbar	`[TEST] tracepath -n IP`	MTU mit Ping und Route prüfen

Zwischenhops können ICMP-Antworten begrenzen. Verlust an einem einzelnen Zwischenhop ist kein Beweis für weitergeleiteten Paketverlust, wenn nachfolgende Hops und das Ziel keinen entsprechenden Verlust zeigen.

12. DNS-Konfiguration prüfen

Aufgabe	Windows	Linux	macOS
DNS-Server	`[RO][SENS] Get-DnsClientServerAddress`	`[RO][SENS] resolvectl status`	`[RO][SENS] scutil --dns`
DNS-Suffixe	`[RO] Get-DnsClient`	`[RO] resolvectl status`	`[RO][SENS] scutil --dns`
Klassische Konfiguration	`[RO][SENS] ipconfig /all`	`[RO][SENS] cat /etc/resolv.conf`	`[RO][SENS] scutil --dns`
Hosts-Datei	`[RO][SENS] Get-Content "$env:SystemRoot\System32\drivers\etc\hosts"`	`[RO][SENS] cat /etc/hosts`	`[RO][SENS] cat /etc/hosts`
DNS-Cache	`[RO][SENS] Get-DnsClientCache`	Resolverabhängig	`[RO][SENS] dscacheutil -cachedump -entries Host` – Ausgabe abhängig von macOS-Version

`/etc/resolv.conf` kann unter Linux automatisch erzeugt sein und bildet bei lokalen Stub-Resolvern nicht zwingend die vollständige Resolverlogik ab.

---

13. DNS-Abfragen durchführen

Aufgabe	Windows	Linux	macOS
Standardabfrage	`[TEST] Resolve-DnsName HOST`	`[TEST] dig HOST`	`[TEST] dig HOST`
A-Record	`[TEST] Resolve-DnsName HOST -Type A`	`[TEST] dig HOST A`	`[TEST] dig HOST A`
AAAA-Record	`[TEST] Resolve-DnsName HOST -Type AAAA`	`[TEST] dig HOST AAAA`	`[TEST] dig HOST AAAA`
PTR-Record	`[TEST] Resolve-DnsName IP -Type PTR`	`[TEST] dig -x IP`	`[TEST] dig -x IP`
Bestimmter DNS-Server	`[TEST] Resolve-DnsName HOST -Server DNS_SERVER`	`[TEST] dig @DNS_SERVER HOST`	`[TEST] dig @DNS_SERVER HOST`
DNS über TCP	`[TEST] Resolve-DnsName HOST -Server DNS_SERVER -TcpOnly`	`[TEST] dig +tcp @DNS_SERVER HOST`	`[TEST] dig +tcp @DNS_SERVER HOST`
Kurze Ausgabe	`[TEST] (Resolve-DnsName HOST -Type A).IPAddress`	`[TEST] dig +short HOST`	`[TEST] dig +short HOST`
Vollständige Ablaufverfolgung	Nicht direkt vergleichbar	`[TEST] dig +trace HOST`	`[TEST] dig +trace HOST`
Klassischer Test	`[TEST] nslookup HOST DNS_SERVER`	`[TEST] nslookup HOST DNS_SERVER`	`[TEST] nslookup HOST DNS_SERVER`

`dig +trace` fragt mehrere DNS-Server ab und darf nur verwendet werden, wenn direkte externe DNS-Abfragen erlaubt sind.

---

14. DNS-Cache kontrolliert leeren

Diese Befehle verändern den lokalen Cache und können die Reproduzierbarkeit beeinflussen.

Betriebssystem	Befehl
Windows	`[CHANGE] ipconfig /flushdns`
Linux mit systemd-resolved	`[CHANGE][PRIV] sudo resolvectl flush-caches`
macOS	`[CHANGE][PRIV] sudo dscacheutil -flushcache`

Unter Linux hängt der Befehl vom tatsächlich verwendeten Resolver ab. Vorher prüfen:

```
[RO] systemctl is-active systemd-resolved
```

Das Leeren eines DNS-Caches behebt keine fehlerhafte DNS-Zone, falsche Delegierung oder falsche Serverkonfiguration.

---

15. TCP-Listener, UDP-Endpunkte und Verbindungen

Aufgabe	Windows	Linux	macOS
TCP-Verbindungen	`[RO] Get-NetTCPConnection`	`[RO] ss -tan`	`[RO] netstat -anv -p tcp`
TCP-Listener	`[RO] Get-NetTCPConnection -State Listen`	`[RO] ss -ltn`	`[RO] lsof -nP -iTCP -sTCP:LISTEN`
UDP-Endpunkte	`[RO] Get-NetUDPEndpoint`	`[RO] ss -lun`	`[RO] lsof -nP -iUDP`
Listener mit Prozess	Prozess-ID über `OwningProcess`	`[RO][PRIV] sudo ss -ltnp`	`[RO][PRIV] sudo lsof -nP -iTCP -sTCP:LISTEN`
Bestehende TCP-Verbindungen	`[RO] Get-NetTCPConnection -State Established`	`[RO] ss -tn state established`	`[RO] netstat -anv -p tcp`
Klassische Übersicht	`[RO] netstat -ano`	`[RO] ss -tuln`	`[RO] netstat -anv`
TCP-Statistik	`[RO] netstat -s -p tcp`	`[RO] nstat` beziehungsweise `[RO] netstat -s`	`[RO] netstat -s -p tcp`

