2.2 PowerShell-Netzwerkdiagnose

PowerShell stellt Netzwerkdaten als strukturierte Objekte bereit. Dadurch lassen sich Informationen gezielt filtern, sortieren, vergleichen und exportieren.

Die Cmdlets dieser Seite gehören zu Windows-Modulen wie `NetAdapter`, `NetTCPIP`, `DnsClient` und `NetSecurity`. Sie sind für die Netzwerkdiagnose unter Windows vorgesehen.

Kennzeichnungen

Kennzeichnung	Bedeutung
`[RO]`	Nur lesender Befehl; verändert keine Konfiguration
`[TEST]`	Führt einen aktiven Netzwerk- oder Verbindungstest aus
`[PRIV]`	Benötigt möglicherweise eine PowerShell mit Administratorrechten
`[CHANGE]`	Verändert einen Zustand oder eine Konfiguration
`[DISRUPT]`	Kann eine bestehende Verbindung oder einen Dienst unterbrechen
`[FILE]`	Schreibt Informationen in eine Datei
`[SENS]`	Ausgabe kann sensible System-, Netzwerk- oder Prozessdaten enthalten

> Für die erste Diagnose möglichst mit `[RO]`-Befehlen beginnen. Veränderungen sollten erst erfolgen, wenn die Ursache ausreichend eingegrenzt und eine Rückfallmöglichkeit vorhanden ist.

---

1. PowerShell und benötigte Netzwerkmodule prüfen

Bevor ein Cmdlet verwendet wird, sollte geprüft werden, welche PowerShell-Version und welche Windows-Netzwerkmodule vorhanden sind.

Aufgabe	PowerShell-Befehl
PowerShell-Version anzeigen	`[RO] $PSVersionTable`
Betriebssysteminformationen anzeigen	`[RO] Get-ComputerInfo | Select-Object WindowsProductName, WindowsVersion, OsBuildNumber`
Netzwerkmodule suchen	`[RO] Get-Module -ListAvailable NetAdapter, NetTCPIP, DnsClient, NetSecurity`

Aufgabe	PowerShell-Befehl
Befehle eines Moduls auflisten	`[RO] Get-Command -Module NetTCPIP`
Hilfe zu einem Cmdlet anzeigen	`[RO] Get-Help Test-NetConnection -Full`
Verwendungsbeispiele anzeigen	`[RO] Get-Help Resolve-DnsName -Examples`
Syntax eines Cmdlets anzeigen	`[RO] Get-Command Get-NetAdapter -Syntax`

Typische Fehler

Meldung oder Beobachtung	Mögliche Bedeutung
`The term ... is not recognized`	Cmdlet oder Modul ist nicht vorhanden beziehungsweise nicht geladen
`Access denied`	Administratorrechte oder zusätzliche Berechtigungen erforderlich
Keine Ausgabe	Filter ist möglicherweise zu eng oder es existiert kein passendes Objekt
Einzelne Eigenschaften fehlen	Unterschiedliche Windows-, PowerShell- oder Modulversion

2. Netzwerkadapter und Verbindungsstatus prüfen

`Get-NetAdapter` zeigt physische und virtuelle Netzwerkadapter einschließlich Status, Geschwindigkeit, MAC-Adresse und Schnittstellenindex.

Aufgabe	PowerShell-Befehl
Alle sichtbaren Netzwerkadapter anzeigen	`[RO] Get-NetAdapter`
Auch ausgeblendete Adapter anzeigen	`[RO] Get-NetAdapter -IncludeHidden`
Wichtige Eigenschaften auswählen	`[RO] Get-NetAdapter | Select-Object Name, InterfaceDescription, Status, LinkSpeed, MacAddress, ifIndex`
Nur aktive Adapter anzeigen	`[RO] Get-NetAdapter | Where-Object Status -eq "Up"`
Nicht aktive Adapter anzeigen	`[RO] Get-NetAdapter | Where-Object Status -ne "Up"`
Adapter nach Status gruppieren	`[RO] Get-NetAdapter -IncludeHidden | Group-Object Status`
Detailinformationen eines Adapters anzeigen	`[RO] Get-NetAdapter -Name "Ethernet" | Format-List *`
Adapterstatistik anzeigen	`[RO] Get-NetAdapterStatistics`
Fehler- und Verwerfungszähler anzeigen	`[RO] Get-NetAdapterStatistics | Select-Object Name, ReceivedPacketErrors, OutboundPacketErrors, ReceivedDiscardedPackets, OutboundDiscardedPackets`

