Holger Reibold

BEGLEITHEFT

Praxislabor NetBird

Missionen, Spiel und Übungslabor
zum Buch „Das NetBird-Handbuch“

BRAIN-MEDIA.DE

© 2026 Brain-Media.de. Dieses Begleitheft ergänzt das Buch „Das NetBird-Handbuch. Zero-Trust-Netzwerke mit WireGuard aufbauen, betreiben und prüfsicher dokumentieren“. Es darf von Käuferinnen und Käufern des Buchs für den eigenen Gebrauch ausgedruckt werden.

Die Saarmetall GmbH ist ein fiktives Unternehmen. NetBird und WireGuard sind Marken ihrer jeweiligen Inhaber. Das Labor verwendet NetBird in Version 0.81.3.

Brain-Media.de · Dr. Holger Reibold · Huber-Müller-Str. 52 · 66113 Saarbrücken
info@brain-media.de · www.brain-media.de

Stand: Oktober 2026 · Laborskript saarmetall-lab.sh, Version 1.0

INHALT

Inhaltsverzeichnis

1 Vom Lesen zum Handeln 4

2 Die Missionen im Browser 7

3 Das eigene Labor einrichten 9

4 Werkzeuge: netbird, docker exec und die REST-API 12

5 Die zehn Missionen 14

- Mission 1 bis 3: Client und Verbindung 14
- Mission 4: Setup-Keys 16
- Mission 5 und 6: Richtlinien 16
- Mission 7: Networks und Routing Peers 17
- Mission 8: DNS im Overlay 18
- Mission 9: Posture Checks 19
- Mission 10: Der fremde Peer 19

6 Wenn etwas nicht klappt 21

Anhang A: Spickzettel REST-API 23

Anhang B: Lösungswege 26

KAPITEL 1

Vom Lesen zum Handeln

Das NetBird-Handbuch erklärt, wie die Saarmetall GmbH ihr altes SSL-VPN durch ein Zero-Trust-Netz ersetzt: mit Gruppen, Richtlinien, Setup-Keys, Routing Peers, Nameserver-Gruppen und Posture Checks. Wer das Buch gelesen hat, kennt die Konzepte. Sicher im Betrieb wird man aber erst, wenn man sie selbst angewendet hat, und vor allem, wenn man einen Fehler darin gesucht und behoben hat.

Genau dafür gibt es das Praxislabor. Es besteht aus zwei Teilen, die dieselben zehn Missionen anbieten:

- **AdminMission im Browser.** Die Missionen laufen in einer Simulation auf praxislabor.brain-media.de. Sie brauchen nichts zu installieren, und nichts kann kaputtgehen. Die Missionen werden mit einer Frage aus dem Buch freigeschaltet.
- **Das eigene Labor.** Das Skript `saarmetall-lab.sh` baut auf einer Debian-VM das Netz der Saarmetall GmbH nach: eine selbst betriebene NetBird-Instanz, sieben Peers als Docker-Container und zwei CNC-Steuerungen im OT-Netz. Es baut den Fehler einer Mission ein und prüft hinterher, ob Sie ihn behoben haben.

Beide Teile verwenden dieselben Befehle: den NetBird-Client `netbird`, `docker exec` für die Peers und `curl` gegen die REST-API der Instanz. Was Sie im Browser üben, funktioniert im Labor genauso. Eine sinnvolle Reihenfolge ist: Kapitel im Buch lesen, Mission im Browser lösen, dieselbe Mission im Labor wiederholen. Beim zweiten Mal merken Sie, ob Sie das Prinzip verstanden haben oder sich nur an die Hinweise erinnern.

Die Saarmetall GmbH im Labor

Das Labor übernimmt die Namen, Gruppen und Richtlinien aus dem Buch. Es bildet nur den Ausschnitt nach, den die Missionen brauchen. Abbildung 1.1 zeigt die Bestandteile.