Bestimmten TCP-Port prüfen:

Windows:

```
[RO] Get-NetTCPConnection -State Listen -LocalPort PORT `
    -ErrorAction SilentlyContinue
```

Linux:

```
[RO] ss -ltn 'sport = :PORT'
```

macOS:

```
[RO] lsof -nP -iTCP:PORT -sTCP:LISTEN
```

---

16. Prozess zu einem Port ermitteln

Windows:

```
[RO] Get-NetTCPConnection -State Listen -LocalPort PORT |
    Select-Object LocalAddress, LocalPort, State, OwningProcess
```

Anschließend:

```
[RO] Get-Process -Id PID
```

Alternativ:

```
[RO] netstat -ano
[RO] tasklist /FI "PID eq PID"
```

Linux:

```
[RO][PRIV] sudo ss -ltnp 'sport = :PORT'
```

```
[RO][PRIV] sudo lsof -nP -iTCP:PORT -sTCP:LISTEN
```

macOS:

```
[RO][PRIV] sudo lsof -nP -iTCP:PORT -sTCP:LISTEN
```

```
[RO] ps -p PID -o pid,ppid,user,command
```

---

17. TCP- und UDP-Porttests

Aufgabe	Windows	Linux	macOS
TCP-Port	`[TEST] Test-NetConnection HOST -Port PORT`	`[TEST] nc -vz -w 5 HOST PORT`	`[TEST] nc -vz -w 5 HOST PORT`

Aufgabe	Windows	Linux	macOS
Detaillierter TCP-Test	`[TEST] Test-NetConnection HOST -Port PORT -InformationLevel Detailed`	`[TEST] nc -vz -w 5 HOST PORT`	`[TEST] nc -vz -w 5 HOST PORT`
Nmap TCP Connect	`[TEST] nmap -sT -p PORT HOST`	`[TEST] nmap -sT -p PORT HOST`	`[TEST] nmap -sT -p PORT HOST`
Nmap UDP	Administrator-Konsole: `[TEST][PRIV] nmap -sU -p PORT HOST`	`[TEST][PRIV] sudo nmap -sU -p PORT HOST`	`[TEST][PRIV] sudo nmap -sU -p PORT HOST`
Netcat UDP	Falls Netcat installiert: `[TEST] nc -vzu HOST PORT`	`[TEST] nc -vzu -w 3 HOST PORT`	`[TEST] nc -vzu -w 3 HOST PORT`

UDP-Tests sind ohne gültige Anfrage des Anwendungsprotokolls häufig nicht eindeutig. `Test-NetConnection -Port` ist ein TCP-Test.

Nmap und Portscans dürfen nur gegen ausdrücklich freigegebene Systeme verwendet werden.

---

18. HTTP, HTTPS und TLS prüfen

Aufgabe	Windows	Linux	macOS
HTTP-Header	`[TEST] curl.exe -I http://HOST/`	`[TEST] curl -I http://HOST/`	`[TEST] curl -I http://HOST/`
HTTPS-Header	`[TEST] curl.exe -I https://HOST/`	`[TEST] curl -I https://HOST/`	`[TEST] curl -I https://HOST/`
Ausführlicher Test	`[TEST][SENS] curl.exe -v https://HOST/`	`[TEST][SENS] curl -v https://HOST/`	`[TEST][SENS] curl -v https://HOST/`
Nur IPv4	`[TEST] curl.exe -4 -I https://HOST/`	`[TEST] curl -4 -I https://HOST/`	`[TEST] curl -4 -I https://HOST/`
Nur IPv6	`[TEST] curl.exe -6 -I https://HOST/`	`[TEST] curl -6 -I https://HOST/`	`[TEST] curl -6 -I https://HOST/`
TLS-Handshake	Falls OpenSSL installiert: `[TEST][SENS] openssl s_client -connect HOST:443 -servername HOST`	`[TEST][SENS] openssl s_client -connect HOST:443 -servername HOST`	`[TEST][SENS] openssl s_client -connect HOST:443 -servername HOST`

HTTP-Zeitanteile:

```
[TEST][SENS] curl -sS -o /dev/null \
  -w 'DNS: %{time_namelookup}\nConnect: %{time_connect}\nTLS: %{time_appconnect}\nStartTransfer: %{time_starttransfer}\nTotal: %{time_total}\nHTTP: %{http_code}\n' \
  https://HOST/
```

`curl -k` deaktiviert die Zertifikatsprüfung und darf nicht als dauerhafte Lösung verwendet werden.