Aufgabe	PowerShell-Befehl
Treiberinformationen anzeigen	`[RO] Get-NetAdapter | Select-Object Name, DriverDescription, DriverVersion, DriverDate`
Verbindungstyp und Netzwerkprofil anzeigen	`[RO] Get-NetConnectionProfile`

Wichtige Eigenschaften

Eigenschaft	Bedeutung
`Status`	Betriebszustand des Adapters
`LinkSpeed`	Ausgehandelte Verbindungsgeschwindigkeit
`MacAddress`	Hardwareadresse des Netzwerkadapters
`ifIndex`	Schnittstellenindex für weitere Cmdlets
`ReceivedPacketErrors`	Fehlerhaft empfangene Pakete
`OutboundPacketErrors`	Fehler beim Senden
`ReceivedDiscardedPackets`	Empfangene, aber verworfene Pakete
`OutboundDiscardedPackets`	Zu sendende, aber verworfene Pakete

Hinweise zur Auswertung

- `Status = Up` bedeutet nur, dass der Adapter aktiv und eine Verbindung erkannt wurde.
- Ein aktiver Adapter bestätigt nicht automatisch eine funktionierende IP-Konfiguration.
- Eine unerwartet niedrige `LinkSpeed` kann auf Kabel-, Port-, Treiber- oder Aushandlungsprobleme hinweisen.
- Steigende Fehler- oder Verwerfungszähler können auf Treiberprobleme, Überlastung oder eine gestörte Verbindung hinweisen.
- Virtuelle Adapter von VPN-, Hyper-V-, Container- oder Sicherheitssoftware müssen von physischen Adaptern unterschieden werden.

---

3. IP-Konfiguration vollständig erfassen

`Get-NetIPConfiguration` liefert eine zusammengefasste Ansicht der IP-Konfiguration. Für einzelne Eigenschaften stehen zusätzliche Cmdlets zur Verfügung.

Aufgabe	PowerShell-Befehl
Aktive IP-Konfiguration anzeigen	`[RO] Get-NetIPConfiguration`
Alle IP-Konfigurationen anzeigen	`[RO] Get-NetIPConfiguration -All`
Ausführliche Ausgabe erzeugen	`[RO] Get-NetIPConfiguration -All | Format-List *`

Aufgabe	PowerShell-Befehl
IPv4-Adressen anzeigen	`[RO] Get-NetIPAddress -AddressFamily IPv4`
IPv6-Adressen anzeigen	`[RO] Get-NetIPAddress -AddressFamily IPv6`
Verwendbare IPv4-Adressen anzeigen	`[RO] Get-NetIPAddress -AddressFamily IPv4 | Where-Object AddressState -eq "Preferred"`
Loopback- und APIPA-Adressen ausblenden	`[RO] Get-NetIPAddress -AddressFamily IPv4 | Where-Object { $_.IPAddress -notlike "127.*" -and $_.IPAddress -notlike "169.254.*" }`
IP-Schnittstellen anzeigen	`[RO] Get-NetIPInterface`
IPv4-Schnittstellen nach Metrik sortieren	`[RO] Get-NetIPInterface -AddressFamily IPv4 | Sort-Object InterfaceMetric`
Standardgateways anzeigen	`[RO] Get-NetRoute -DestinationPrefix "0.0.0.0/0"`
DNS-Serveradressen anzeigen	`[RO] Get-DnsClientServerAddress`
Nur IPv4-DNS-Server anzeigen	`[RO] Get-DnsClientServerAddress -AddressFamily IPv4`
DHCP-Status anzeigen	`[RO] Get-NetIPInterface | Select-Object InterfaceAlias, AddressFamily, Dhcp, ConnectionState`

Wichtige Prüfpunkte

Prüffeld	Erwartung beziehungsweise Fehlerhinweis
IP-Adresse	Muss zum vorgesehenen Netzwerk passen
Präfixlänge	Muss zur Netzmaske des Netzwerks passen
Standardgateway	Muss im Regelfall aus dem lokalen Netz erreichbar sein
DNS-Server	Muss erreichbar und für die benötigten Zonen zuständig sein
DHCP	Muss zur vorgesehenen statischen oder dynamischen Konfiguration passen
InterfaceMetric	Niedrigerer Wert wird bei konkurrierenden Schnittstellen bevorzugt
`169.254.x.x`	Kann auf eine fehlgeschlagene DHCP-Zuweisung hinweisen
Mehrere Standardrouten	Können zu einem unerwarteten Verbindungsweg führen

Kompakte Bestandsaufnahme

```
Get-NetIPConfiguration -All |
    Select-Object InterfaceAlias,
                  InterfaceDescription,
```

```
    NetProfile,
    IPv4Address,
    IPv6Address,
    IPv4DefaultGateway,
    DNSServer
```

4. Erreichbarkeit und TCP-Ports testen

`Test-NetConnection` kombiniert mehrere klassische Netzwerkprüfungen und liefert strukturierte Ergebnisse.