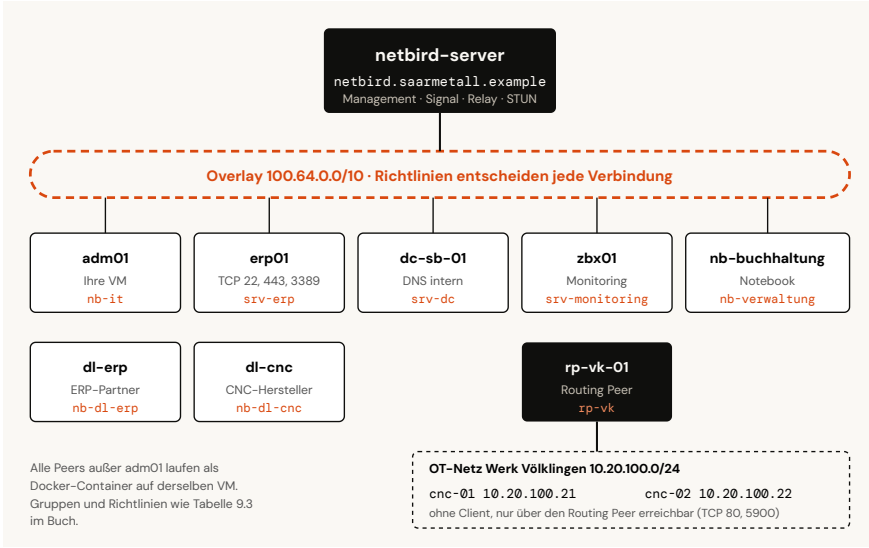


Abbildung 1.1: Das Saarmetall-Labor. Eine VM trägt den NetBird-Server, sieben Peers und das OT-Netz mit zwei Steuerungen. Ihr Arbeitsplatz ist die VM selbst (adm01).

Peer	Gruppen	Rolle im Buch
`adm01`	nb-it, dev-linux	Ihr Arbeitsplatz in der IT
`erp01`	srv-erp, loc-saarbruecken	ERP-Server (Kapitel 1.6, Tabelle 1.2)
`dc-sb-01`	srv-dc, loc-saarbruecken	Domänencontroller, DNS für intern.saarmetall.example (Kapitel 12)
`zbx01`	srv-monitoring	Zabbix-Monitoring
`rp-vk-01`	rp-vk, loc-voelklingen	Routing Peer Werk Völklingen (Tabelle 11.1)
`nb-buchhaltung`	nb-verwaltung, dev-linux	Notebook der Buchhaltung
`dl-erp`, `dl-cnc`	nb-dl-erp, nb-dl-cnc	Zugänge der Dienstleister (Kapitel 5.3, 9.4)

Tabelle 1.1: Die Peers des Labors

Die Richtlinien im Ausgangszustand entsprechen Tabelle 9.3 des Buchs: *IT auf Server*, *Verwaltung auf ERP* (TCP 443, gebunden an den Posture Check *Client aktuell*), *ERP-Dienstleister* (TCP 3389), *CNC-Fernwartung* (TCP 5900 auf die

Ressourcengruppe res-cnc-hersteller), *DNS intern* (UDP 53) und *Monitoring*. Die Default-Richtlinie ist abgeschaltet. Das Network *Werk Völklingen* enthält den Routing Peer rp-vk-01 und die Ressource cnc-01. Die Nameserver-Gruppe *ns-intern* verweist für intern.saarmetall.example auf dc-sb-01.

KAPITEL 2

Die Missionen im Browser

AdminMission ist ein Lernspiel im Browser. Sie arbeiten an einer simulierten Kommandozeile, bekommen Tickets von Kolleginnen und Kollegen der Saarmetall GmbH und lösen sie mit echten Befehlen. Das Spiel prüft nach jedem Befehl, welche Ziele erfüllt sind.

Freischalten

1. Öffnen Sie **praxislabor.brain-media.de** und wählen Sie AdminMission.
2. Im Katalog der Buchpakete wählen Sie *Das NetBird-Handbuch* und *Freischalten*.
3. Beantworten Sie die Frage. Die Antwort steht im Buch, meist in einer Tabelle des Fallbeispiels.

Die Freischaltung gilt für diesen Browser. Sie wird lokal gespeichert, ein Konto ist nicht nötig. Wer den Browser wechselt, beantwortet die Frage dort noch einmal.

So arbeiten Sie im Spiel

Jede Mission beginnt mit einem Ticket und einem Auftrag. Rechts sehen Sie die Ziele, die erfüllt sein müssen. Gestufte Hinweise helfen, wenn Sie nicht weiterkommen. Die erste Stufe gibt eine Richtung vor, die letzte nennt den Befehl. Wer ohne Hinweise auskommt, erhält mehr Punkte.