---

19. Windows-Firewall prüfen

Aufgabe	Befehl
Aktives Netzwerkprofil	[RO] Get-NetConnectionProfile
Firewallprofile	[RO] Get-NetFirewallProfile
Ausführliche Profile	[RO] Get-NetFirewallProfile | Format-List *
Aktivierte Regeln	[RO][SENS] Get-NetFirewallRule -Enabled True
Blockierungsregeln	[RO][SENS] Get-NetFirewallRule -Enabled True -Action Block
Eingehende Erlaubnisregeln	[RO][SENS] Get-NetFirewallRule -Enabled True -Direction Inbound -Action Allow
Portfilter	[RO][SENS] Get-NetFirewallPortFilter
Adressfilter einer Regel	[RO][SENS] Get-NetFirewallRule -DisplayName "REGEL" | Get-NetFirewallAddressFilter
Programmfilter	[RO][SENS] Get-NetFirewallRule -DisplayName "REGEL" | Get-NetFirewallApplicationFilter
Aktiver Richtlinienspeicher	[RO][SENS] Get-NetFirewallRule -PolicyStore ActiveStore
Klassische Übersicht	[RO] netsh advfirewall show allprofiles

Regeln für TCP-Port 443 suchen:

```
[RO][SENS] Get-NetFirewallPortFilter -Protocol TCP |
    Where-Object LocalPort -eq "443" |
    Get-NetFirewallRule |
    Format-Table DisplayName, Enabled, Direction, Action, Profile
```

---

20. Linux-Firewall prüfen

System	Aufgabe	Befehl
firewalld	Status	[RO] firewall-cmd --state
firewalld	Aktive Zonen	[RO] firewall-cmd --get-active-zones

System	Aufgabe	Befehl
firewalld	Zone einer Schnittstelle	`[RO] firewall-cmd --get-zone-of-interface=INTERFACE`
firewalld	Zonenkonfiguration	`[RO][SENS] firewall-cmd --zone=ZONE --list-all`
firewalld	TCP-Port prüfen	`[RO] firewall-cmd --zone=ZONE --query-port=443/tcp`
firewalld	Dienste	`[RO] firewall-cmd --zone=ZONE --list-services`
firewalld	Permanente Konfiguration	`[RO][SENS] firewall-cmd --permanent --zone=ZONE --list-all`
nftables	Regelwerk	`[RO][PRIV][SENS] sudo nft list ruleset`
nftables	Regeln mit Handles	`[RO][PRIV][SENS] sudo nft -a list ruleset`
iptables	IPv4-Regeln	`[RO][PRIV][SENS] sudo iptables -L -n -v --line-numbers`
iptables	IPv6-Regeln	`[RO][PRIV][SENS] sudo ip6tables -L -n -v --line-numbers`
ufw	Status	`[RO][PRIV][SENS] sudo ufw status verbose`
ufw	Nummerierte Regeln	`[RO][PRIV][SENS] sudo ufw status numbered`

Runtime- und permanente firewalld-Konfiguration müssen getrennt verglichen werden.

---

21. macOS-Firewall und pf prüfen

Aufgabe	Befehl
Anwendungsfirewallstatus	`[RO] /usr/libexec/ApplicationFirewall/socketfilterfw --getglobalstate`
Alle eingehenden Verbindungen blockieren	`[RO] /usr/libexec/ApplicationFirewall/socketfilterfw --getblockall`
Stealth-Modus	`[RO] /usr/libexec/ApplicationFirewall/socketfilterfw --getstealthmode`
Konfigurierte Anwendungen	`[RO][PRIV][SENS] sudo /usr/libexec/ApplicationFirewall/socketfilterfw --listapps`
pf-Status	`[RO][PRIV] sudo pfctl -s info`
pf-Regeln	`[RO][PRIV][SENS] sudo pfctl -s rules`
pf-NAT	`[RO][PRIV][SENS] sudo pfctl -s nat`
pf-Zustände	`[RO][PRIV][SENS] sudo pfctl -s states`
Regeln mit Zählern	`[RO][PRIV][SENS] sudo pfctl -vvs rules`

Die macOS-Anwendungsfirewall und `pf` sind getrennte Filterebenen.

---

22. NAT und Portweiterleitung

Aufgabe	Windows	Linux	macOS
Windows-NAT	`[RO][PRIV][SENS] Get-NetNat`	Nicht zutreffend	Nicht zutreffend
Statische Zuordnungen	`[RO][PRIV][SENS] Get-NetNatStaticMapping`	Nicht zutreffend	Nicht zutreffend
NAT-Sitzungen	`[RO][PRIV][SENS] Get-NetNatSession`	Conntrack verwenden	pf-Zustände verwenden
nftables-NAT	Nicht zutreffend	`[RO][PRIV][SENS] sudo nft -a list ruleset`	Nicht zutreffend
iptables-NAT	Nicht zutreffend	`[RO][PRIV][SENS] sudo iptables -t nat -L -n -v --line-numbers`	Nicht zutreffend
pf-NAT	Nicht zutreffend	Nicht standardmäßig allgemein vorhanden	`[RO][PRIV][SENS] sudo pfctl -vvs nat`
IPv4-Forwarding	Rollenabhängig	`[RO] sysctl net.ipv4.ip_forward`	`[RO] sysctl net.inet.ip.forwarding`
Conntrack	Windows-NAT-Sitzungen	`[RO][PRIV][SENS] sudo conntrack -L`	`[RO][PRIV][SENS] sudo pfctl -s states`

Öffentlich sichtbare IPv4-Adresse:

Windows:

```
[TEST][SENS] Invoke-RestMethod -Uri "https://api.ipify.org"
```

Linux und macOS:

```
[TEST][SENS] curl -4 https://api.ipify.org
```

Dabei wird die öffentliche Quelladresse an einen externen Dienst übermittelt.