Aufgabe	PowerShell-Befehl
Grundlegende Erreichbarkeit testen	`[TEST] Test-NetConnection 192.0.2.10`
Detaillierte Ausgabe anzeigen	`[TEST] Test-NetConnection 192.0.2.10 -InformationLevel Detailed`
Standardgateway testen	`[TEST] Test-NetConnection 192.0.2.1 -InformationLevel Detailed`
Internet-IP testen	`[TEST] Test-NetConnection 1.1.1.1 -InformationLevel Detailed`
HTTPS-Port testen	`[TEST] Test-NetConnection example.com -Port 443 -InformationLevel Detailed`
SSH-Port testen	`[TEST] Test-NetConnection 192.0.2.10 -Port 22 -InformationLevel Detailed`
RDP-Port testen	`[TEST] Test-NetConnection 192.0.2.10 -Port 3389 -InformationLevel Detailed`
SMB-Port testen	`[TEST] Test-NetConnection 192.0.2.10 -Port 445 -InformationLevel Detailed`
Routenverfolgung durchführen	`[TEST] Test-NetConnection example.com -TraceRoute`
Nur boolesches Testergebnis ausgeben	`[TEST] Test-NetConnection example.com -Port 443 -InformationLevel Quiet`

> Die Adressen `192.0.2.0/24`, `198.51.100.0/24` und `203.0.113.0/24` sind Dokumentationsnetze. Sie müssen in echten Prüfungen durch die tatsächlichen Zieladressen ersetzt werden.

Wichtige Ergebnisfelder

Eigenschaft	Bedeutung
`ComputerName`	Angegebenes Ziel

Eigenschaft	Bedeutung
`RemoteAddress`	Aufgelöste Zieladresse
`NameResolutionResults`	Ergebnisse der Namensauflösung
`SourceAddress`	Verwendete lokale Quelladresse
`InterfaceAlias`	Verwendete Netzwerkschnittstelle
`NetRoute`	Ausgewählte Route
`PingSucceeded`	Ergebnis des ICMP-Tests
`TcpTestSucceeded`	Ergebnis des TCP-Verbindungsversuchs
`RemotePort`	Getesteter Zielport

Interpretation

Ergebnis	Mögliche Bedeutung
`PingSucceeded = False` TCP-Test erfolgreich	ICMP wird möglicherweise blockiert; der getestete Dienst ist trotzdem erreichbar
Namensauflösung schlägt fehl, IP-Test funktioniert	DNS-Problem wahrscheinlich
Gateway erreichbar, externes Ziel nicht erreichbar	Routing, Firewall, Provider oder Upstream-System prüfen
IP-Adresse erreichbar, TCP-Port nicht erreichbar	Dienst, Zielport oder Firewall prüfen
Falsches `InterfaceAlias`	Routingmetrik, VPN oder mehrere aktive Adapter prüfen
Falsche `SourceAddress`	IP-Konfiguration oder Routenauswahl prüfen

---

5. DNS-Auflösung gezielt untersuchen

`Resolve-DnsName` erlaubt präzisere DNS-Abfragen als ein einfacher Verbindungstest.

