9.4 Lokale Host-Firewall prüfen

Ziel dieser Seite

Diese Seite beschreibt die systematische Diagnose einer lokalen Host-Firewall unter Windows, Linux und macOS sowie ihre Abgrenzung von:

- fehlenden Listenern;
- falschen Bindungsadressen;
- Netzwerkfirewalls;
- Access Control Lists;
- NAT und Portweiterleitungen;
- Proxys und Load Balancern;
- Container- und Kubernetes-Regeln;
- VPN- und Endpoint-Security-Filtern;
- Anwendungsfehlern.

Nach der Bearbeitung muss nachvollziehbar geprüft werden können:

- welche lokale Firewalltechnik tatsächlich aktiv ist;
- welches Netzwerkprofil, welche Zone oder welche Schnittstelle gilt;
- welche eingehende und ausgehende Standardaktion verwendet wird;
- welche Regel den betroffenen Datenverkehr erlaubt oder blockiert;
- aus welcher Richtlinienquelle die wirksame Regel stammt;
- ob TCP, UDP, ICMPv4 und ICMPv6 getrennt berücksichtigt wurden;
- ob ein Paket den Host erreicht;
- ob es durch die Host-Firewall verworfen wird;
- ob ein Prozess oder Dienst trotz passender Regel nicht lauscht;
- ob Container-, Hypervisor- oder Endpoint-Security-Filter beteiligt sind;
- wie eine Änderung sicher vorbereitet und verifiziert wird.

Eine Host-Firewall darf nicht vorsorglich vollständig deaktiviert werden. Zuerst müssen der betroffene Datenfluss und die tatsächlich wirksame Regel nachgewiesen werden.

Sicherheits- und Wirkungsklassen

Kennzeichnung	Bedeutung
LESEND	Erfasst ausschließlich vorhandene Zustände.
NETZAKTIV	Erzeugt Netzwerkverkehr zum geprüften Ziel.
SENSITIV	Kann interne Regeln, Adressen, Programme oder Kommunikationsbeziehungen sichtbar machen.
ÄNDERND	Verändert Firewallkonfiguration oder Protokollierung.

Kennzeichnung	Bedeutung
AUSFALLRISIKO	Kann bestehende Verbindungen, Verwaltung oder Dienste beeinträchtigen.

Firewallregeln und Protokolle können sicherheitsrelevante Informationen enthalten:

- interne Netze;
- Verwaltungsports;
- erlaubte Quelladressen;
- Anwendungspfade;
- Dienstnamen;
- Benutzer- und Gruppenbezüge;
- VPN-Schnittstellen;
- Sicherheitsprodukte;
- Cloud- und Containerbereiche.

Diese Informationen müssen geschützt gespeichert und weitergegeben werden.

Aufgabe einer lokalen Host-Firewall

Eine lokale Host-Firewall filtert Netzwerkverkehr direkt auf einem Endgerät oder Server.

Abhängig vom Betriebssystem kann sie prüfen:

- eingehenden Verkehr zum lokalen Host;
- ausgehenden Verkehr des lokalen Hosts;
- weitergeleiteten Verkehr;
- Transportprotokoll;
- lokale und entfernte Adressen;
- lokale und entfernte Ports;
- Netzwerkprofil oder Zone;
- Schnittstelle;
- Programm;
- Dienst;
- Benutzer;
- Paket- oder Verbindungszustand;
- IPsec-Authentifizierung;
- Container- oder VM-Netzwerkpfade.

Eine Host-Firewall ersetzt keine Netzwerkfirewall. Beide Ebenen können gleichzeitig filtern.

Datenpfad vereinfachen

Eingehender Datenverkehr:

```
Client
→ Netzwerkfirewall
→ Servernetzwerkschnittstelle
→ lokale Host-Firewall
→ lokaler Socket
→ Anwendung
```

Ausgehender Datenverkehr:

```
Anwendung
→ lokaler Socket
→ lokale Host-Firewall
→ Servernetzwerkschnittstelle
→ Netzwerkfirewall
→ Ziel
```

Container- oder VM-Verkehr kann zusätzlich einen Weiterleitungspfad verwenden:

```
Client
→ Hostnetzwerkschnittstelle
→ Host-Firewall oder NAT
→ virtuelle Bridge
→ Container- oder VM-Firewall
→ Anwendung
```

Jede Ebene muss getrennt nachgewiesen werden.

Host-Firewall, Listener und Anwendung unterscheiden

Befund	Aussage
Dienst läuft	Prozessstatus ist vorhanden
Listener vorhanden	Socket wurde gebunden
lokale Verbindung funktioniert	lokaler Netzwerkpfad funktioniert
entfernte Verbindung funktioniert	der getestete Ende-zu-Ende-Pfad funktioniert
Firewallregel vorhanden	eine konfigurierte Regel existiert
Firewallregel wirksam	Regel gilt für den konkreten Datenfluss
Paket wird geloggt	Paket wurde an einem bestimmten Filterpunkt erfasst

Befund	Aussage
Anwendung antwortet	Transport und Anwendung funktionieren für den Test

Eine vorhandene Allow-Regel beweist nicht, dass:

- das richtige Profil aktiv ist;
- die Regel aktiviert ist;
- Protokoll und Port stimmen;
- die Quelladresse zum Regelbereich gehört;
- das richtige Programm verwendet wird;
- keine Blockregel Vorrang hat;
- eine zentral verwaltete Richtlinie die Regel überschreibt;
- der Dienst tatsächlich lauscht.

Prüfdaten des betroffenen Flows

Vor der Regelsuche muss der Datenfluss exakt dokumentiert werden.

```
Richtung:
<eingehend oder ausgehend>

Transportprotokoll:
<TCP, UDP, ICMPv4, ICMPv6 oder anderes>

Quelladresse:
<IP-Adresse oder Netz>

Quellport:
<Port oder dynamischer Bereich>

Zieladresse:
<IP-Adresse>

Zielport:
<Port>

Lokales Programm:
<vollständiger Pfad>

Lokaler Dienst:
```

```
<Dienstname>

Schnittstelle:
<Ethernet, WLAN, VPN, Loopback oder virtuell>

Netzwerkprofil oder Zone:
<Domain, Private, Public oder Linux-Zone>

Zeitpunkt:
<Datum, Uhrzeit und Zeitzone>
```

Ohne diese Angaben kann nicht sicher entschieden werden, ob eine Firewallregel zum Fehler passt.

Eingehend, ausgehend und weitergeleitet

Richtung	Bedeutung
eingehend	Verkehr ist an den lokalen Host adressiert
ausgehend	Verkehr wird durch einen lokalen Prozess erzeugt
weitergeleitet	Verkehr durchquert den Host zu einem anderen Endpunkt

Unter Linux entsprechen diese Pfade häufig den Netfilter-Hooks beziehungsweise Basischains:

```
input
output
forward
```

Ein Paket für einen Container oder eine virtuelle Maschine kann den Hostpfad `forward` verwenden, obwohl der Client den Hostport anspricht.

Eine Regel im lokalen `input`-Pfad muss deshalb nicht für weitergeleiteten Containerverkehr gelten.

Loopback ist kein vollständiger Firewalltest

Ein Test gegen:

```
127.0.0.1
::1
localhost
```

prüft den Loopbackpfad.

Dieser Test bildet nicht zwingend ab:

- die physische Netzwerkschnittstelle;
- das aktive Firewallprofil;
- die externe Quelladresse;
- eine VPN-Schnittstelle;
- NAT oder Portweiterleitung;
- eine Netzwerkfirewall;
- den Container-Weiterleitungspfad.

Auch ein Test gegen die eigene LAN-Adresse vom selben Host kann lokal geroutet werden und muss nicht denselben Pfad wie ein entfernter Client verwenden.

Für eine vollständige Prüfung ist ein repräsentativer externer Client erforderlich.

Firewallzustände richtig interpretieren

Aktion	Typisches Verhalten
Allow oder Accept	passender Verkehr darf den Filterpunkt passieren
Drop	Paket wird ohne aktive Fehlermeldung verworfen
Reject	Paket wird verworfen und es kann eine Ablehnung zurückgesendet werden
Log	Ereignis wird protokolliert; allein keine Allow- oder Blockaktion
Default Deny	nicht ausdrücklich erlaubter Verkehr wird blockiert
Default Allow	nicht ausdrücklich blockierter Verkehr wird erlaubt

Ein TCP-Reset kann entstehen durch:

- geschlossenen TCP-Port;
- Anwendung;
- lokale Firewall mit Reject-Verhalten;
- Netzwerkfirewall;
- Proxy;
- Load Balancer.

Ein Timeout kann durch stilles Drop-Verhalten entstehen, beweist aber nicht automatisch eine Firewallblockierung.

Wirksame Richtlinie statt nur Konfigurationsdatei prüfen

Firewallregeln können stammen aus:

- lokaler Konfiguration;
- Gruppenrichtlinie;
- MDM;
- Sicherheitsbaseline;
- Endpoint-Security-Software;
- Cloudmanagement;
- Containerplattform;
- Hypervisor;
- VPN-Client;
- Service Manager;
- temporärer Laufzeitkonfiguration.

Entscheidend ist die tatsächlich aktive Richtlinie.