---

23. Proxy prüfen

Aufgabe	Windows	Linux	macOS

Proxyvariablen	`[RO][SENS] Get-ChildItem Env: | Where-Object Name -Match 'proxy'`	`[RO][SENS] env | grep -i proxy`	`[RO][SENS] env | grep -i proxy`
WinHTTP-Proxy	`[RO][SENS] netsh winhttp show proxy`	Nicht zutreffend	Nicht zutreffend
Erweiterter WinHTTP-Proxy	`[RO][SENS] netsh winhttp show advproxy`	Nicht zutreffend	Nicht zutreffend
Benutzerproxy	Registry `Internet Settings` prüfen	Desktop- und anwendungsabhängig	`[RO][SENS] scutil --proxy`
HTTP-Proxy	Anwendungsspezifisch	Umgebungs- oder Anwendungskonfiguration	`[RO][SENS] networksetup -getwebproxy "Wi-Fi"`
HTTPS-Proxy	Anwendungsspezifisch	Umgebungs- oder Anwendungskonfiguration	`[RO][SENS] networksetup -getsecurewebproxy "Wi-Fi"`
PAC-URL	Registry beziehungsweise WinHTTP	Desktop- und anwendungsabhängig	`[RO][SENS] networksetup -getautoproxyurl "Wi-Fi"`
Proxy-Ausnahmen	Registry beziehungsweise WinHTTP	`NO_PROXY` prüfen	`[RO][SENS] networksetup -getproxybypassdomains "Wi-Fi"`

Windows-Benutzereinstellungen:

```
[RO][SENS] Get-ItemProperty `
    "HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet Settings" |
    Select-Object ProxyEnable, ProxyServer,
                  ProxyOverride, AutoConfigURL, AutoDetect
```

Expliziter Proxytest:

```
[TEST][SENS] curl -v \
    --proxy http://PROXY:PORT \
    https://HOST/
```

Proxy ausdrücklich umgehen:

```
[TEST][SENS] curl -v --noproxy "*" https://HOST/
```

Ein Proxy-Bypass darf nur ausgeführt werden, wenn die Sicherheitsrichtlinie dies erlaubt.

---

24. VPN prüfen

Aufgabe	Windows	Linux	macOS
VPN-Profile	`[RO][SENS] Get-VpnConnection`	`[RO][SENS] nmcli connection show`	`[RO][SENS] scutil --nc list`
Geräteweite Profile	`[RO][PRIV][SENS] Get-VpnConnection -AllUserConnection`	Clientabhängig	MDM- und clientabhängig
Aktive RAS-Verbindung	`[RO] rasdial`	`[RO][SENS] nmcli connection show --active`	`[RO][SENS] scutil --nc status "VPN-NAME"`
Virtuelle Schnittstellen	`[RO] Get-NetAdapter -IncludeHidden`	`[RO] ip -br link`	`[RO] ifconfig`
VPN-Routen	`[RO] Get-NetRoute`	`[RO] ip route show`	`[RO] netstat -rn`
DNS über VPN	`[RO][SENS] Get-DnsClientServerAddress`	`[RO][SENS] resolvectl status`	`[RO][SENS] scutil --dns`
WireGuard	Falls installiert: `[RO][PRIV][SENS] wg show`	`[RO][PRIV][SENS] sudo wg show`	Falls installiert: `[RO][PRIV][SENS] sudo wg show`
RAS-Protokoll	`[RO][PRIV][SENS] Get-WinEvent -LogName "Microsoft-Windows-RasClient/Operational"`	Clientabhängig	Clientabhängig
NetworkManager-Protokoll	Nicht zutreffend	`[RO][PRIV][SENS] sudo journalctl -u NetworkManager --since "-30 minutes"`	Nicht zutreffend

Windows – Split Tunneling prüfen:

```
[RO][SENS] Get-VpnConnection |
    Select-Object Name, ConnectionStatus,
                  SplitTunneling, TunnelType, ServerAddress
```

---

25. WLAN prüfen

Aufgabe	Windows	Linux	macOS
Aktuelle Verbindung	`[RO][SENS] netsh wlan show interfaces`	`[RO][SENS] iw dev INTERFACE link`	`[RO][SENS] networksetup -getairportnetwork INTERFACE`
Sichtbare Netze	`[RO][SENS] netsh wlan show networks mode=bssid`	`[TEST][SENS] nmcli device wifi list`	Wireless Diagnostics beziehungsweise `wdutil` falls unterstützt
WLAN-Schnittstellen	`[RO] Get-NetAdapter`	`[RO] iw dev`	`[RO] networksetup -listallhardwareports`
Treiber	`[RO][SENS] netsh wlan show drivers`	`[RO][SENS] lspci -k` beziehungsweise `lsusb`	`[RO][SENS] system_profiler SPAirPortDataType`