Aufgabe	PowerShell-Befehl
Namen mit Standardeinstellungen auflösen	`[TEST] Resolve-DnsName example.com`
Nur DNS verwenden	`[TEST] Resolve-DnsName example.com -DnsOnly`
IPv4-Adresse abfragen	`[TEST] Resolve-DnsName example.com -Type A -DnsOnly`
IPv6-Adresse abfragen	`[TEST] Resolve-DnsName example.com -Type AAAA -DnsOnly`
Mailserver abfragen	`[TEST] Resolve-DnsName example.com -Type MX -DnsOnly`
Nameserver abfragen	`[TEST] Resolve-DnsName example.com -Type NS -DnsOnly`

Aufgabe	PowerShell-Befehl
TXT-Einträge abfragen	`[TEST] Resolve-DnsName example.com -Type TXT -DnsOnly`
Reverse-DNS-Abfrage durchführen	`[TEST] Resolve-DnsName 192.0.2.10 -Type PTR -DnsOnly`
Bestimmten DNS-Server verwenden	`[TEST] Resolve-DnsName example.com -Server 192.0.2.53 -DnsOnly`
Nur lokalen DNS-Cache abfragen	`[RO] Resolve-DnsName example.com -CacheOnly`
DNS-Cache anzeigen	`[RO] Get-DnsClientCache`
Cache nach einem Namen durchsuchen	`[RO] Get-DnsClientCache | Where-Object Entry -like "*example.com*"`
Konfigurierte DNS-Server anzeigen	`[RO] Get-DnsClientServerAddress`
DNS-Clientkonfiguration anzeigen	`[RO] Get-DnsClient`

DNS-Vergleichstest

```
Resolve-DnsName example.com -DnsOnly
Resolve-DnsName example.com -Server 192.0.2.53 -DnsOnly
Resolve-DnsName example.com -Server 1.1.1.1 -DnsOnly
```

Auswertung

Beobachtung	Mögliche Ursache
Interner DNS-Server antwortet nicht	DNS-Dienst, Firewall, Routing oder Erreichbarkeit prüfen
Externer DNS-Server funktioniert, interner nicht	Problem wahrscheinlich beim internen Resolver
Interne Namen funktionieren extern nicht	Normal, wenn die Zone nur intern vorhanden ist
Unterschiedliche Antworten verschiedener Resolver	Split-DNS, Cache, Replikationsverzug oder unterschiedliche Zonen
`NXDOMAIN`	Name existiert aus Sicht des verwendeten DNS-Servers nicht
`SERVFAIL`	DNS-Server konnte die Anfrage nicht erfolgreich verarbeiten
Timeout	DNS-Server nicht erreichbar, blockiert oder zu langsam
Auflösung funktioniert nur aus dem Cache	Aktuelle DNS-Kommunikation möglicherweise gestört

> Öffentliche DNS-Server dürfen nicht unüberlegt zum Test interner Namen verwendet werden. Interne Hostnamen, Domänennamen und Strukturen können sensible Informationen darstellen.

6. Routing und ausgewählten Netzwerkweg prüfen

Aufgabe	PowerShell-Befehl
Gesamte Routingtabelle anzeigen	`[RO] Get-NetRoute`
IPv4-Routen anzeigen	`[RO] Get-NetRoute -AddressFamily IPv4`
IPv6-Routen anzeigen	`[RO] Get-NetRoute -AddressFamily IPv6`
Standardroute anzeigen	`[RO] Get-NetRoute -DestinationPrefix "0.0.0.0/0"`
Routen nach Metrik sortieren	`[RO] Get-NetRoute -AddressFamily IPv4 | Sort-Object RouteMetric`
Route einer Schnittstelle anzeigen	`[RO] Get-NetRoute -InterfaceAlias "Ethernet"`
Wahrscheinlich verwendete Route bestimmen	`[RO] Find-NetRoute -RemoteIPAddress 1.1.1.1`
Schnittstellenmetriken anzeigen	`[RO] Get-NetIPInterface -AddressFamily IPv4 | Sort-Object InterfaceMetric`
Routenverfolgung durchführen	`[TEST] Test-NetConnection example.com -TraceRoute`

Wichtige Eigenschaften

Eigenschaft	Bedeutung
`DestinationPrefix`	Zielnetz der Route
`NextHop`	Nächster Router beziehungsweise Gateway
`InterfaceAlias`	Verwendete Netzwerkschnittstelle
`RouteMetric`	Metrik der einzelnen Route
`InterfaceMetric`	Metrik der Schnittstelle
`Publish`	Gibt an, ob die Route veröffentlicht wird
`Protocol`	Herkunft beziehungsweise Routingprotokoll der Route

Typische Fehlerbilder

- Mehrere Standardrouten konkurrieren miteinander.
- Ein VPN installiert eine bevorzugte Route.
- Eine Route verwendet den falschen Adapter.
- Das Standardgateway befindet sich nicht im erwarteten lokalen Netz.
- Eine spezifischere Route überschreibt die allgemeine Standardroute.
- Eine niedrige Metrik führt zu einem unerwarteten Netzwerkweg.