Die Simulation kennt die Befehle, die Sie für die Missionen brauchen: `netbird` mit `status`, `up` und `down`, `systemctl`, `docker ps`, `docker exec`, `docker logs`, `docker start`, `curl` für die API und `jq` mit den gängigen Filtern. Innerhalb der Container stehen `netbird`, `nc`, `ping`, `curl`, `nslookup`, `dig` und `getent` bereit.

Praxistipp

Schreiben Sie die Befehle aus dem Spiel mit. Im Labor brauchen Sie dieselben Befehle wieder, und eine eigene Befehlsliste ist im Betrieb mehr wert als jede Musterlösung.

KAPITEL 3

Das eigene Labor einrichten

Das Labor läuft in einer einzigen virtuellen Maschine. Es installiert Docker, startet darin den NetBird-Server und die Peers und richtet die VM selbst als Admin-Rechner adm01 ein. Planen Sie für die Einrichtung eine halbe Stunde ein, den größten Teil davon für das Laden der Images.

Die virtuelle Maschine

Ressource	Empfehlung	Bemerkung
Betriebssystem	Debian 13 (Trixie)	netinst, nur SSH-Server und Standard-Systemwerkzeuge
Prozessorkerne	2	10 Container laufen gleichzeitig
Arbeitsspeicher	4 GB	2 GB reichen knapp
Systemplatte	20 GB	Images und Daten belegen rund 2 GB
Netzwerk	NAT oder Bridge, Internetzugang	für Pakete und Docker-Images
Benutzer	`trainee` mit sudo-Rechten	wie im Linux-Labor des Verlags

Tabelle 3.1: Eckdaten der Labor-VM

Wer bereits die VM aus „Linux-Administration lernen“ betreibt, legt für NetBird besser eine zweite VM an. Beide Labore verändern Netz und Dienste, und sie sollen sich nicht gegenseitig stören.

> **Achtung**
>
> Das Skript installiert Docker und einen NetBird-Client, trägt netbird.saarmetall.example in /etc/hosts ein und vertraut einer eigenen Labor-CA. Verwenden Sie es nur in einer Übungs-VM, nie auf einem produktiven Rechner. Legen Sie vor der Installation einen Schnappschuss an. Damit kehren Sie jederzeit zum sauberen System zurück.

Installation

```
wget https://praxislabor.brain-media.de/labor/saarmetall-lab.sh
sha256sum saarmetall-lab.sh
chmod +x saarmetall-lab.sh
sudo ./saarmetall-lab.sh install
```

Vergleichen Sie die Prüfsumme mit dem Wert auf der Downloadseite, bevor Sie das Skript ausführen. Lesen Sie es außerdem durch. Es ist kommentiert und zeigt, wie man eine NetBird-Instanz vollständig über die API einrichtet, was Kapitel 16.4 des Buchs beschreibt.

Die Installation läuft in diesen Schritten ab:

1. Pakete installieren: Docker, Docker Compose, jq, curl.
2. Labor-CA und Zertifikat für netbird.saarmetall.example erzeugen, config.yaml und docker-compose.yml schreiben.
3. Images laden: `netbirdio/netbird-server` und `netbirdio/netbird` in Version 0.81.3. Aus dem Client-Image entsteht das Peer-Image mit kleinen Testdiensten.
4. Den Server starten und den ersten Benutzer per `/api/setup` anlegen. Dabei entsteht der API-Token, den Sie in `~/.netbird-token` finden.
5. Gruppen anlegen, die Peers mit Einmal-Setup-Keys anmelden und ihnen die Gruppen aus dem Buch zuweisen.
6. Den NetBird-Client auf der VM einrichten und adm01 anmelden.
7. Richtlinien, Posture Check, Network mit Routing Peer und die Nameserver-Gruppe anlegen.

Melden Sie sich danach einmal ab und wieder an, damit Ihr Benutzer der Gruppe docker angehört. `sudo ./saarmetall-lab.sh status` zeigt, ob alle Peers verbunden sind.