Eine Konfigurationsdatei oder GUI kann vom wirksamen Kernelzustand abweichen, wenn:

- Änderungen noch nicht geladen wurden;
- Laufzeit- und permanente Konfiguration unterschiedlich sind;
- eine zentrale Richtlinie Vorrang besitzt;
- ein anderer Firewallmanager verwendet wird;
- Containerwerkzeuge eigene Regeln erzeugt haben;
- ein Sicherheitsprodukt zusätzliche Filter installiert hat.

Windows-Firewallprofile

Windows verwendet drei Firewallprofile:

Profil	Typische Verwendung
`Domain`	Netzwerk mit erkannter und authentifizierter Active-Directory-Domäne
`Private`	als vertrauenswürdig eingestuftes privates Netzwerk
`Public`	nicht vertrauenswürdiges oder öffentliches Netzwerk

Regeln können für ein oder mehrere Profile gelten.

Ein häufiger Fehler ist:

```
Allow-Regel gilt nur für Domain oder Private.
Die aktive Schnittstelle verwendet jedoch Public.
```

Windows kann unterschiedliche Profile gleichzeitig für unterschiedliche Schnittstellen verwenden.

Windows: aktive Netzwerkprofile prüfen

LESEND

```
Get-NetConnectionProfile |
  Select-Object `
    InterfaceAlias,
    InterfaceIndex,
    Name,
    NetworkCategory,
    IPv4Connectivity,
    IPv6Connectivity
```

Zu prüfen sind:

- richtige Schnittstelle;
- aktive Netzwerkkategorie;
- VPN-Adapter;
- virtuelle Adapter;
- unerwartetes `Public`-Profil;
- mehrere gleichzeitig aktive Profile;
- IPv4- und IPv6-Konnektivität.

Das Domainprofil darf nicht manuell als Ersatz für eine fehlerhafte Domänenerkennung erzwungen werden. Zuerst müssen DNS, Erreichbarkeit, Authentifizierung und Netzwerkidentifikation geprüft werden.

Windows: Firewallprofile prüfen

LESEND

```
Get-NetFirewallProfile |
  Select-Object `
    Name,
    Enabled,
    DefaultInboundAction,
    DefaultOutboundAction,
    AllowInboundRules,
    AllowLocalFirewallRules,
    AllowLocalIPsecRules,
    NotifyOnListen,
```

```
  LogFileName,
  LogMaxSizeKilobytes,
  LogAllowed,
  LogBlocked
```

Wirksame Profile aus dem aktiven Richtlinienspeicher:

```
Get-NetFirewallProfile `
  -PolicyStore ActiveStore |
  Format-List *
```

Zu dokumentieren sind:

- ist das Profil aktiviert?
- welche Standardaktion gilt eingehend?
- welche Standardaktion gilt ausgehend?
- sind lokale Regeln zulässig?
- sind lokal konfigurierte IPsec-Regeln zulässig?
- werden blockierte Pakete protokolliert?
- wo liegt die Protokolldatei?
- welche maximale Größe besitzt sie?

Die Standardaktionen dürfen nicht ungeprüft als `Block` beziehungsweise `Allow` angenommen werden.

Windows: aktive Regeln erfassen

LESEND

Alle wirksamen Regeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore
```

Aktivierte Regeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Enabled True
```

Aktivierte eingehende Allow-Regeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Enabled True `
  -Direction Inbound `
  -Action Allow
```

Aktivierte eingehende Blockregeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Enabled True `
  -Direction Inbound `
  -Action Block
```

Aktivierte ausgehende Blockregeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Enabled True `
  -Direction Outbound `
  -Action Block
```

Herkunft der Regeln:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Enabled True |
  Select-Object `
    Name,
    DisplayName,
    Direction,
    Action,
    Profile,
    PolicyStoreSourceType,
    PolicyStoreSource
```

`ActiveStore` stellt die zusammengeführte aktive Richtlinie der auf das System wirkenden Policy Stores dar. Eine Abfrage ohne `-PolicyStore ActiveStore` kann eine andere Sicht liefern.

Windows: vollständige Regeldarstellung

LESEND

```
Show-NetFirewallRule `
  -PolicyStore ActiveStore
```

Für eine bekannte Regel:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Format-List *
```

Die eigentlichen Port-, Adress-, Programm- und Dienstbedingungen werden in zugeordneten Filterobjekten verwaltet.

Windows: Portfilter prüfen

LESEND

Alle Portfilter:

```
Get-NetFirewallPortFilter `
  -PolicyStore ActiveStore
```

Regeln für lokalen Port `443` suchen:

```
Get-NetFirewallPortFilter `
  -PolicyStore ActiveStore |
  Where-Object LocalPort -eq "443" |
  Get-NetFirewallRule |
  Select-Object `
    Name,
    DisplayName,
    Enabled,
    Direction,
    Action,
    Profile,
```

```
    PolicyStoreSourceType,
    PolicyStoreSource
```

Regeln für entfernten Port `443`:

```
Get-NetFirewallPortFilter `
  -PolicyStore ActiveStore |
  Where-Object RemotePort -eq "443" |
  Get-NetFirewallRule |
  Select-Object `
    Name,
    DisplayName,
    Enabled,
    Direction,
    Action,
    Profile
```

Danach muss der vollständige Portfilter der gefundenen Regel geprüft werden:

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Get-NetFirewallPortFilter |
  Format-List *
```

Zu prüfen sind:

- TCP oder UDP;
- lokaler Port;
- entfernter Port;
- ICMP-Typ;
- dynamische Schlüsselwörter;
- `Any` oder konkreter Wert.

Windows: Adressfilter prüfen

LESEND

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Get-NetFirewallAddressFilter |
  Format-List *
```

Zu prüfen sind:

- `LocalAddress`;
- `RemoteAddress`;
- einzelne Hosts;
- Subnetze;
- `Any`;
- lokale Subnetze;
- IPv4 oder IPv6;
- dynamische oder richtlinienbasierte Bereiche.

Eine Allow-Regel für:

```
RemoteAddress = 192.0.2.0/24
```

gilt nicht für einen Client aus:

```
198.51.100.0/24
```

Windows: Programmfilter prüfen

LESEND

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Get-NetFirewallApplicationFilter |
  Format-List *
```

Zu prüfen sind:

- vollständiger Programmpfad;
- aktuell gestartete ausführbare Datei;
- geänderter Installationspfad;

- neue Programmversion;
- 32-Bit- oder 64-Bit-Pfad;
- symbolische Verknüpfungen;
- Launcher und eigentlicher Serverprozess;
- `Any` oder konkretes Programm.

Eine Regel für einen alten Anwendungspfad gilt nicht automatisch für eine neue Programmdatei.

Windows: Dienstfilter prüfen

LESEND

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Get-NetFirewallServiceFilter |
  Format-List *
```

Zu prüfen sind:

- konkreter Windows-Dienst;
- Dienstname statt Anzeigename;
- gemeinsam verwendeter Prozess;
- Service Hardening;
- `Any` oder konkreter Dienst.

Eine Regel kann für einen Dienst gelten, obwohl mehrere Dienste denselben Prozess wie `svchost.exe` verwenden.

Windows: Schnittstellenfilter prüfen

LESEND

```
Get-NetFirewallRule `
  -PolicyStore ActiveStore `
  -Name "<Regelname>" |
  Get-NetFirewallInterfaceFilter |
  Format-List *
```

Zu prüfen sind:

- konkrete Schnittstelle;

- Schnittstellentyp;
- LAN;
- WLAN;
- Remote Access;
- VPN;
- virtuelle Schnittstelle.

Eine Regel für LAN muss nicht für einen VPN- oder WLAN-Pfad gelten.

Windows: Regelbedingungen vollständig dokumentieren

Für eine gefundene Regel müssen mindestens folgende Informationen zusammengeführt werden:

```
Name:
<Regelname>

Status:
<aktiviert oder deaktiviert>

Richtung:
<eingehend oder ausgehend>

Aktion:
<Allow oder Block>

Profile:
<Domain, Private, Public>

Protokoll:
<TCP, UDP, ICMPv4 oder ICMPv6>

Lokaler Port:
<Port>

Entfernter Port:
<Port>

Lokale Adresse:
<Adresse oder Netz>
```

```
Entfernte Adresse:
<Adresse oder Netz>

Programm:
<Pfad>

Dienst:
<Dienstname>

Schnittstelle:
<Filter>

Richtlinienquelle:
<Local, Group Policy, MDM oder andere>
```

Erst der Vergleich aller Bedingungen mit dem konkreten Datenfluss zeigt, ob die Regel tatsächlich passt.

Windows-Regelvorrang

Für Windows-Firewallregeln gilt unter anderem:

- eine ausdrücklich definierte Allow-Regel kann die eingehende Standardblockierung übersteuern;
- eine ausdrücklich definierte Blockregel besitzt Vorrang vor einer widersprechenden Allow-Regel;
- spezifischere Regeln können gegenüber weniger spezifischen Regeln maßgeblich sein;
- zentrale Richtlinien können lokale Konfigurationsmöglichkeiten begrenzen;
- Service-Hardening- und IPsec-Anforderungen können zusätzlich wirken.

Windows-Firewallregeln dürfen nicht wie eine einfache von oben nach unten abgearbeitete ACL-Liste interpretiert werden.

Windows: Gruppenrichtlinienbezug prüfen

LESEND

Zusammenfassung der angewendeten Computerrichtlinien:

```
gpresult /scope computer /r
```

Wirksame Firewallregeln enthalten zusätzlich:

```
PolicyStoreSourceType
PolicyStoreSource
```

Zu prüfen sind:

- stammt die Regel aus lokaler Konfiguration?
- stammt sie aus einer Domänenrichtlinie?
- dürfen lokale Firewallregeln zusammengeführt werden?
- wurde die erwartete GPO angewendet?
- existiert eine Blockregel aus einer anderen Richtlinie?
- wird die Einstellung durch MDM verwaltet?
- erscheint eine lokale Änderung nach Richtlinienaktualisierung erneut oder verschwindet sie?

Eine zentral verwaltete Regel darf nicht durch eine lokale Parallelregel umgangen werden.

Windows: Firewallprotokoll prüfen

Der Protokollpfad kann je Profil abweichen und muss über `Get-NetFirewallProfile` bestimmt werden.

Typischer Standardpfad:

```
%SystemRoot%\System32\LogFiles\Firewall\pfirewall.log
```

Letzte Einträge:

LESEND · SENSITIV

```
Get-Content `
  "$env:SystemRoot\System32\LogFiles\Firewall\pfirewall.log" `
  -Tail 100
```

Nach Adresse oder Port suchen:

```
Get-Content `
  "$env:SystemRoot\System32\LogFiles\Firewall\pfirewall.log" `
  -Tail 1000 |
  Select-String "192.0.2.100|443"
```

Zu korrelieren sind:

- Aktion;
- Datum und Uhrzeit;
- Protokoll;
- Quell- und Zieladresse;
- Quell- und Zielport;
- Richtung beziehungsweise Pfad;
- betroffene Schnittstelle;
- identischer Testzeitpunkt.

Fehlende Einträge beweisen nicht, dass die Firewall nicht beteiligt ist. Protokollierung kann deaktiviert, begrenzt oder an einem anderen Pfad konfiguriert sein.

Windows: temporäre Protokollierung

Vor einer Änderung müssen die vorhandenen Profilwerte dokumentiert werden.

Aktuellen Zustand sichern:

```
Get-NetFirewallProfile |
  Select-Object `
    Name,
    LogFileName,
    LogMaxSizeKilobytes,
    LogAllowed,
    LogBlocked
```

Blockprotokollierung für ein bestimmtes Profil aktivieren:

ÄNDERND · SENSITIV

```
Set-NetFirewallProfile `
  -Profile Domain `
  -LogBlocked True
```

Die Änderung muss:

- autorisiert sein;
- auf das betroffene Profil begrenzt werden;
- hinsichtlich Speicherbedarf überwacht werden;
- nach dem Test auf den dokumentierten Ausgangswert zurückgesetzt werden.

Das Aktivieren der Protokollierung für erlaubte Verbindungen kann sehr große Datenmengen erzeugen.

Windows Filtering Platform prüfen

Windows Filtering Platform, kurz WFP, bildet die Filterplattform für Windows-Firewall, IPsec und weitere Filterkomponenten.

Relevante Security-Ereignisse können sein:

Ereignis-ID	Bedeutung
`5152`	WFP hat ein Paket blockiert.
`5154`	Anwendung oder Dienst durfte auf eingehende Verbindungen lauschen.
`5155`	Anwendung oder Dienst wurde am Lauschen gehindert.
`5156`	WFP hat eine Verbindung erlaubt.
`5157`	WFP hat eine Verbindung blockiert.
`5158`	Bindung an einen lokalen Port wurde erlaubt.
`5159`	Bindung an einen lokalen Port wurde blockiert.

Blockereignisse lesen:

LESEND · SENSITIV

```
Get-WinEvent `
  -FilterHashtable @{
    LogName = "Security"
    Id      = 5152, 5155, 5157, 5159
  } `
  -MaxEvents 100
```

Diese Ereignisse stehen nur zur Verfügung, wenn die entsprechenden Überwachungsrichtlinien aktiv waren.

Zu prüfen sind:

- Application Name;
- Process ID;
- Direction;
- Source Address;
- Source Port;

- Destination Address;
- Destination Port;
- Protocol;
- Filter Run-Time ID;
- Layer Name;
- Zeitpunkt.

Sehr umfangreiche WFP-Überwachung kann viele Ereignisse erzeugen und muss gezielt eingesetzt werden.

Windows: netsh als ergänzende Sicht

LESEND

Profile:

```
netsh advfirewall show allprofiles
```

Regeln:

```
netsh advfirewall firewall show rule name=all verbose
```

Überwachungszustand:

```
netsh advfirewall monitor show firewall
```

Der ältere Kontext:

```
netsh firewall
```

sollte nicht mehr für die aktuelle Verwaltung verwendet werden. Für moderne Systeme sind PowerShell-NetSecurity-Cmdlets und `netsh advfirewall` vorgesehen.

Windows: Drittanbieterfilter berücksichtigen

Verkehr kann blockiert werden, obwohl keine passende Windows-Firewall-Blockregel sichtbar ist.

Mögliche zusätzliche Filter:

- Endpoint Detection and Response;
- Antiviren-Netzwerkfilter;

- Data Loss Prevention;
- VPN-Client;
- Webfilter;
- Network-Extension- oder WFP-Callout-Treiber;
- Zero-Trust-Agent;
- Hypervisorfilter;
- Cloud-Sicherheitsagent.

Hinweise:

- Fehler beginnt nach Installation oder Update eines Sicherheitsprodukts;
- WFP-Ereignis verweist auf einen fremden Filter;
- Windows-Firewallprotokoll zeigt keinen Drop;
- Paket erreicht die Schnittstelle, aber nicht die Anwendung;
- nur ein bestimmter Prozess ist betroffen;
- eine andere Anwendung auf demselben Portpfad funktioniert.

Sicherheitssoftware darf nicht ohne Freigabe beendet oder deinstalliert werden.

Windows: Hyper-V, WSL und Container

Verkehr zu einer VM, WSL-Instanz oder einem Container kann zusätzliche Filterebenen durchlaufen.

Zu unterscheiden sind:

- Windows-Host-Firewall;
- Hyper-V-Firewall;
- virtueller Switch;
- Host Network Service;
- Container-NAT;
- Gastbetriebssystem-Firewall;
- Kubernetes- oder CNI-Regeln.

Verfügbare Hyper-V-Firewall-Cmdlets prüfen:

```
Get-Command `
  -Name "*NetFirewallHyperV*" `
  -ErrorAction SilentlyContinue
```

Wenn entsprechende Cmdlets vorhanden sind, müssen Hyper-V-Profile und Regeln zusätzlich geprüft werden.

Eine Allow-Regel im normalen Hostprofil beweist nicht automatisch, dass Verkehr zu einer VM oder WSL-Instanz erlaubt wird.

Linux: verwendetes Firewall-Backend bestimmen

Auf Linux-Systemen können mehrere Werkzeuge vorhanden sein:

- nftables;
- iptables-legacy;
- iptables-nft;
- firewalld;
- UFW;
- Container- oder Kuberneteskomponenten;
- eBPF-, XDP- oder `tc`-Filter.

Zuerst muss bestimmt werden, welches System die aktive Richtlinie verwaltet.

Versionen prüfen:

LESEND

```
nft --version
```

```
iptables --version
```

Eine iptables-Ausgabe kann auf ein Backend hinweisen:

```
iptables vX.Y.Z (nf_tables)
```

oder:

```
iptables vX.Y.Z (legacy)
```

Aktive Manager:

```
systemctl is-active firewalld
```

```
systemctl is-active ufw
```

Die Installation oder Aktivität eines Managers beweist nicht, dass keine weiteren Regeln existieren.

Linux: nftables-Regelsatz prüfen

LESEND · SENSITIV

```
sudo nft list ruleset
```

Mit Regel-Handles:

```
sudo nft -a list ruleset
```

Zu prüfen sind:

- Tabellenfamilie;
- Tabellenname;
- Basischains;
- Hook;
- Priorität;
- Chain Policy;
- Sprünge in weitere Chains;
- Regeln;
- Sets;
- Maps;
- Protokoll;
- Adressen;
- Ports;
- Schnittstellen;
- Verbindungszustände;
- Counter;
- Log-Aktionen;
- `accept`, `drop` oder `reject`.

Wichtige Tabellenfamilien:

Familie	Bedeutung
`inet`	gemeinsame Verarbeitung von IPv4 und IPv6
`ip`	IPv4
`ip6`	IPv6
`bridge`	Bridgeverkehr
`arp`	ARP
`netdev`	früher Paketpfad an Netzgeräten

Eine Regel in `inet` kann sowohl IPv4 als auch IPv6 betreffen. Separate `ip`- und `ip6`-Regeln müssen getrennt geprüft werden.

Linux: nftables-Hooks prüfen

Typische Hooks:

Hook	Datenpfad
`input`	Verkehr zum lokalen Host
`output`	lokal erzeugter Verkehr
`forward`	weitergeleiteter Verkehr
`prerouting`	vor der Routingentscheidung
`postrouting`	nach der Routingentscheidung
`ingress`	früher Eingangspfad

Der Name einer Chain muss nicht ihrem Hook entsprechen. Entscheidend ist die tatsächliche Chain-Definition.

Beispiel:

```
chain host_in {
    type filter hook input priority filter;
    policy drop;
}
```

Eine frei benannte Chain ohne Hook wirkt nur, wenn eine andere Regel in sie springt.

Linux: nftables-Counter auswerten

Regeln können Zähler für Pakete und Bytes enthalten.

Beispielausgabe:

```
counter packets 42 bytes 3360 drop
```

Für einen kontrollierten Test:

1. aktuellen Zählerstand dokumentieren;
2. genau einen Test ausführen;
3. Regelsatz erneut lesen;
4. Zählerdifferenz prüfen;
5. Quelladresse, Zielport und Protokoll abgleichen.

Ein unveränderter Zähler beweist nicht automatisch, dass die Regel nicht betroffen ist:

- die Regel besitzt eventuell keinen Counter;
- eine frühere Regel beendet die Verarbeitung;
- der Datenfluss verwendet eine andere Chain;
- die Aufzeichnung erfolgt in einem anderen Namespace;
- ein XDP- oder `tc`-Filter verwirft früher.

Zähler dürfen nicht vorsorglich zurückgesetzt werden, weil dadurch Beweisdaten verloren gehen.

Linux: iptables-Regeln prüfen

LESEND · SENSITIV

IPv4:

```
sudo iptables \
  -L \
  -n \
  -v \
  --line-numbers
```

IPv6:

```
sudo ip6tables \
  -L \
  -n \
  -v \
  --line-numbers
```

Vollständiger IPv4-Regelsatz mit Countern:

```
sudo iptables-save \
  -c
```

Vollständiger IPv6-Regelsatz:

```
sudo ip6tables-save \
  -c
```

Wichtige Optionen:

Option	Bedeutung
`-L`	Chains und Regeln anzeigen
`-n`	numerische Adressen und Ports
`-v`	ausführliche Anzeige und Counter
`--line-numbers`	Regelpositionen anzeigen
`-c` bei `iptables-save`	Counter mit ausgeben

`iptables -L` allein bildet nicht immer alle Tabellen und benutzerdefinierten Zusammenhänge ausreichend ab. `iptables-save` liefert eine vollständigere Regelsatzdarstellung.

Linux: INPUT, OUTPUT und FORWARD unterscheiden

Typische Bedeutung:

```
INPUT:
Paket ist an den Linux-Host selbst adressiert.

OUTPUT:
Paket wurde auf dem Linux-Host erzeugt.

FORWARD:
Paket wird durch den Linux-Host weitergeleitet.
```

Containerverkehr kann beispielsweise durchlaufen:

```
PREROUTING
→ FORWARD
→ POSTROUTING
```

Eine Allow-Regel in `INPUT` hilft in diesem Fall nicht zwingend.

Linux: firewalld-Zustand prüfen

LESEND

Dienstzustand:

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

Aktive Zonen und Zuordnungen:

```
firewall-cmd --get-active-zones
```

Standardzone:

```
firewall-cmd --get-default-zone
```

Alle Zonen:

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

Bestimmte aktive Zone:

```
firewall-cmd \
  --zone=<Zone> \
  --list-all
```

Rich Rules:

```
firewall-cmd \
  --zone=<Zone> \
  --list-rich-rules
```

Richtlinienobjekte:

```
firewall-cmd --get-policies
```

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

Zu prüfen sind:

- welche Schnittstelle gehört zu welcher Zone?
- existiert eine Quelladresszuordnung?
- welche Services sind erlaubt?
- welche Ports sind erlaubt?
- welche Protokolle sind erlaubt?
- existieren Rich Rules?

- existieren ICMP-Blocks?
- existieren aktive Policy Objects?
- welches Zielverhalten besitzt die Zone?

Die Standardzone ist nicht automatisch die tatsächlich für jede Schnittstelle wirksame Zone.

Linux: firewalld-Laufzeit und permanente Konfiguration

firewalld unterscheidet:

Konfiguration	Wirkung
Runtime	aktuell aktiv, geht ohne Übernahme bei Reload oder Neustart verloren
Permanent	gespeichert, wird bei Reload oder Start zur Runtime-Konfiguration

Runtime-Konfiguration einer Zone:

```
firewall-cmd \
  --zone=<Zone> \
  --list-all
```

Permanente Konfiguration:

```
firewall-cmd \
  --permanent \
  --zone=<Zone> \
  --list-all
```

Beide Ausgaben müssen verglichen werden.

Mögliche Fehler:

- Regel ist permanent gespeichert, aber noch nicht geladen;
- Regel existiert nur zur Laufzeit und verschwindet nach Neustart;
- Reload hat eine Diagnosefreigabe entfernt;
- falsche Zone wurde geändert;
- Schnittstelle wechselte die Zone;
- Quelladressbindung besitzt eine andere Wirkung als erwartet.

Ein `firewall-cmd --reload` ist eine Änderung und darf nicht als erster Diagnoseschritt verwendet werden.

Linux: bestimmten firewalld-Port prüfen

Runtime:

```
firewall-cmd \
  --zone=<Zone> \
  --query-port=443/tcp
```

Service:

```
firewall-cmd \
  --zone=<Zone> \
  --query-service=https
```

Permanent:

```
firewall-cmd \
  --permanent \
  --zone=<Zone> \
  --query-port=443/tcp
```

Ein positives Ergebnis beweist nur, dass Port oder Service in dieser Zone konfiguriert ist. Es beweist nicht:

- dass die Schnittstelle zu dieser Zone gehört;
- dass keine Rich Rule blockiert;
- dass der Dienst lauscht;
- dass eine vorgelagerte Firewall erlaubt;
- dass der Client aus dem passenden Quellbereich kommt.

Linux: UFW prüfen

LESEND

Status und Standardrichtlinien:

```
sudo ufw status verbose
```

Nummerierte Regeln:

```
sudo ufw status numbered
```

Von UFW verwaltete hinzugefügte Regeln:

```
sudo ufw show added
```

Vollständigere Netfilter-Sicht:

```
sudo ufw show raw
```

Zu prüfen sind:

- aktiv oder inaktiv;
- eingehende Standardaktion;
- ausgehende Standardaktion;
- IPv4- und IPv6-Regeln;
- Quelladressen;
- Ports;
- Protokolle;
- Schnittstellen;
- Reihenfolge;
- gerouteter Verkehr.

`ufw status` zeigt nicht zwingend alle Regeln, die außerhalb von UFW erzeugt wurden. Ein Status `inactive` beweist deshalb nicht, dass im Kernel keine anderen Netfilterregeln aktiv sind.

Linux: vorhandene Firewallprotokolle prüfen

Kernelmeldungen im betroffenen Zeitraum:

LESEND · SENSITIV

```
journalctl \
  -k \
  --since "15 minutes ago"
```

firewalld-Dienstmeldungen:

```
journalctl \
  -u firewalld \
```

```
  --since "15 minutes ago"
```

UFW-Dienstmeldungen:

```
journalctl \
  -u ufw \
  --since "15 minutes ago"
```

Zu beachten:

- Dienstmeldungen sind nicht automatisch Paketlogs;
- eine Regel muss eine Log-Aktion besitzen, damit sie Paketdetails erzeugt;
- Logziele können Journal, Kernelringpuffer, Syslog oder NFLOG sein;
- Logpräfixe müssen zum Datenfluss passen;
- umfangreiche Paketprotokollierung kann Systeme und Datenträger belasten.

Neue Logregeln dürfen nur gezielt, rate-limitiert und zeitlich begrenzt erstellt werden.

Linux: eBPF-, XDP- und tc-Filter berücksichtigen

Pakete können vor oder außerhalb der erwarteten nftables- oder iptables-Regel verworfen werden.

Schnittstellendetails:

```
ip \
  -details \
  link show \
  dev <Schnittstelle>
```

`tc`-Ingressfilter:

```
sudo tc \
  filter show \
  dev <Schnittstelle> \
  ingress
```

`tc`-Egressfilter:

```
sudo tc \
  filter show \
  dev <Schnittstelle> \
  egress
```

Falls `bpftool` vorhanden ist:

```
sudo bpftool net
```

Mögliche Verursacher:

- CNI-Plugin;
- Kubernetes NetworkPolicy;
- Sicherheitsagent;
- DDoS-Schutz;
- Service Mesh;
- XDP-Programm;
- Traffic Control;
- Cloud- oder Hostingagent.

Diese Filter dürfen nicht ohne Kenntnis ihres Besitzers entfernt werden.

---

macOS: zwei Firewallbereiche unterscheiden

Unter macOS müssen mindestens zwei unterschiedliche Mechanismen getrennt betrachtet werden:

Mechanismus	Aufgabe
Application Layer Firewall	steuert eingehende Verbindungen anhand von Apps und Diensten
Packet Filter `pf`	paket- und regelbasierte Filterung auf Netzwerkebene

Zusätzlich können vorhanden sein:

- Network Extensions;
- Endpoint-Security-Produkte;
- VPN-Filter;
- Content Filter;
- DNS-Proxy;
- Drittanbieter-Firewall;
- MDM-Konfigurationsprofile.

Die macOS Application Firewall wird nicht primär als frei konfigurierbare Portfirewall verwaltet. Apple beschreibt sie als anwendungsbezogene Firewall.

---

macOS: Application Firewall in der Oberfläche prüfen

Aktueller Pfad:

```
Apple-Menü
→ Systemeinstellungen
→ Netzwerk
→ Firewall
```

Zu prüfen sind:

- Firewall aktiviert oder deaktiviert;
- „Alle eingehenden Verbindungen blockieren“;
- Liste zugelassener und blockierter Apps;
- automatisch zugelassene integrierte Software;
- automatisch zugelassene signierte Software;
- Tarnmodus;
- MDM-Verwaltung.

Die genaue Darstellung kann je nach macOS-Version und Geräteverwaltung abweichen.

---

macOS: Application Firewall im Terminal prüfen

LESEND

Gesamtstatus:

```
sudo /usr/libexec/ApplicationFirewall/socketfilterfw \
  --getglobalstate
```

Block-All-Zustand:

```
sudo /usr/libexec/ApplicationFirewall/socketfilterfw \
  --getblockall
```

Tarnmodus:

```
sudo /usr/libexec/ApplicationFirewall/socketfilterfw \
  --getstealthmode
```

Anwendungsliste:

```
sudo /usr/libexec/ApplicationFirewall/socketfilterfw \
  --listapps
```

Lokale Hilfe und verfügbare Optionen:

```
/usr/libexec/ApplicationFirewall/socketfilterfw \
  --help
```

Zusammenfassung über System Profiler:

```
system_profiler SPFirewallDataType
```

Zu prüfen sind:

- richtiger Anwendungspfad;
- signierte oder integrierte Anwendung;
- Allow- oder Blockstatus;
- globales Blockieren;
- MDM-Vorgabe;
- geänderte App nach Update;
- Launcher und tatsächlicher Listenerprozess.

---

macOS: pf-Zustand prüfen

LESEND · SENSITIV

Status:

```
sudo pfctl -s info
```

Filterregeln:

```
sudo pfctl -sr
```

NAT-Regeln:

```
sudo pfctl -sn
```

Zustandstabelle:

```
sudo pfctl -ss
```

Gesamtübersicht:

```
sudo pfctl -sa
```

Zu prüfen sind:

- ist `pf` aktiviert?
- welche Regeln und Anchors sind geladen?
- existieren Block- oder Pass-Regeln?
- welche Schnittstellen werden verwendet?
- existieren NAT- oder Redirect-Regeln?
- passt ein State zum betroffenen Datenfluss?
- verwaltet ein Systemdienst oder Sicherheitsprodukt den Anchor?

Apple weist darauf hin, dass Packet Filter keine unterstützte API für die Integration eigener Softwareprodukte ist. Systembestandteile können die Regeln verwalten und verändern. Die Diagnose vorhandener Zustände ist deshalb von einer dauerhaften eigenen Produktintegration zu unterscheiden.

---

macOS: Network Extensions und Sicherheitsfilter

Installierte System Extensions:

LESEND

```
systemextensionsctl list
```

Mögliche Filterkomponenten:

- Content Filter;
- Packet Tunnel;
- App Proxy;
- DNS Proxy;

- Endpoint Security;
- Hersteller-VPN;
- Zero-Trust-Agent.

Eine deaktivierte Application Firewall beweist nicht, dass keine Network Extension den Verkehr filtert.

Sicherheits- oder VPN-Erweiterungen dürfen nicht ohne Freigabe entfernt oder deaktiviert werden.

Docker und Host-Firewall

Docker kann auf dem Host eigene iptables- oder nftables-Regeln erzeugen, um:

- Bridge-Netzwerke zu isolieren;
- veröffentlichte Ports weiterzuleiten;
- NAT und Masquerading umzusetzen;
- DNS-Verkehr in Netzwerk-Namespaces zu behandeln.

Zu prüfen sind:

```
docker info
```

```
docker network ls
```

```
docker ps \
  --format 'table {{.Names}}\t{{.Ports}}'
```

```
docker inspect \
  <Containername> \
  --format '{{json .NetworkSettings.Ports}}'
```

Wichtige Punkte:

- veröffentlichter Containerverkehr kann den Forward-Pfad verwenden;
- Docker kann eigene Chains oder nftables-Tabellen erzeugen;
- Docker-Regeln dürfen nicht vorsorglich gelöscht werden;
- das Abschalten der Docker-Firewallverwaltung kann Containerkommunikation beschädigen;
- UFW-, firewalld-, iptables- und Docker-Sichten müssen zusammen betrachtet werden;
- Backend und Docker-Version müssen berücksichtigt werden.

Ein freigegebener Hostport beweist nicht, dass die Anwendung im Container lauscht.

Kubernetes und lokale Host-Firewall

Auf einem Kubernetes-Knoten können zusätzlich wirken:

- kube-proxy;
- CNI-Plugin;
- NetworkPolicy;
- eBPF-Regeln;
- NodePort;
- Service;
- Ingress;
- Host-Firewall des Knotens;
- Cloud-Sicherheitsgruppe.

NetworkPolicy ist nicht dasselbe wie die allgemeine Host-Firewall.

Zu prüfen sind:

```
kubectl get networkpolicy \
  --all-namespaces
```

```
kubectl get service \
  --all-namespaces
```

```
kubectl get pods \
  --all-namespaces \
  -o wide
```

```
kubectl get endpointslice \
  --all-namespaces
```

Bei hostnahen Problemen müssen zusätzlich CNI- und Knotenkonfiguration berücksichtigt werden.

TCP-Fehlerbilder einer lokalen Firewall

Paketbefund	Mögliche Einordnung
SYN erreicht Host nicht	vorgelagerter Pfad oder falsche Zieladresse

Paketbefund	Mögliche Einordnung
SYN erreicht Host, Firewall loggt Drop	lokale Host-Firewall bestätigt
SYN erreicht Host, kein Listener	Betriebssystem kann RST senden
SYN erreicht Host, Listener vorhanden, keine Antwort	lokale Filterung, Prozess oder Überlastung
Host sendet SYN/ACK, Client erhält es nicht	ausgehender Filter oder Rückweg
Verbindung wird sofort zurückgesetzt	Reject, fehlender Listener oder Anwendung
lokaler Test funktioniert, externer nicht	Bindung, Profil, Host-Firewall oder Netzwerkpfad
nur ein Quellnetz fehlerhaft	RemoteAddress- oder Zonenregel
nur IPv6 fehlerhaft	fehlende IPv6-Regel oder ICMPv6-Problem
nur nach VPN-Verbindung fehlerhaft	Profil-, Schnittstellen- oder VPN-Filter

---

UDP-Fehlerbilder einer lokalen Firewall

Paketbefund	Mögliche Einordnung
UDP-Anfrage erreicht Host nicht	vorgelagerter Pfad
Anfrage erreicht Host, Drop wird geloggt	lokale Host-Firewall
Anfrage erreicht Host und Prozess nicht	Hostfilter, Namespace oder Puffer
Anwendung antwortet, Antwort verlässt Host nicht	ausgehende Hostfilterung
keine Antwort und kein ICMP	geöffnet/still oder Drop
ICMP Port Unreachable	kein passender UDP-Endpunkt oder aktive Ablehnung
nur große Datagramme scheitern	Fragment-, MTU- oder ICMP-Filterung
Broadcast oder Multicast scheitert	Schnittstellen-, Zonen- oder Gruppenregel

Ein allgemeiner TCP-Porttest ist kein Nachweis für UDP.

ICMP und ICMPv6 prüfen

Firewallregeln müssen ICMPv4 und ICMPv6 getrennt betrachten.

Zu prüfen sind:

- Echo Request und Echo Reply;
- Destination Unreachable;
- Port Unreachable;
- Time Exceeded;
- Fragmentation Needed;
- Packet Too Big;

- Parameter Problem;
- notwendige IPv6-Kontrollmeldungen.

Das Blockieren von Ping beweist nicht, dass TCP oder UDP blockiert wird.

Umgekehrt kann Ping funktionieren, während ein TCP- oder UDP-Port blockiert ist.

Das vollständige Blockieren von ICMP oder ICMPv6 kann Path MTU Discovery und Diagnose beeinträchtigen.

Paketaufzeichnung und Firewallprotokoll kombinieren

Eine Paketaufzeichnung allein zeigt nicht immer, ob ein Paket alle lokalen Filterstufen passiert hat. Der Aufzeichnungspunkt kann vor oder nach bestimmten Filtern liegen.

Belastbarer Nachweis:

1. Testzeitpunkt genau dokumentieren.
2. Paketaufzeichnung auf dem Host starten.
3. Firewallcounter oder Protokoll erfassen.
4. Genau einen Test ausführen.
5. Listener und Prozesszustand erfassen.
6. Firewallprotokoll mit dem Flow abgleichen.
7. Paketaufzeichnung und Regelcounter vergleichen.
8. Test vom repräsentativen Client wiederholen.

Windows-Paketaufzeichnung mit pktmon

`pktmon` kann Paketerfassung und Drop-Erkennung innerhalb des Windows-Netzwerkstacks unterstützen.

Filter zurücksetzen:

SENSITIV · ÄNDERND

```
pktmon stop
```

```
pktmon filter remove
```

Filter für TCP-Port `443`:

```
pktmon filter add TCP443 -t TCP -p 443
```

Aufzeichnung starten:

```
pktmon start --capture --pkt-size 0 --file-name C:\Temp\firewall443.etl
```

Nach reproduziertem Fehler stoppen:

```
pktmon stop
```

Konvertieren:

```
pktmon etl2pcap C:\Temp\firewall443.etl --out C:\Temp\firewall443.pcapng
```

Filter entfernen:

```
pktmon filter remove
```

Das Zielverzeichnis muss vorhanden sein. Aufzeichnung und Filter müssen anschließend zurückgenommen werden.

Linux-Paketaufzeichnung

LESEND · SENSITIV

TCP-Port `443`:

```
sudo tcpdump \
  -ni any \
  'tcp port 443' \
  -c 200
```

UDP-Port `53` und ICMP:

```
sudo tcpdump \
  -ni any \
  '(udp port 53) or icmp or icmp6' \
  -c 200
```

Bestimmte Schnittstelle und Gegenstelle:

```
sudo tcpdump \
  -ni <Schnittstelle> \
  'host 192.0.2.100 and tcp port 443' \
  -c 200
```

Eine sichtbare eingehende Anfrage beweist, dass sie den Capturepunkt erreicht hat. Ob sie die Anwendung erreicht, muss zusätzlich mit Firewallcounter, Socket- und Anwendungsbefund geprüft werden.

---

macOS-Paketaufzeichnung

Verfügbare Schnittstellen:

```
tcpdump -D
```

Bestimmte Schnittstelle:

```
sudo tcpdump \
  -ni en0 \
  'host 192.0.2.100 and tcp port 443' \
  -c 200
```

Loopback:

```
sudo tcpdump \
  -ni lo0 \
  'tcp port 8080' \
  -c 200
```

Application Firewall, `pf`, Network Extension und Anwendung müssen getrennt ausgewertet werden.

Lokalen und entfernten Test vergleichen

Windows:

NETZAKTIV

```
Test-NetConnection `
  -ComputerName "127.0.0.1" `
  -Port 443
```

```
Test-NetConnection `
  -ComputerName "192.0.2.25" `
  -Port 443
```

Von einem entfernten Client:

```
Test-NetConnection `
  -ComputerName "app.example.test" `
  -Port 443 `
  -InformationLevel Detailed
```

Linux und macOS:

```
nc -vz 127.0.0.1 443
```

```
nc -vz 192.0.2.25 443
```

Von einem entfernten Client:

```
nc -vz app.example.test 443
```

Auswertung:

lokaler Test	entfernter Test	Mögliche Einordnung
erfolgreich	erfolgreich	getesteter TCP-Pfad funktioniert
erfolgreich	fehlerhaft	Bindung, Host-Firewall oder Netzwerkpfad
fehlerhaft	fehlerhaft	Listener, Dienst, Port oder lokale Filterung
Loopback erfolgreich, LAN-IP fehlerhaft	Bindung oder schnittstellenbezogene Regel	
IPv4 erfolgreich, IPv6 fehlerhaft	IPv6-Bindung oder IPv6-Firewall	

lokaler Test	entfernter Test	Mögliche Einordnung
ohne VPN erfolgreich, mit VPN fehlerhaft	Profil-, Schnittstellen- oder VPN-Regel	

Host-Firewall von Netzwerkfirewall abgrenzen

Nachweis	Einordnung
Paket erreicht Serverschnittstelle nicht	Problem vor dem Server
Paket erreicht Host und Host-Firewall loggt Drop	lokale Host-Firewall bestätigt
Paket erreicht Host, kein Drop, kein Listener	Dienst- oder Bindungsproblem
Paket erreicht Host, Listener vorhanden, Anwendung loggt nichts	lokaler Filter, Endpoint Security oder Anwendung
Antwort verlässt Serverschnittstelle	lokaler ausgehender Pfad wahrscheinlich passiert
Antwort verlässt Host nicht und Outbound-Drop wird geloggt	lokale ausgehende Firewall
Client- und Serveraufzeichnung unterscheiden sich	Verlust oder Filterung im Zwischenpfad
anderer Server im selben Netz funktioniert	hostspezifische Konfiguration wahrscheinlich

Eine Paketaufzeichnung an nur einem Punkt reicht häufig nicht zur eindeutigen Abgrenzung.

Hypothese und Gegenbeweis

Beispiel:

```
Hypothese:
Die aktive Windows-Firewall blockiert eingehende TCP-Verbindungen auf
Port 443, weil die Allow-Regel nur für das Domainprofil gilt, die
aktive Schnittstelle aber dem Public-Profil zugeordnet ist.

Erwarteter Befund:
Get-NetConnectionProfile zeigt Public.
Die Regel gilt nur für Domain.
Der lokale Listener auf Port 443 ist vorhanden.
Der lokale Test funktioniert.
Der entfernte Test schlägt fehl.
Das Firewallprotokoll zeigt einen Drop für den Testflow.