7. ARP- und IPv6-Nachbartabelle prüfen

`Get-NetNeighbor` zeigt bekannte Nachbarn auf direkt angeschlossenen Netzen. Bei IPv4 entspricht dies funktional weitgehend der ARP-Tabelle; bei IPv6 werden Einträge des Neighbor Discovery Protocols angezeigt.

Aufgabe	PowerShell-Befehl
Gesamte Nachbartabelle anzeigen	`[RO] Get-NetNeighbor`
IPv4-Nachbarn anzeigen	`[RO] Get-NetNeighbor -AddressFamily IPv4`
IPv6-Nachbarn anzeigen	`[RO] Get-NetNeighbor -AddressFamily IPv6`
Nachbarn eines Adapters anzeigen	`[RO] Get-NetNeighbor -InterfaceAlias "Ethernet"`
Einträge nach Zustand sortieren	`[RO] Get-NetNeighbor | Sort-Object State, InterfaceAlias`
Bestimmte IP-Adresse suchen	`[RO] Get-NetNeighbor -IPAddress 192.0.2.10`
Erreichbare Einträge anzeigen	`[RO] Get-NetNeighbor | Where-Object State -eq "Reachable"`
Unvollständige Einträge anzeigen	`[RO] Get-NetNeighbor | Where-Object State -eq "Incomplete"`

Wichtige Zustände

Zustand	Bedeutung
`Reachable`	Nachbar wurde kürzlich erfolgreich erreicht
`Stale`	Eintrag ist vorhanden, wurde aber länger nicht bestätigt
`Delay`	Erreichbarkeitsprüfung wird verzögert
`Probe`	Aktive Erreichbarkeitsprüfung läuft
`Incomplete`	Adressauflösung konnte noch nicht abgeschlossen werden
`Permanent`	Statischer beziehungsweise dauerhafter Eintrag

Fehlerhinweise

- `Incomplete` kann auf ein nicht erreichbares Ziel, falsches VLAN, Layer-2-Probleme oder eine falsche IP-Konfiguration hinweisen.
- Wechselnde MAC-Adressen zu derselben IP-Adresse können auf doppelte IP-Adressen, Hochverfügbarkeit oder Sicherheitsprobleme hinweisen.
- Kein Eintrag bedeutet nicht automatisch einen Fehler. Möglicherweise wurde das Ziel noch nicht angesprochen oder befindet sich nicht im lokalen Netz.

---

8. TCP-Verbindungen, offene Ports und Prozesse untersuchen

`Get-NetTCPConnection` zeigt lokale TCP-Endpunkte und bestehende TCP-Verbindungen.

Aufgabe	PowerShell-Befehl
Alle TCP-Verbindungen anzeigen	`[RO] Get-NetTCPConnection`
Lauschende TCP-Ports anzeigen	`[RO] Get-NetTCPConnection -State Listen`
Bestehende Verbindungen anzeigen	`[RO] Get-NetTCPConnection -State Established`
Verbindungen zu einem Zielport suchen	`[RO] Get-NetTCPConnection -RemotePort 443`
Lokalen Port untersuchen	`[RO] Get-NetTCPConnection -LocalPort 443`
Nach Zustand gruppieren	`[RO] Get-NetTCPConnection | Group-Object State`
Verbindungen sortiert anzeigen	`[RO] Get-NetTCPConnection | Sort-Object State, LocalPort`
UDP-Endpunkte anzeigen	`[RO] Get-NetUDPEndpoint`
Prozess über PID ermitteln	`[RO] Get-Process -Id 1234`

Lauschende Ports mit Prozessnamen anzeigen

```
Get-NetTCPConnection -State Listen |
    Select-Object LocalAddress,
                  LocalPort,
                  OwningProcess,
                  @{
                     Name = "ProcessName"
                     Expression = {
                        (Get-Process -Id $_.OwningProcess -ErrorAction SilentlyContinue).ProcessName
                     }
                  } |
    Sort-Object LocalPort
```

Bestehende Verbindungen mit Prozessnamen anzeigen

```
Get-NetTCPConnection -State Established |
    Select-Object LocalAddress,
                  LocalPort,
                  RemoteAddress,
                  RemotePort,
                  OwningProcess,
                  @{
                     Name = "ProcessName"
                     Expression = {
```

```
            (Get-Process -Id $_.OwningProcess -ErrorAction SilentlyContinue).ProcessName
        }
    }
```