Aufgabe	Windows	Linux	macOS
Fähigkeiten	`[RO][SENS] netsh wlan show wirelesscapabilities`	`[RO][SENS] iw list`	`[RO][SENS] system_profiler SPAirPortDataType`
Gespeicherte Profile	`[RO][SENS] netsh wlan show profiles`	`[RO][SENS] nmcli connection show`	WLAN-Einstellungen beziehungsweise MDM
Stationsstatistik	Informationen unter `show interfaces`	`[RO][SENS] iw dev INTERFACE station dump`	Falls unterstützt: `[RO][PRIV][SENS] sudo wdutil info`
Regulierungsdomäne	Treiber- und Regionseinstellungen	`[RO] iw reg get`	Systemverwaltet
WLAN-Protokoll	WLAN-AutoConfig-Ereignisse	NetworkManager und Kernel	Wireless Diagnostics
WLAN-Bericht	`[RO][PRIV][FILE][SENS] netsh wlan show wlanreport`	Protokolle getrennt sichern	Wireless Diagnostics erzeugt ein Diagnosearchiv

Windows-Ereignisse:

```
[RO][PRIV][SENS] Get-WinEvent `
    -LogName "Microsoft-Windows-WLAN-AutoConfig/Operational" `
    -MaxEvents 100
```

---

26. Leistung, Paketverlust und Bandbreite

Aufgabe	Windows	Linux	macOS
100 Pingtests	`[TEST] ping -n 100 IP`	`[TEST] ping -c 100 IP`	`[TEST] ping -c 100 IP`
Pfad und Verlust	`[TEST] pathping /n IP`	Falls installiert: `[TEST] mtr -n -r -c 100 IP`	Falls installiert: `[TEST] mtr -n -r -c 100 IP`
TCP-Durchsatz	`[TEST] iperf3 -c SERVER_IP`	`[TEST] iperf3 -c SERVER_IP`	`[TEST] iperf3 -c SERVER_IP`
Gegenrichtung	`[TEST] iperf3 -c SERVER_IP -R`	`[TEST] iperf3 -c SERVER_IP -R`	`[TEST] iperf3 -c SERVER_IP -R`
30 Sekunden	`[TEST] iperf3 -c SERVER_IP -t 30`	`[TEST] iperf3 -c SERVER_IP -t 30`	`[TEST] iperf3 -c SERVER_IP -t 30`
Parallele Streams	`[TEST] iperf3 -c SERVER_IP -P 4`	`[TEST] iperf3 -c SERVER_IP -P 4`	`[TEST] iperf3 -c SERVER_IP -P 4`
UDP 10 Mbit/s	`[TEST] iperf3 -c SERVER_IP -u -b 10M -t 30`	`[TEST] iperf3 -c SERVER_IP -u -b 10M -t 30`	`[TEST] iperf3 -c SERVER_IP -u -b 10M -t 30`
JSON-Ausgabe	`[TEST][FILE][SENS] iperf3 -c SERVER_IP -J > test.json`	`[TEST][FILE][SENS] iperf3 -c SERVER_IP -J > test.json`	`[TEST][FILE][SENS] iperf3 -c SERVER_IP -J > test.json`
Schnittstellenfehler	`[RO] Get-NetAdapterStatistics`	`[RO] ip -s link`	`[RO] netstat -ib`

Lasttests dürfen nur auf freigegebenen Systemen und mit kontrollierter Datenrate durchgeführt werden.

---

27. Docker- und Containernetzwerke

Aufgabe	Befehl
Laufende Container und Ports	`[RO] docker ps --format 'table {{.Names}}\t{{.Ports}}'`
Portzuordnung	`[RO] docker port CONTAINER`
Netzwerke	`[RO] docker network ls`
Netzwerkdetails	`[RO][SENS] docker network inspect NETZWERK`
Containerdetails	`[RO][SENS] docker inspect CONTAINER`
Containerprotokoll	`[RO][SENS] docker logs --tail 100 CONTAINER`
Containerprozesse	`[RO] docker top CONTAINER`
Container-IP-Konfiguration	Über `docker inspect` und Netzwerkdefinition prüfen
Veröffentlichte Hostports	`[RO] docker ps --format 'table {{.Names}}\t{{.Ports}}'`

Typische Portzuordnung:

```
0.0.0.0:8080->80/tcp
```

Nur Loopback:

```
127.0.0.1:8080->80/tcp
```

`EXPOSE` im Dockerfile veröffentlicht einen Port nicht automatisch auf dem Host.

---

28. Prozesse und Dienste prüfen

Aufgabe	Windows	Linux	macOS
Prozesse	`[RO] Get-Process`	`[RO] ps aux`	`[RO] ps aux`
Prozess per PID	`[RO] Get-Process -Id PID`	`[RO] ps -fp PID`	`[RO] ps -p PID -o pid,ppid,user,command`
Dienste	`[RO] Get-Service`	`[RO] systemctl --type=service`	`[RO] launchctl list`

Aufgabe	Windows	Linux	macOS
Bestimmter Dienst	`[RO] Get-Service -Name DIENST`	`[RO] systemctl status DIENST`	Dienstabhängig
Aktuelle Logs	Ereignisprotokoll beziehungsweise Anwendung	`[RO][PRIV][SENS] sudo journalctl -u DIENST --since "-30 minutes"`	Unified Logging beziehungsweise Anwendung
CPU-Prozesse	`[RO] Get-Process | Sort-Object CPU -Descending`	`[RO] top`	`[RO] top -l 1`