Die Befehle des Labors

Aufruf	Wirkung
`sudo ./saarmetall-lab.sh install`	Labor einmalig einrichten
`sudo ./saarmetall-lab.sh mission 5`	Ausgangszustand herstellen und den Fehler aus Mission 5 einbauen
`sudo ./saarmetall-lab.sh check 5`	Ziele von Mission 5 prüfen
`sudo ./saarmetall-lab.sh check 3 zbx01`	Antwort einer Mission prüfen
`sudo ./saarmetall-lab.sh reset`	Ausgangszustand herstellen
`sudo ./saarmetall-lab.sh list`	Missionen mit Buchkapitel anzeigen
`sudo ./saarmetall-lab.sh status`	Server, Token und Peers prüfen

Tabelle 3.2: Aufrufe des Laborskripts

Nach der Installation liegt das Skript zusätzlich unter `/usr/local/sbin/saarmetall-lab`, Sie können es also auch ohne Pfad aufrufen. Die Prüfung gibt für jedes Ziel ein Ergebnis aus:

```
trainee@adm01:~$ sudo saarmetall-lab check 5
Mission 5: Default-Allow muss weg
  [ok]    Keine aktive Default-Richtlinie mehr
  [ok]    Die Buchhaltung erreicht das ERP auf TCP 443
  [ok]    Die Buchhaltung erreicht das ERP nicht per SSH (TCP 22)
  [fehlt] dl-cnc erreicht das ERP nicht mehr
  [ok]    Der Monitoring-Server erreicht seine Agenten weiterhin
4 von 5 Zielen erfüllt.
```

Die Prüfungen arbeiten wie im Spiel mit dem tatsächlichen Zustand: Das Skript fragt die API ab und baut aus den Containern echte Verbindungen auf. Ein [ok] bei „erreicht nicht“ bedeutet, dass NetBird die Verbindung verworfen hat.

KAPITEL 4

Werkzeuge: netbird, docker exec und die REST-API

Der Client auf adm01

Auf der VM läuft der NetBird-Client als systemd-Dienst `netbird`. Die Befehle aus Anhang A des Buchs funktionieren unverändert: `netbird status`, `netbird status -d`, `netbird up`, `netbird down`.

Die Peers in den Containern

Jeder Peer ist ein Container mit eigenem NetBird-Client. Befehle führen Sie mit `docker exec` darin aus:

```
docker ps
docker exec nb-buchhaltung netbird status
docker exec nb-buchhaltung nc -zv -w 3 erp01.netbird.selfhosted 44
3
docker exec nb-buchhaltung nslookup erp01.intern.saarmetall.exampl
e
docker logs nb-buchhaltung
```

`nc -zv` ist das wichtigste Prüfwerkzeug. „succeeded“ heißt: Die Richtlinie lässt die Verbindung zu und der Dienst antwortet. „timed out“ heißt: NetBird hat die Pakete verworfen, weil keine Richtlinie passt. „Connection refused“ heißt: Die Verbindung kommt an, aber auf dem Port lauscht kein Dienst.

Die REST-API

Ein Dashboard bringt das Labor nicht mit. Sie verwalten die Instanz wie in Kapitel 16.4 des Buchs über die REST-API. Das ist näher an der Automatisierung im Betrieb, und jede Änderung ist ein nachvollziehbarer Befehl.

```
TOKEN=$(cat ~/.netbird-token)
API=https://netbird.saarmetall.example/api
curl -s -H "Authorization: Token $TOKEN" $API/peers | jq '.[] | {n
ame, ip, connected}'
```

Zwei Regeln ersparen die meisten Fehlermeldungen:

- **Gruppen als ID.** Richtlinien, Setup-Keys und Nameserver-Gruppen erwarten Gruppen-IDs, nicht Namen. Die ID ermitteln Sie mit `jq -r '.[] | select(.name == "nb-verwaltung") | .id'` aus `GET $API/groups`.
- **PUT ersetzt das ganze Objekt.** Wer eine Richtlinie ändert, schickt sie vollständig zurück: Name, enabled und alle Regeln. Am einfachsten lesen Sie das Objekt mit GET in eine Datei, bearbeiten es und schicken es mit `curl -X PUT -d @datei.json` zurück. In den Regeln ersetzen Sie dabei die Gruppenobjekte durch ihre IDs.

Anhang A fasst die Aufrufe zusammen, die Sie für die Missionen brauchen.

KAPITEL 5

Die zehn Missionen

Jede Mission nennt das passende Buchkapitel, das Ticket, den Auftrag und die Ziele, die das Spiel und das Laborskript prüfen. Die Lösungswege stehen in Anhang B. Versuchen Sie es zuerst ohne.

Mission 1 bis 3: Client und Verbindung

Mission 1: Der Agent schläft

Buch: Kapitel 6.4 Von: Lena Kraus (Administration)

> *Dein Arbeitsplatz adm01 taucht im Dashboard seit dem Wochenende als offline auf. Der NetBird-Client ist installiert, aber offenbar läuft er nicht. Ohne ihn erreichst du keinen einzigen Server.*

Auftrag: Bring den NetBird-Dienst auf adm01 wieder zum Laufen, sorge dafür, dass er beim Booten startet, und prüfe die Verbindung mit netbird status.