Interpretation

Beobachtung	Mögliche Bedeutung
Erwarteter Port fehlt im Zustand `Listen`	Dienst läuft nicht, lauscht auf anderem Port oder konnte den Port nicht binden
Port lauscht nur auf `127.0.0.1`	Dienst ist nur lokal erreichbar
Port lauscht auf `0.0.0.0`	Dienst lauscht grundsätzlich auf allen IPv4-Schnittstellen
Port lauscht auf `::`	Dienst lauscht grundsätzlich auf IPv6 und möglicherweise abhängig von der Anwendung auch auf IPv4
Viele Verbindungen im Zustand `SYN_SENT`	Ziel, Route, Dienst oder Firewall antwortet möglicherweise nicht
Viele Verbindungen im Zustand `TIME_WAIT`	Kann bei vielen kurzlebigen TCP-Verbindungen normal sein
Unerwarteter Prozess lauscht auf einem Port	Dienstzuordnung und Sicherheitslage prüfen

> Eine Verbindung im Zustand `Listen` bestätigt nur, dass lokal ein Prozess auf dem Port wartet. Sie bestätigt nicht, dass der Port aus einem anderen Netz erreichbar ist.

---

9. Windows-Firewall diagnostizieren

Die Firewall sollte zunächst nur ausgelesen werden. Ein vollständiges Abschalten der Firewall ist kein geeigneter erster Diagnoseschritt.

Aufgabe	PowerShell-Befehl
Firewallprofile anzeigen	`[RO] Get-NetFirewallProfile`
Status der Profile kompakt anzeigen	`[RO] Get-NetFirewallProfile | Select-Object Name, Enabled, DefaultInboundAction, DefaultOutboundAction`
Aktivierte Regeln anzeigen	`[RO] Get-NetFirewallRule -Enabled True`
Aktivierte Blockierungsregeln anzeigen	`[RO] Get-NetFirewallRule -Enabled True -Action Block`

Aufgabe	PowerShell-Befehl
Eingehende Regeln anzeigen	`[RO] Get-NetFirewallRule -Direction Inbound`
Regel anhand des Anzeigenamens suchen	`[RO] Get-NetFirewallRule -DisplayName "*Remote Desktop*"`
Portfilter anzeigen	`[RO] Get-NetFirewallPortFilter`
Regeln für lokalen Port 443 suchen	`[RO] Get-NetFirewallPortFilter | Where-Object LocalPort -eq 443 | Get-NetFirewallRule`
Regeln für TCP-Port 445 suchen	`[RO] Get-NetFirewallPortFilter | Where-Object { $_.Protocol -eq "TCP" -and $_.LocalPort -eq 445 } | Get-NetFirewallRule`
Adressfilter einer Regel anzeigen	`[RO] Get-NetFirewallRule -DisplayName "REGELNAME" | Get-NetFirewallAddressFilter`
Portfilter einer Regel anzeigen	`[RO] Get-NetFirewallRule -DisplayName "REGELNAME" | Get-NetFirewallPortFilter`
Anwendungsfilter einer Regel anzeigen	`[RO] Get-NetFirewallRule -DisplayName "REGELNAME" | Get-NetFirewallApplicationFilter`

Warum zeigt `Get-NetFirewallRule` nicht alle Ports und Adressen direkt an?

Windows speichert Bedingungen wie Ports, Programme und Adressen in zugeordneten Filterobjekten. Deshalb werden für eine vollständige Analyse zusätzlich folgende Cmdlets verwendet:

- `Get-NetFirewallPortFilter`
- `Get-NetFirewallAddressFilter`
- `Get-NetFirewallApplicationFilter`
- `Get-NetFirewallServiceFilter`
- `Get-NetFirewallInterfaceFilter`

Sinnvolle Prüfreihenfolge

1. Aktives Netzwerkprofil mit `Get-NetConnectionProfile` feststellen.
2. Firewallstatus des Profils mit `Get-NetFirewallProfile` prüfen.
3. Prüfen, ob der Dienst lokal auf dem erwarteten Port lauscht.
4. Passende eingehende oder ausgehende Regel suchen.
5. Port-, Adress-, Programm- und Profilfilter der Regel prüfen.
6. Verbindung von einem autorisierten Testsystem aus testen.

> Die Firewall nicht pauschal deaktivieren. Dadurch verändert sich die Sicherheitslage und das Testergebnis bildet die ursprüngliche Konfiguration nicht mehr korrekt ab.