Linux-Dienstkonfiguration:

```
[RO][PRIV][SENS] sudo systemctl cat DIENST
```

Linux-Dienstumgebung:

```
[RO][PRIV][SENS] sudo systemctl show DIENST \
    --property=Environment \
    --property=EnvironmentFiles
```

---

29. Paketmitschnitt – Schnittstellen und Basisbefehle

Aufgabe	Windows	Linux	macOS
Capture-Schnittstellen	`[RO] dumpcap -D`	`[RO] tcpdump -D`	`[RO] tcpdump -D`
Vollständige Aufzeichnung	`[TEST][PRIV][FILE][SENS] dumpcap -i NUMMER -w capture.pcapng`	`[TEST][PRIV][FILE][SENS] sudo tcpdump -i INTERFACE -nn -w capture.pcap`	`[TEST][PRIV][FILE][SENS] sudo tcpdump -i INTERFACE -nn -w capture.pcap`
Bestimmter Host	Dumpcap mit `-f "host IP"`	tcpdump mit `'host IP'`	tcpdump mit `'host IP'`
TCP-Port 443	Dumpcap mit `-f "tcp port 443"`	tcpdump mit `'tcp port 443'`	tcpdump mit `'tcp port 443'`
DNS	Dumpcap mit `-f "port 53"`	tcpdump mit `'port 53'`	tcpdump mit `'port 53'`
DHCPv4	Dumpcap mit `-f "udp port 67 or udp port 68"`	tcpdump mit entsprechendem Filter	tcpdump mit entsprechendem Filter
PCAP lesen	`[RO][SENS] tshark -r capture.pcapng`	`[RO][SENS] tcpdump -nn -r capture.pcap`	`[RO][SENS] tcpdump -nn -r capture.pcap`
Dateiinformationen	`[RO][SENS] capinfos capture.pcapng`	`[RO][SENS] capinfos capture.pcap`	`[RO][SENS] capinfos capture.pcap`

Windows – 60 Sekunden:

```
[TEST][PRIV][FILE][SENS] dumpcap -i NUMMER `
    -a duration:60 `
    -w capture.pcapng
```

Linux oder macOS – bestimmter Datenfluss:

```
[TEST][PRIV][FILE][SENS] sudo tcpdump -i INTERFACE -nn \
    'host CLIENT_IP and host SERVER_IP and tcp port PORT' \
    -w capture.pcap
```

---

30. Paketmitschnitt – Ringspeicher

Dumpcap – alle fünf Minuten wechseln, zwölf Dateien:

```
[TEST][PRIV][FILE][SENS] dumpcap -i NUMMER `
    -b duration:300 `
    -b files:12 `
    -w ringbuffer.pcapng
```

tcpdump – größenbasiert, maximal zehn Dateien:

```
[TEST][PRIV][FILE][SENS] sudo tcpdump -i INTERFACE -nn \
    -C 100 \
    -W 10 \
    -w ringbuffer.pcap
```

Vor einer Langzeitaufzeichnung prüfen:

- verfügbaren Speicherplatz,
- Datenschutz,
- Dateiberechtigungen,
- Rotationsverhalten der installierten Version,
- erwartete Datenrate,
- Aufbewahrungsdauer.

---

31. Windows Pktmon

Aufgabe	Befehl
Hilfe	`[RO] pktmon help`
Status	`[RO][PRIV] pktmon status`
Komponenten	`[RO][PRIV][SENS] pktmon list`
Aufzeichnung starten	`[TEST][PRIV][FILE][SENS] pktmon start --capture --pkt-size 0 --file-name pktmon.etl`
Aufzeichnung beenden	`[TEST][PRIV][FILE][SENS] pktmon stop`
Zähler	`[RO][PRIV][SENS] pktmon counters`
Nach PCAPNG konvertieren	`[RO][FILE][SENS] pktmon etl2pcap pktmon.etl --out pktmon.pcapng`
Nur Drops konvertieren	`[RO][FILE][SENS] pktmon etl2pcap pktmon.etl --drop-only --out pktmon-drops.pcapng`

Pktmon kann Pakete an mehreren Stellen des Windows-Netzwerkstacks erfassen. Die ursprüngliche ETL-Datei enthält Informationen, die bei der PCAPNG-Konvertierung teilweise verloren gehen können.

---

32. Wichtige Wireshark Display Filter

Aufgabe	Display Filter
IPv4-Adresse	`ip.addr == 192.0.2.20`
IPv6-Adresse	`ipv6.addr == 2001:db8::20`
Zwei Systeme	`ip.addr == 192.0.2.20 && ip.addr == 192.0.2.53`
TCP-Port	`tcp.port == 443`
UDP-Port	`udp.port == 53`
ARP	`arp`
IPv6 Neighbor Discovery	`icmpv6`
DHCPv4	`dhcp`
DHCPv6	`dhcpv6`
DNS	`dns`
DNS-Anfrage	`dns.flags.response == 0`
DNS-Antwort	`dns.flags.response == 1`