Gegenbeweis:
```

Die Schnittstelle verwendet Domain und eine wirksame Allow-Regel
passt vollständig zu Protokoll, Port, Quelladresse und Programm.

Testmethode:
Profil-, Regel-, Listener-, Protokoll- und Paketprüfung.

Risiko:
Die lesenden Prüfungen verändern den Firewallzustand nicht.

Linux-Beispiel:

Hypothese:
Die Netzwerkschnittstelle befindet sich in der firewalld-Zone public,
aber die Freigabe wurde in der Zone internal angelegt.

Erwarteter Befund:
firewall-cmd --get-active-zones ordnet die Schnittstelle public zu.
Port 443 ist in public nicht erlaubt, in internal jedoch vorhanden.
Der Listener existiert und ein externer Test schlägt fehl.

Gegenbeweis:
Die Schnittstelle ist internal zugeordnet oder eine andere wirksame
Regel erlaubt den konkreten Flow.

Kontrollierte Maßnahmen

Maßnahme	Voraussetzung	Risiko
vorhandene passende Regel aktivieren	Regel ist geprüft und nur deaktiviert	Dienst wird erreichbar
Profilzuordnung korrigieren	falsche Netzwerkkategorie nachgewiesen	Vertrauensniveau des Netzwerks ändert sich
gezielte Allow-Regel erstellen	konkreter legitimer Flow bestätigt	zusätzliche Angriffsfläche
zu breite Regel einschränken	unnötiger Geltungsbereich bestätigt	legitime Clients können ausfallen
falsche Blockregel korrigieren	Blockregel als Ursache nachgewiesen	Schutzwirkung kann reduziert werden
Programmpfad aktualisieren	Anwendungspfad hat sich nachweislich geändert	falsches Programm könnte freigegeben werden
RemoteAddress-Bereich korrigieren	Quellnetz stimmt nachweislich nicht	weitere Quellen können Zugriff erhalten

Maßnahme	Voraussetzung	Risiko
TCP-/UDP-Protokoll korrigieren	Protokollverwechslung bestätigt	zusätzlicher Transportweg wird geöffnet
IPv6-Regel ergänzen	Dienst soll IPv6 nutzen und Block ist bestätigt	zusätzliche IPv6-Erreichbarkeit
firewalld-Runtime und Permanent angleichen	Abweichung bestätigt	Wirkung über Neustart hinaus
Container-Weiterleitungsregel korrigieren	Forward-Pfad bestätigt	mehrere Container können betroffen sein
Endpoint-Security-Ausnahme beantragen	Produktfilter als Ursache bestätigt	zentrale Sicherheitswirkung
Logging zeitlich begrenzt aktivieren	vorhandene Daten reichen nicht	Speicher- und Datenschutzrisiko

Vor jeder Änderung müssen dokumentiert werden:

- genaue Regel;
- Richtlinienquelle;
- betroffener Flow;
- Sicherheitsauswirkung;
- verantwortlicher Besitzer;
- Freigabe;
- Ausgangszustand;
- Rückweg;
- Erfolgskriterium;
- vorgesehenes Testverfahren.

Systematischer Diagnoseablauf

1. Exakte Fehlermeldung aufnehmen.
2. Zeitpunkt und Zeitzone dokumentieren.
3. Client, Server und Anwendung bestimmen.
4. TCP, UDP, ICMPv4 oder ICMPv6 bestimmen.
5. Quell- und Zieladressen dokumentieren.
6. Quell- und Zielports dokumentieren.
7. Listener und Bindungsadresse prüfen.
8. Besitzenden Prozess bestimmen.
9. lokalen Loopbacktest durchführen.
10. konkrete lokale Schnittstellenadresse testen.
11. Test von einem repräsentativen entfernten Client durchführen.
12. Paketfluss eingehend, ausgehend oder weitergeleitet bestimmen.
13. aktive Firewalltechnik bestimmen.
14. aktives Profil, aktive Zone oder Schnittstelle bestimmen.
15. Standardaktionen prüfen.
16. wirksame Regeln erfassen.

17. Allow- und Blockregeln prüfen.
18. Protokoll- und Portfilter vergleichen.
19. Adressbereiche vergleichen.
20. Programm- und Dienstfilter vergleichen.
21. Schnittstellenfilter vergleichen.
22. Richtlinienquelle bestimmen.
23. zentrale Richtlinien und lokale Zusammenführung prüfen.
24. Firewallcounter und vorhandene Protokolle auswerten.
25. bei Bedarf zeitlich begrenzte Protokollierung aktivieren.
26. Paketaufzeichnung am Host durchführen.
27. Paketaufzeichnung mit einem zweiten Punkt vergleichen.
28. Container-, VM- und Namespacepfade berücksichtigen.
29. Endpoint Security, VPN und zusätzliche Filter berücksichtigen.
30. Hypothese und Gegenbeweis formulieren.
31. Genau eine kontrollierte Änderung durchführen.
32. identischen Test wiederholen.
33. Anwendung statt nur Porttest verifizieren.
34. weitere repräsentative Clients prüfen.
35. temporäre Protokollierung und Filter zurücknehmen.
36. Ursache, Maßnahme und Prävention dokumentieren.

Befundmatrix

Befund	Mögliche Einordnung	Nächster Nachweis
kein Listener	kein Firewallproblem nachgewiesen	Dienst und Bindung prüfen
lokaler Test funktioniert, remote nicht	Firewall, Bindung oder Netzwerkpfad	Serveraufzeichnung
Paket erreicht Server nicht	vorgelagerter Pfad	Netzwerkfirewall und Routing
Paket erreicht Server, Drop wird geloggt	lokale Host-Firewall	passende Regel bestimmen
Paket erreicht Server, kein Drop, kein App-Log	Endpoint Filter oder Anwendung	Prozess- und WFP/eBPF-Befund
Allow-Regel vorhanden, falsches Profil	Regel nicht wirksam	aktive Profilzuordnung
Allow-Regel vorhanden, falsche Zone	Regel nicht wirksam	firewalld-Zuordnung
Allow-Regel vorhanden, falsches Protokoll	TCP-/UDP-Verwechslung	Portfilter
Allow-Regel vorhanden, falsche Quelladresse	Scope passt nicht	Adressfilter
Allow-Regel vorhanden, alter Programmpfad	App-Filter passt nicht	Prozesspfad
Blockregel und Allow-Regel treffen zu	Blockregel kann Vorrang besitzen	vollständige Regelbedingungen
lokale Regel verschwindet	zentrale Verwaltung	PolicyStoreSource oder MDM

Befund	Mögliche Einordnung	Nächster Nachweis
UFW inaktiv, Verkehr blockiert	andere Netfilterregeln	`nft list ruleset`
firewalld-Port permanent, nicht runtime	noch nicht wirksam	Runtime vergleichen
firewalld-Port runtime, nicht permanent	verschwindet bei Reload	Permanent vergleichen
nft-Counter steigt an Drop-Regel	Regel trifft den Flow	Zeit und 5-Tupel prüfen
iptables-Counter bleibt unverändert	anderer Pfad oder Backend	nftables und Namespace prüfen
Application Firewall erlaubt App, trotzdem blockiert	`pf`, Network Extension oder VPN	weitere Filter prüfen
Docker-Port veröffentlicht, Hostzugriff scheitert	Forward-/Docker-Regel oder Container	Docker-Regeln und Listener
Hostdienst funktioniert, Container nicht	Namespace oder Forward-Pfad	Container-Socket und NAT
nur IPv6 blockiert	fehlende IPv6-Regel	`ip6`, `inet` oder ICMPv6
nur VPN betroffen	Schnittstellen-, Profil- oder VPN-Filter	Zustand mit und ohne VPN
Antwort verlässt Host, Client sieht sie nicht	Problem nach dem Host	Rückweg und Netzwerkfirewall

---

Typische Diagnosefehler

- Firewall prüfen, bevor der Listener geprüft wurde.
- Dienststatus mit Listener gleichsetzen.
- lokalen Loopbacktest als externen Nachweis verwenden.
- nur die GUI prüfen.
- persistente Konfiguration statt aktiver Richtlinie prüfen.
- Windows `PersistentStore` mit `ActiveStore` verwechseln.
- nur nach einer Allow-Regel suchen.
- Blockregeln nicht berücksichtigen.
- Windows-Regeln wie eine einfache ACL-Liste interpretieren.
- aktives Windows-Profil nicht prüfen.
- VPN- und virtuelle Schnittstellen ignorieren.
- Programmpfad und Dienstfilter nicht prüfen.
- Quelladressbereich nicht prüfen.
- TCP und UDP verwechseln.
- IPv4- und IPv6-Regeln nicht getrennt prüfen.
- Ping als vollständigen Firewalltest verwenden.
- ICMPv6 vollständig blockieren.
- firewalld-Standardzone mit aktiver Zone gleichsetzen.
- firewalld-Runtime und Permanent nicht vergleichen.
- UFW-Status als vollständigen Netfilterzustand bewerten.
- iptables und nftables unkontrolliert parallel verändern.

- Chain-Hooks und Policies nicht prüfen.
- Regelcounter ohne Vorher-Nachher-Vergleich bewerten.
- Containerverkehr im `INPUT`- statt `FORWARD`-Pfad suchen.
- Docker-Regeln vorsorglich löschen.
- Kubernetes NetworkPolicy mit Host-Firewall gleichsetzen.
- macOS Application Firewall mit `pf` gleichsetzen.
- Network Extensions und Endpoint Security ignorieren.
- Paketaufzeichnung allein als Nachweis der Firewallaktion verwenden.
- Firewallprotokollierung dauerhaft und unbegrenzt aktivieren.
- die Firewall vollständig deaktivieren.
- Sicherheitssoftware beenden oder deinstallieren.
- mehrere Regeln gleichzeitig ändern.
- keine Rückfallmöglichkeit dokumentieren.
- nur einen einzelnen erfolgreichen Test durchführen.
- temporäre Diagnosefreigaben aktiv lassen.

Verifikation

Nach einer Maßnahme müssen mindestens folgende Punkte geprüft werden:

- der erwartete Listener ist vorhanden;
- richtige Adresse und richtiger Port sind gebunden;
- der richtige Prozess besitzt den Socket;
- das richtige Firewallprofil ist aktiv;
- die richtige Linux-Zone ist aktiv;
- richtige Schnittstelle ist zugeordnet;
- wirksame Standardaktionen entsprechen dem Sollzustand;
- die vorgesehene Regel ist aktiviert;
- Regelrichtung stimmt;
- TCP, UDP, ICMPv4 oder ICMPv6 stimmt;
- lokale und entfernte Ports stimmen;
- lokale und entfernte Adressbereiche stimmen;
- Programm- und Dienstfilter stimmen;
- Profil- beziehungsweise Zonenumfang stimmt;
- keine widersprechende Blockregel greift;
- Richtlinienquelle ist dokumentiert;
- lokaler Test funktioniert;
- Test über die konkrete Schnittstellenadresse funktioniert;
- entfernter Test funktioniert;
- ursprüngliche Anwendung funktioniert;
- IPv4 funktioniert, sofern vorgesehen;
- IPv6 funktioniert, sofern vorgesehen;
- UDP funktioniert, sofern vorgesehen;
- notwendige ICMP- und ICMPv6-Meldungen funktionieren;
- Container- oder VM-Pfad funktioniert;

- Firewallcounter zeigen den erwarteten Pfad;
- keine neuen unerwarteten Drops entstehen;
- Sicherheitsumfang wurde nicht unnötig erweitert;
- temporäre Regeln wurden entfernt;
- temporäre Protokollierung wurde zurückgesetzt;
- mehrere repräsentative Clients funktionieren;
- Ursache, Maßnahme und Prävention wurden dokumentiert.

Eine vollständig deaktivierte Firewall ist keine gültige Verifikation einer sicheren Lösung.

Dokumentationsvorlage

```
Störung:
<exakte Beschreibung>

Zeitpunkt:
<Datum, Uhrzeit und Zeitzone>

Client:
<Hostname und IP-Adresse>

Server:
<Hostname und IP-Adresse>

Anwendung:
<Dienst und Prozess>

Richtung:
<eingehend, ausgehend oder weitergeleitet>

Transportprotokoll:
<TCP, UDP, ICMPv4 oder ICMPv6>

Quelladresse und Port:
<Adresse und Port>

Zieladresse und Port:
<Adresse und Port>

Listener:
```

<Adresse, Port, Prozess und PID>

Schnittstelle:

<Name und Typ>

Firewalltechnik:

<Windows Firewall, nftables, firewalld, UFW, pf oder andere>

Aktives Profil oder Zone:

<Wert>

Standardaktion:

<eingehend und ausgehend>

Wirksame Regel:

<Name, Aktion und Bedingungen>

Richtlinienquelle:

<lokal, GPO, MDM oder anderes>

Portfilter:

<Befund>

Adressfilter:

<Befund>

Programm- oder Dienstfilter:

<Befund>

Firewallprotokoll:

<Zeitpunkt und Aktion>

Paketaufzeichnung:

<eingehender und ausgehender Befund>

Zusätzliche Filter:

<Endpoint Security, VPN, Container oder Hypervisor>

Nachgewiesene Ursache:

```
<technischer Befund>

Gegenbeweis ausgeschlossen durch:
<Test und Ergebnis>

Durchgeführte Maßnahme:
<genau eine kontrollierte Änderung>

Sicherheitsauswirkung:
<Beschreibung>

Risiko und Rückweg:
<Beschreibung>

Verifikation:
<identischer Test, Anwendung und weitere Systeme>

Prävention:
<Monitoring oder Konfigurationsverbesserung>
```

Checkliste

- [ ] exakte Fehlermeldung dokumentiert
- [ ] Zeitpunkt und Zeitzone erfasst
- [ ] Client und Server bestimmt
- [ ] Anwendung bestimmt
- [ ] Prozess und Dienst bestimmt
- [ ] Richtung bestimmt
- [ ] TCP, UDP, ICMPv4 oder ICMPv6 bestimmt
- [ ] Quelladresse erfasst
- [ ] Zieladresse erfasst
- [ ] Quellport erfasst
- [ ] Zielport erfasst
- [ ] Listener geprüft
- [ ] Bindungsadresse geprüft
- [ ] Prozessbesitzer geprüft
- [ ] lokalen Loopbacktest durchgeführt

- [ ] konkrete lokale Adresse getestet
- [ ] entfernten Clienttest durchgeführt
- [ ] Host-, Container- und VM-Pfad unterschieden
- [ ] aktive Firewalltechnik bestimmt
- [ ] Windows-Profil geprüft
- [ ] Linux-Zone geprüft
- [ ] Schnittstellenzuordnung geprüft
- [ ] Firewallstatus geprüft
- [ ] eingehende Standardaktion geprüft
- [ ] ausgehende Standardaktion geprüft
- [ ] wirksame aktive Regeln erfasst
- [ ] Allow-Regeln geprüft
- [ ] Blockregeln geprüft
- [ ] Regelrichtung geprüft
- [ ] Profil- oder Zonenumfang geprüft
- [ ] Protokollfilter geprüft
- [ ] lokalen Port geprüft
- [ ] entfernten Port geprüft
- [ ] lokale Adresse geprüft
- [ ] entfernte Adresse geprüft
- [ ] Programmpfad geprüft
- [ ] Dienstfilter geprüft
- [ ] Schnittstellenfilter geprüft
- [ ] Richtlinienquelle bestimmt
- [ ] lokale Regelzusammenführung geprüft
- [ ] GPO oder MDM berücksichtigt
- [ ] Firewallcounter ausgewertet
- [ ] vorhandene Protokolle ausgewertet
- [ ] Client- und Serverzeit korreliert
- [ ] Paketaufzeichnung durchgeführt
- [ ] zweiten Aufzeichnungspunkt berücksichtigt
- [ ] IPv4 geprüft
- [ ] IPv6 geprüft
- [ ] ICMPv4 geprüft

- [ ] ICMPv6 geprüft
- [ ] Container-Firewallregeln berücksichtigt
- [ ] Hypervisorfilter berücksichtigt
- [ ] VPN-Filter berücksichtigt
- [ ] Endpoint Security berücksichtigt
- [ ] nftables und iptables unterschieden
- [ ] firewalld-Runtime geprüft
- [ ] firewalld-Permanent geprüft
- [ ] UFW-Rohzustand bei Bedarf geprüft
- [ ] macOS Application Firewall geprüft
- [ ] macOS `pf` geprüft
- [ ] macOS Network Extensions berücksichtigt
- [ ] Hypothese formuliert
- [ ] Gegenbeweis festgelegt
- [ ] Sicherheitsauswirkung dokumentiert
- [ ] Risiko und Rückweg dokumentiert
- [ ] nur eine kontrollierte Maßnahme durchgeführt
- [ ] identischen Test wiederholt
- [ ] ursprüngliche Anwendung getestet
- [ ] weitere repräsentative Clients geprüft
- [ ] temporäre Regel entfernt
- [ ] Protokollierung zurückgesetzt
- [ ] Ursache und Prävention dokumentiert

Schnellreferenz

Aufgabe	Befehl
Windows-Netzwerkprofil	`Get-NetConnectionProfile`
Windows-Firewallprofile	`Get-NetFirewallProfile`
wirksame Windows-Regeln	`Get-NetFirewallRule -PolicyStore ActiveStore`
aktivierte Windows-Regeln	`Get-NetFirewallRule -PolicyStore ActiveStore -Enabled True`
Windows-Blockregeln	`Get-NetFirewallRule -PolicyStore ActiveStore -Enabled True -Action Block`
Windows-Portfilter	`Get-NetFirewallPortFilter -PolicyStore ActiveStore`
Windows-Adressfilter	`Get-NetFirewallAddressFilter -PolicyStore ActiveStore`

Aufgabe	Befehl
Windows-Programmfilter	`Get-NetFirewallApplicationFilter -PolicyStore ActiveStore`
Windows-Dienstfilter	`Get-NetFirewallServiceFilter -PolicyStore ActiveStore`
vollständige Windows-Regeln	`Show-NetFirewallRule -PolicyStore ActiveStore`
Windows-Firewallprotokoll	`Get-Content "$env:SystemRoot\System32\LogFiles\Firewall\pfirewall.log" -Tail 100`
Windows-WFP-Blockereignisse	`Get-WinEvent -FilterHashtable @{LogName="Security"; Id=5152,5155,5157,5159}`
Windows-netsh-Profile	`netsh advfirewall show allprofiles`
Windows-netsh-Regeln	`netsh advfirewall firewall show rule name=all verbose`
nftables-Regelsatz	`sudo nft list ruleset`
nftables mit Handles	`sudo nft -a list ruleset`
iptables-Regeln	`sudo iptables -L -n -v --line-numbers`
IPv6-iptables	`sudo ip6tables -L -n -v --line-numbers`
iptables-Gesamtsicht	`sudo iptables-save -c`
firewalld-Zustand	`firewall-cmd --state`
aktive firewalld-Zonen	`firewall-cmd --get-active-zones`
firewalld-Zone	`firewall-cmd --zone=<Zone> --list-all`
permanente firewalld-Zone	`firewall-cmd --permanent --zone=<Zone> --list-all`
firewalld-Port prüfen	`firewall-cmd --zone=<Zone> --query-port=<Port>/<Protokoll>`
UFW-Status	`sudo ufw status verbose`
nummerierte UFW-Regeln	`sudo ufw status numbered`
vollständige UFW-Sicht	`sudo ufw show raw`
Kernelprotokoll	`journalctl -k --since "15 minutes ago"`
macOS-Firewallstatus	`sudo /usr/libexec/ApplicationFirewall/socketfilterfw --getglobalstate`
macOS-App-Liste	`sudo /usr/libexec/ApplicationFirewall/socketfilterfw --listapps`
macOS-Firewallübersicht	`system_profiler SPFirewallDataType`
macOS-pf-Status	`sudo pfctl -s info`
macOS-pf-Regeln	`sudo pfctl -sr`
macOS-pf-States	`sudo pfctl -ss`
macOS-System-Extensions	`systemextensionsctl list`
Windows-TCP-Test	`Test-NetConnection -ComputerName <Ziel> -Port <Port>`
Linux/macOS-TCP-Test	`nc -vz <Ziel> <Port>`
Linux-Aufzeichnung	`sudo tcpdump -ni any 'tcp port <Port>' -c 200`

Befehle und Maßnahmen, die nicht unkontrolliert als erste Diagnose verwendet werden dürfen

```
Set-NetFirewallProfile -Enabled False
netsh advfirewall set allprofiles state off
Disable-NetFirewallRule ohne exakte Regelbegrenzung
Enable-NetFirewallRule ohne exakte Regelbegrenzung
Remove-NetFirewallRule
New-NetFirewallRule ohne dokumentierten Scope
netsh int ip reset
netsh winsock reset
nft flush ruleset
iptables -F
iptables -X
ip6tables -F
ufw disable
ufw reset
firewall-cmd --reload
firewall-cmd --complete-reload
systemctl stop firewalld
systemctl restart firewalld
pfctl -d
pfctl -F all
Änderung oder Entfernen von Docker-Regeln
Deaktivieren der Docker-Firewallverwaltung
Entfernen von Kubernetes- oder CNI-Regeln
Deaktivieren eines VPN-Filters
Beenden oder Deinstallieren von Endpoint Security
dauerhafte ungefilterte Paketprotokollierung
```

Ein vollständiges Deaktivieren der Firewall verändert den Sicherheitszustand, kann Richtlinien verletzen und liefert keinen präzisen Nachweis darüber, welche Regel den ursprünglichen Fehler verursacht hat.

Quellen

Offizielle Microsoft-Dokumentation

- Microsoft Learn – Windows Firewall overview

- Microsoft Learn – Windows Firewall rules
- Microsoft Learn – Manage Windows Firewall with the command line
- Microsoft Learn – Configure Windows Firewall logging
- Microsoft Learn – Get-NetFirewallProfile
- Microsoft Learn – Get-NetFirewallRule
- Microsoft Learn – Show-NetFirewallRule
- Microsoft Learn – Get-NetFirewallPortFilter
- Microsoft Learn – Get-NetFirewallAddressFilter
- Microsoft Learn – Get-NetFirewallApplicationFilter
- Microsoft Learn – Get-NetFirewallServiceFilter
- Microsoft Learn – WFP auditing and logging
- Microsoft Learn – Event 5152
- Microsoft Learn – Event 5157
- Microsoft Learn – Packet Monitor
- Microsoft Learn – netsh advfirewall

Offizielle Netfilter- und Linux-Dokumentation

- nftables Documentation
- nftables – Counters
- Linux man-pages – nft(8)
- Linux man-pages – iptables(8)
- Linux man-pages – iptables-save(8)
- Linux man-pages – tcpdump(8)

Offizielle firewalld- und Ubuntu-Dokumentation

- firewalld – Documentation
- firewalld – firewall-cmd
- firewalld – Runtime and permanent configuration
- firewalld – Zone configuration
- firewalld – Policy Objects
- Ubuntu Manpage – ufw(8)

Offizielle Apple-Dokumentation

- Apple – Change Firewall settings on Mac

- Apple – Block connections to your Mac with a firewall
- Apple Platform Security – Firewall security in macOS
- Apple Deployment – Firewall payload settings
- Apple – TCP and UDP ports used by Apple software products
- Apple Developer – Packet Filter is not API

Offizielle Containerdokumentation

- Docker Docs – Packet filtering and firewalls
- Docker Docs – Docker with iptables
- Docker Docs – Docker with nftables
- Docker Docs – Port publishing and mapping
- Kubernetes – Network Policies
- Kubernetes – Services

Für diese Seite wurden keine Community-Berichte oder Social-Media-Aussagen als technische Nachweise verwendet.