10. Diagnoseausgaben filtern, vergleichen und exportieren

PowerShell-Ausgaben bestehen aus Objekten. Die Filterung sollte deshalb möglichst vor der Formatierung erfolgen.

Aufgabe	PowerShell-Befehl
Bestimmte Eigenschaften auswählen	`[RO] Get-NetAdapter | Select-Object Name, Status, LinkSpeed`
Objekte filtern	`[RO] Get-NetAdapter | Where-Object Status -ne "Up"`
Ergebnisse sortieren	`[RO] Get-NetRoute | Sort-Object RouteMetric`
Ergebnisse gruppieren	`[RO] Get-NetTCPConnection | Group-Object State`
Anzahl ermitteln	`[RO] (Get-NetTCPConnection -State Established).Count`
Tabellenansicht erzeugen	`[RO] Get-NetAdapter | Format-Table -AutoSize`
Detailansicht erzeugen	`[RO] Get-NetIPConfiguration | Format-List *`
CSV-Datei exportieren	`[RO][FILE][SENS] Get-NetTCPConnection | Export-Csv -Path ".\tcp-connections.csv" -NoTypeInformation -Encoding UTF8`
Objekte vollständig speichern	`[RO][FILE][SENS] Get-NetIPConfiguration -All | Export-Clixml -Path ".\ip-configuration.xml"`
Textprotokoll erzeugen	`[RO][FILE][SENS] Get-NetIPConfiguration -All | Format-List * | Out-File ".\ip-configuration.txt" -Encoding utf8`

Zwei Messzeitpunkte vergleichen

```
$Vorher = Get-NetTCPConnection
Start-Sleep -Seconds 10
$Nachher = Get-NetTCPConnection

Compare-Object $Vorher $Nachher -Property State, LocalAddress, LocalPort, RemoteAddress, RemotePort
```

Hinweise

- `Where-Object` und `Select-Object` verarbeiten Objekte.
- `Format-Table` und `Format-List` sind für die Darstellung am Ende einer Pipeline gedacht.
- Für eine spätere Weiterverarbeitung sind `Export-Csv` oder `Export-Clixml` geeigneter als formatierter Text.
- Exportdateien können IP-Adressen, DNS-Namen, Prozesskennungen, Verbindungen und interne Netzstrukturen enthalten.
- Diagnoseexporte müssen entsprechend den betrieblichen Datenschutz- und Sicherheitsvorgaben behandelt werden.

11. Eingreifende Befehle nur nach der Diagnose verwenden

Die folgenden Befehle verändern einen Zustand und gehören nicht zur rein lesenden Bestandsaufnahme.

Maßnahme	PowerShell-Befehl	Auswirkung
DNS-Clientcache leeren	`[CHANGE][PRIV] Clear-DnsClientCache`	Lokal gespeicherte DNS-Antworten werden entfernt
Netzwerkadapter neu starten	`[CHANGE][PRIV][DISRUPT] Restart-NetAdapter -Name "Ethernet"`	Adapter wird deaktiviert und erneut aktiviert
DHCP-Lease erneuern	`[CHANGE][PRIV][DISRUPT] ipconfig /release; ipconfig /renew`	Vorhandene DHCP-Adresse wird zunächst freigegeben
DNS-Registrierung anfordern	`[CHANGE][PRIV] Register-DnsClient`	Dynamische DNS-Registrierung wird angestoßen

Vor einem Eingriff prüfen

- Besteht eine lokale oder entfernte Administrationsverbindung?
- Wird die eigene Remoteverbindung durch den Eingriff getrennt?
- Ist die aktuelle Konfiguration dokumentiert?
- Ist die Ursache ausreichend eingegrenzt?
- Gibt es einen Rückfallweg oder lokalen Zugriff?
- Sind Auswirkungen auf Benutzer, Dienste, VPN-Verbindungen oder Cluster bekannt?
- Ist ein Wartungsfenster beziehungsweise eine Freigabe erforderlich?

> “ `Restart-NetAdapter` kann eine Remoteverbindung sofort unterbrechen. Der Befehl darf auf entfernten Produktivsystemen nur mit abgesichertem Rückfallweg verwendet werden.