Aufgabe	Display Filter
HTTP-Anfrage	`http.request`
HTTP-Antwort	`http.response`
TLS-Handshake	`tls.handshake`
TCP-SYN	`tcp.flags.syn == 1`
Erstes SYN	`tcp.flags.syn == 1 && tcp.flags.ack == 0`
TCP-Reset	`tcp.flags.reset == 1`
TCP-FIN	`tcp.flags.fin == 1`
Wiederholung	`tcp.analysis.retransmission`
Doppelte ACKs	`tcp.analysis.duplicate_ack`
Out-of-Order	`tcp.analysis.out_of_order`
Zero Window	`tcp.analysis.zero_window`
ICMP	`icmp`
ICMPv6	`icmpv6`
IKE	`isakmp`
IPsec NAT-T	`udp.port == 4500`
ESP	`esp`

33. TShark-Auswertung

Datei mit Display Filter lesen:

```
[RO][SENS] tshark -r capture.pcapng \
    -Y 'tcp.port == 443'
```

DNS-Anfragen:

```
[RO][SENS] tshark -r capture.pcapng \
    -Y 'dns.flags.response == 0' \
    -T fields \
    -e frame.time \
    -e ip.src \
    -e dns.qry.name
```

HTTP-Statuscodes:

```
[RO][SENS] tshark -r capture.pcapng \
    -Y 'http.response' \
    -T fields \
    -e frame.time \
    -e ip.src \
    -e ip.dst \
    -e http.response.code
```

TCP-Wiederholungen:

```
[RO][SENS] tshark -r capture.pcapng \
    -Y 'tcp.analysis.retransmission'
```

Mitschnitte zusammenführen:

```
[RO][FILE][SENS] mergecap \
    -w combined.pcapng \
    client.pcapng \
    server.pcapng
```

---

34. Typische Symptom-zu-Befehl-Zuordnung

Symptom	Erste sinnvolle Prüfungen
Kein Netzwerk	Adapterstatus, IP-Konfiguration, Gateway
Adresse `169.254.x.x`	DHCP, VLAN, DHCP-Paketmitschnitt
Gateway nicht erreichbar	ARP/NDP, Subnetz, WLAN, Kabel, Switchport
IP funktioniert, Hostname nicht	DNS-Server, `Resolve-DnsName` `dig`
Host antwortet, Dienst nicht	Listener, Porttest, Firewall
`Connection refused`	Listener und Prozess prüfen
Timeout	Route, Firewall, Rückweg und Paketmitschnitt
Nur ein Client betroffen	Clientkonfiguration mit Vergleichssystem vergleichen
Nur ein VLAN betroffen	VLAN, Gateway, DHCP-Relay und ACL

Symptom	Erste sinnvolle Prüfungen
Nur extern nicht erreichbar	NAT, Firewall, öffentliche Adresse und CGNAT
Nur intern über öffentlichen Namen fehlerhaft	Hairpin NAT oder Split DNS
Browser funktioniert, Dienst nicht	WinHTTP, Proxyvariablen und Benutzerkontext
VPN verbunden, internes Ziel nicht erreichbar	Tunneladresse, Route, DNS und Netzüberlappung
WLAN verbunden, kein Internet	DHCP, Gateway, DNS und Captive Portal
Verbindung langsam	Gatewayping, Fehlerzähler, iperf3 und Auslastung
Kleine Pakete funktionieren, große nicht	MTU, ICMP und Path-MTU-Discovery
Sporadische Abbrüche	Langzeitmessung, Logs, Link-Flaps und Paketmitschnitt
TCP funktioniert, UDP nicht	UDP-spezifischen Diensttest, Firewall und Mitschnitt
IPv4 funktioniert, IPv6 nicht	IPv6-Adresse, Route, DNS-AAAA und ICMPv6
Container lokal erreichbar, extern nicht	Port Publishing, Bind-Adresse, Host-Firewall und NAT

---

35. Kompakter Diagnoseblock pro Betriebssystem

Windows – rein lesender Basisblock:

```
[RO] Get-Date -Format o
[RO] hostname
[RO] Get-NetAdapter
[RO][SENS] Get-NetIPConfiguration -All
[RO] Get-NetRoute
[RO][SENS] Get-DnsClientServerAddress
[RO][SENS] Get-NetNeighbor
[RO] Get-NetTCPConnection
[RO] Get-NetUDPEndpoint
[RO] Get-NetAdapterStatistics
[RO] Get-NetConnectionProfile
[RO] Get-NetFirewallProfile
```

Linux – rein lesender Basisblock:

```
[RO] date --iso-8601=seconds
[RO] hostnamectl
```

```
[RO] ip -br link
[RO][SENS] ip -br address
[RO] ip route show
[RO] ip -6 route show
[RO][SENS] ip neigh show
[RO][SENS] resolvectl status
[RO] ss -tuln
[RO] ip -s link
```

Falls `resolvectl` nicht vorhanden ist:

```
[RO][SENS] cat /etc/resolv.conf
```

macOS – rein lesender Basisblock:

```
[RO] date "+%Y-%m-%dT%H:%M:%S%z"
[RO] sw_vers
[RO] scutil --get ComputerName
[RO] networksetup -listallhardwareports
[RO][SENS] ifconfig
[RO] netstat -rn
[RO][SENS] arp -an
[RO][SENS] ndp -an
[RO][SENS] scutil --dns
[RO] netstat -anv
[RO] netstat -ib
```

Die Kennzeichnungen `[RO]` und `[SENS]` sind Dokumentationsmarkierungen und werden beim tatsächlichen Kopieren eines Befehls nicht mit eingegeben.