- ☐ Dienst netbird läuft
- ☐ Dienst netbird startet beim Booten
- ☐ adm01 ist mit der Steuerungsebene verbunden

Worum es geht: Der NetBird-Client besteht aus einem Daemon und der Kommandozeile, die über den Unix-Socket /var/run/netbird.sock mit ihm spricht. Läuft der Daemon nicht, scheitert jeder netbird-Befehl. Auf Servern gehört der Dienst immer in den Autostart. Im Buch: Kapitel 6.4.

Mission 2: Anmelden am eigenen Management

Buch: Kapitel 5 und 8.3 Von: Martin Scholl (IT-Leitung)

> *adm01 wurde neu aufgesetzt und hängt noch an keiner Instanz. Wir betreiben NetBird selbst unter https://netbird.saarmetall.example. Lena hat dir einen Einmal-Setup-Key unter /root/netbird-setup-key.txt abgelegt. Er bringt den Rechner automatisch in die Gruppe nb-it.*

Auftrag: Melde adm01 mit dem Setup-Key bei https://netbird.saarmetall.example an und entferne den Schlüssel danach vom Rechner.

☐ adm01 ist verbunden

☐ Angemeldet bei https://netbird.saarmetall.example

☐ adm01 ist in der Gruppe nb-it

☐ Der Setup-Key liegt nicht mehr auf dem Rechner

Worum es geht: Setup-Keys melden Geräte ohne Benutzerinteraktion an und weisen ihnen über auto_groups sofort die richtigen Gruppen zu. Sie sind so sensibel wie Passwörter: einmalig, mit kurzer Laufzeit und nach der Anmeldung vom System entfernen. Im Buch: Kapitel 5.2 und 8.3.

Mission 3: Wer fehlt?

Buch: Kapitel 4.4 Von: Monitoring (automatischer Alarm)

> *Peer-Zähler auf adm01 auffällig: Es sind weniger Peers verbunden als erwartet. Eine Gegenstelle antwortet nicht mehr.*

Auftrag: Finde mit netbird status heraus, welcher Peer offline ist, und melde seinen Namen. Bring ihn danach wieder online.

☐ Fehlenden Peer korrekt gemeldet

☐ zbx01 ist wieder verbunden

Die Antwort geben Sie im Spiel in das Antwortfeld ein, im Labor als zweiten Wert: `sudo saarmetall-lab check 3 ANTWORT`.

Worum es geht: Die Zeile Peers count ist eine schnelle Gesundheitsprüfung. Die Detailansicht nennt für jeden Peer Status, Verbindungsart und letzten Handshake. Bei Lazy Connections ist eine kleinere erste Zahl normal. Im Buch: Kapitel 4.4.

Mission 4: Setup-Keys

Mission 4: Setup-Key abgelehnt

Buch: Kapitel 8.3 Von: Monika Berger (Buchhaltung)

> *Mein Notebook wurde neu installiert und kommt seitdem nicht mehr ins Netz. Die IT hat mir gesagt, es meldet sich automatisch an. Tut es aber nicht. Ich komme nicht ins ERP.*

Auftrag: Finde heraus, warum sich nb-buchhaltung nicht anmelden kann. Lege einen neuen Einmal-Setup-Key mit höchstens 7 Tagen Laufzeit für die Gruppe nb-verwaltung an und melde das Notebook damit an.

- ☐ nb-buchhaltung ist verbunden
- ☐ nb-buchhaltung ist in der Gruppe nb-verwaltung
- ☐ Der verwendete Schlüssel ist einmalig und läuft in höchstens 7 Tagen ab
- ☐ Das ERP antwortet dem Notebook auf Port 443

Worum es geht: Der Server meldet bei abgelaufenen, verbrauchten und widerrufenen Schlüsseln dasselbe: setup key is invalid. Den Grund zeigt erst das Feld state in der API oder im Dashboard. Hier hatte jemand den Schlüssel beim Aufräumen widerrufen. Für Einzelgeräte ist ein Einmal-Key mit kurzer Laufzeit richtig. Wiederverwendbare Keys gehören nur in automatisierte Rollouts und brauchen ebenfalls ein Ablaufdatum. Im Buch: Kapitel 8.3.