---

12. Praktische PowerShell-Diagnosereihenfolge

Schritt 1 – Adapter prüfen

```
Get-NetAdapter |
    Select-Object Name, Status, LinkSpeed, MacAddress, ifIndex
```

Schritt 2 – IP-Konfiguration prüfen

```
Get-NetIPConfiguration -All
```

Schritt 3 – Standardroute und DNS-Server prüfen

```
Get-NetRoute -DestinationPrefix "0.0.0.0/0"
Get-DnsClientServerAddress
```

Schritt 4 – Lokales Standardgateway testen

```
Test-NetConnection 192.0.2.1 -InformationLevel Detailed
```

Schritt 5 – Externe IP-Adresse testen

```
Test-NetConnection 1.1.1.1 -InformationLevel Detailed
```

Schritt 6 – DNS-Auflösung testen

```
Resolve-DnsName example.com -DnsOnly
```

Schritt 7 – Zielport testen

```
Test-NetConnection example.com -Port 443 -InformationLevel Detailed
```

Schritt 8 – Route zum Ziel prüfen

```
Find-NetRoute -RemoteIPAddress 1.1.1.1
Test-NetConnection example.com -TraceRoute
```

Schritt 9 – Lokale Ports und Prozesse prüfen

```
Get-NetTCPConnection -State Listen |
    Sort-Object LocalPort
```

Schritt 10 – Firewallstatus prüfen

```
Get-NetFirewallProfile
Get-NetFirewallRule -Enabled True |
    Select-Object DisplayName, Direction, Action, Profile
```

Diagnoselogik

Ergebnis	Nächster Schwerpunkt
Adapter nicht aktiv	Kabel, WLAN, Adapter, Treiber oder Port prüfen
Keine passende IP-Adresse	DHCP oder statische IP-Konfiguration prüfen
Gateway nicht erreichbar	Lokales Netz, VLAN, WLAN, Switch oder Gateway prüfen
Externe IP erreichbar, DNS-Name nicht	DNS-Konfiguration und Resolver prüfen
Ziel-IP erreichbar, Zielport nicht	Dienst, Portbindung und Firewall prüfen
Falscher Adapter oder falsche Quelladresse	Routing, Metrik, VPN und mehrere Adapter prüfen
Dienst lauscht lokal, ist extern aber nicht erreichbar	Firewall, NAT, Routing und Dienstbindung prüfen

---

Kurzreferenz

Diagnoseziel	Befehl
Adapterstatus	`[RO] Get-NetAdapter`
Adapterstatistik	`[RO] Get-NetAdapterStatistics`
IP-Gesamtübersicht	`[RO] Get-NetIPConfiguration -All`
IP-Adressen	`[RO] Get-NetIPAddress`
Schnittstellen und Metriken	`[RO] Get-NetIPInterface`
Netzwerkprofil	`[RO] Get-NetConnectionProfile`
DNS-Server	`[RO] Get-DnsClientServerAddress`
DNS-Auflösung	`[TEST] Resolve-DnsName example.com -DnsOnly`
DNS-Cache	`[RO] Get-DnsClientCache`
Verbindungstest	`[TEST] Test-NetConnection example.com`
TCP-Porttest	`[TEST] Test-NetConnection example.com -Port 443`
Routenverfolgung	`[TEST] Test-NetConnection example.com -TraceRoute`
Routingtabelle	`[RO] Get-NetRoute`
Route zu einem Ziel	`[RO] Find-NetRoute -RemoteIPAddress 1.1.1.1`
Nachbartabelle	`[RO] Get-NetNeighbor`

Diagnoseziel	Befehl
TCP-Verbindungen	`[RO] Get-NetTCPConnection`
UDP-Endpunkte	`[RO] Get-NetUDPEndpoint`
Firewallprofile	`[RO] Get-NetFirewallProfile`
Firewallregeln	`[RO] Get-NetFirewallRule`

Merksatz

> PowerShell-Netzwerkdiagnose bedeutet nicht, möglichst viele Befehle auszuführen. Entscheidend ist, die Ergebnisse als zusammenhängende Beweiskette auszuwerten: Adapter → IP-Konfiguration → Gateway → Route → DNS → Zielport → lokaler Dienst → Firewall.

Quellen

- Microsoft Learn – NetAdapter-Modul
- Microsoft Learn – NetTCPIP-Modul
- Microsoft Learn – DnsClient-Modul
- Microsoft Learn – NetSecurity-Modul
- Microsoft Learn – Test-NetConnection
- Microsoft Learn – Resolve-DnsName
- Microsoft Learn – Windows-Firewall über die Befehlszeile verwalten

---