---

36. Dokumentationsvorlage für eine Netzwerkdiagnose

```
Störung:
Zeitpunkt:
Zeitzone:
Benutzer:
```

Standort:

Client:

Betriebssystem:

Schnittstelle:

MAC-Adresse:

Linkstatus:

Linkgeschwindigkeit:

WLAN-SSID:

WLAN-BSSID:

VLAN:

IPv4-Adresse:

IPv4-Präfix:

IPv4-Gateway:

IPv6-Adresse:

IPv6-Präfix:

IPv6-Gateway:

DHCP-Server:

DNS-Server:

DNS-Suffixe:

Zielhostname:

Aufgelöste Zieladresse:

Transportprotokoll:

Zielport:

Anwendung:

Loopbacktest:

Gatewaytest:

Interner IP-Test:

Externer IP-Test:

DNS-Test:

Routenergebnis:

Porttest:

Anwendungstest:

Listener vorhanden:

Zugehöriger Prozess:

Firewallstatus:

Passende Firewallregel:

NAT beteiligt:

Proxy beteiligt:

VPN beteiligt:

Container beteiligt:

Paketverlust:

Latenz:

Jitter:

TCP-Durchsatz:

UDP-Verlust:

Schnittstellenfehler:

Paketmitschnitt:

Messpunkt:

Capture Filter:

Relevante Frames:

Systemprotokolle:

Festgestellte Ursache:

Genehmigte Änderung:

Rückfallplan:

Ergebnis der Nachprüfung:

---

37. Abschließende Kontrollfragen

- Wurde die richtige Schnittstelle geprüft?
- Sind IPv4 und IPv6 getrennt betrachtet worden?
- Stimmen Adresse, Präfix, Gateway und DNS?
- Ist das Ziel über IP-Adresse erreichbar?
- Wird der Hostname korrekt aufgelöst?
- Führt die Route über die erwartete Schnittstelle?
- Ist der benötigte Port geöffnet?
- Lauscht der richtige Prozess?
- Funktioniert das eigentliche Anwendungsprotokoll?
- Wurde TCP nicht mit UDP verwechselt?
- Wurden Firewall, NAT, Proxy und VPN berücksichtigt?

- Wurde bei Containern zwischen Host- und Containerport unterschieden?
- Wurde ein Timeout nicht automatisch als Firewallfehler bewertet?
- Wurde Paketverlust an Zwischenhops korrekt interpretiert?
- Wurden Hin- und Rückrichtung geprüft?
- Wurde die Messung mit einem funktionierenden System verglichen?
- Wurde vor einer Änderung der Istzustand dokumentiert?
- Wurde nach der Änderung mit derselben Methode erneut geprüft?
- Enthalten Ausgaben oder Paketmitschnitte sensible Daten?
- Ist die festgestellte Ursache durch konkrete Messwerte belegt?

---

38. Quellen und Befehlsreferenzen

- Microsoft Learn – Windows-Netzwerk-Cmdlets:
  https://learn.microsoft.com/powershell/module/nettcpip/
- Microsoft Learn – Windows-Firewall-Cmdlets:
  https://learn.microsoft.com/powershell/module/netsecurity/
- Microsoft Learn – `netsh wlan`:
  https://learn.microsoft.com/windows-server/administration/windows-commands/netsh-wlan
- Microsoft Learn – `ping`:
  https://learn.microsoft.com/windows-server/administration/windows-commands/ping
- Microsoft Learn – `tracert`:
  https://learn.microsoft.com/windows-server/administration/windows-commands/tracert
- Microsoft Learn – `pathping`:
  https://learn.microsoft.com/windows-server/administration/windows-commands/pathping
- Microsoft Learn – Pktmon:
  https://learn.microsoft.com/windows-server/administration/windows-commands/pktmon
- Linux-Handbuch – iproute2:
  https://man7.org/linux/man-pages/man8/ip.8.html
- Linux-Handbuch – `ss`:
  https://man7.org/linux/man-pages/man8/ss.8.html
- NetworkManager – `nmcli`:
  https://networkmanager.dev/docs/api/latest/nmcli.html
- firewalld – Dokumentation:
  https://firewalld.org/documentation/
- Netfilter – nftables:
  https://netfilter.org/projects/nftables/manpage.html
- Apple – `networksetup`:
  Auf dem Mac lokal mit `man networksetup`
- Apple – Netzwerk- und WLAN-Diagnose:
  https://support.apple.com/de-de/guide/mac-help/mchlf4de377f/mac

- Wireshark – Benutzerhandbuch: https://www.wireshark.org/docs/wsug_html_chunked/
- Wireshark – Befehlsreferenzen: https://www.wireshark.org/docs/man-pages/
- curl – Handbuch: https://curl.se/docs/manpage.html
- ESnet – iperf3: https://software.es.net/iperf/
- WireGuard – Dokumentation: https://www.wireguard.com/quickstart/
- OpenVPN – Referenzhandbuch: https://openvpn.net/community-resources/reference-manual-for-openvpn-2-6/
- Docker – Networking: https://docs.docker.com/engine/network/