Mission 5 und 6: Richtlinien

Mission 5: Default-Allow muss weg

Buch: Kapitel 9.1 bis 9.4 Von: Sabine Weiler (Informationssicherheit)

> *Beim Pentest konnte sich der Dienstleister-Zugang dl-cnc mit dem ERP-Server verbinden. Das darf nicht sein. Offenbar hat jemand die Default-Richtlinie wieder eingeschaltet und dafür „Verwaltung auf ERP“ gelöscht, weil „es dann einfach ging“.*

Auftrag: Schalte die Default-Richtlinie ab oder lösche sie und lege die Richtlinie „Verwaltung auf ERP“ wieder an: nb-verwaltung auf srv-erp, nur TCP 443, in eine Richtung.

- ☐ Keine aktive Default-Richtlinie mehr

☐ Die Buchhaltung erreicht das ERP auf TCP 443

☐ Die Buchhaltung erreicht das ERP nicht per SSH (TCP 22)

☐ dl-cnc erreicht das ERP nicht mehr

☐ Der Monitoring-Server erreicht seine Agenten weiterhin

Worum es geht: Die Default-Richtlinie erlaubt jedem Gerät alles. Sie ist für den ersten Test gedacht und muss weg, bevor produktive Systeme ins Netz kommen. Richtlinien verbinden Rollengruppen mit Ressourcengruppen, nach Protokoll, Port und Richtung. Im Buch: Kapitel 9.1 bis 9.4.

Mission 6: Zu viel für den ERP-Partner

Buch: Kapitel 9.3 Von: Sabine Weiler (Informationssicherheit)

> *Bei der Prüfung der Richtlinien ist mir „ERP-Dienstleister“ aufgefallen: alle Protokolle, in beide Richtungen. Laut Vertrag braucht der Partner nur Remotedesktop auf den ERP-Server. Und der Server soll niemals von sich aus Verbindungen zum Dienstleister aufbauen.*

Auftrag: Beschränke die Richtlinie „ERP-Dienstleister“ auf TCP 3389 und auf eine Richtung, ohne den Zugang des Partners zu unterbrechen.

☐ dl-erp erreicht erp01 per Remotedesktop (TCP 3389)

☐ dl-erp erreicht erp01 nicht auf anderen Ports

☐ erp01 kann keine Verbindung zu dl-erp aufbauen

Worum es geht: Die Richtung einer Regel bezieht sich auf den Verbindungsaufbau, nicht auf den Datenfluss. Antworten fließen immer zurück. Für Zugriffe von Personen auf Server ist die Richtung fast immer einseitig. Ports gehören so eng wie möglich gefasst. Im Buch: Kapitel 9.3.

Mission 7: Networks und Routing Peers

Mission 7: Fernwartung CNC

Buch: Kapitel 11 Von: Martin Scholl (IT-Leitung)

> *Der Hersteller der CNC-Steuerungen meldet, dass er nicht mehr auf cnc-01 (10.20.100.21) im Werk Völklingen kommt. Die Steuerung selbst hat keinen NetBird-Client, sie hängt hinter dem Routing Peer rp-vk-01. Nach dem*

> *Wartungsfenster am Wochenende ist irgendetwas liegen geblieben.*

Auftrag: Sorge dafür, dass dl-cnc die Steuerung cnc-01 über den Routing Peer auf Port 5900 erreicht, aber keine andere Maschine und keinen anderen Port.

- ☐ Der Routing Peer im Network „Werk Völklingen“ ist aktiv
- ☐ dl-cnc erreicht cnc-01 auf Port 5900
- ☐ dl-cnc erreicht cnc-01 nicht auf Port 80
- ☐ dl-cnc erreicht cnc-02 nicht

Worum es geht: Geräte ohne Client erreicht NetBird über Networks: Ein Routing Peer im Zielnetz leitet den Verkehr weiter, Ressourcen beschreiben die erlaubten Ziele als Adresse oder Bereich. Ohne aktiven Router und ohne passende Richtlinie geht nichts. Für Dienstleister gehört jede Maschine als eigene Ressource angelegt. Im Buch: Kapitel 11.

Mission 8: DNS im Overlay

Mission 8: Interne Namen

Buch: Kapitel 12 Von: Monika Berger (Buchhaltung)

> *Mit der Adresse komme ich ins ERP, aber der Link im Handbuch der Buchhaltung, erp01.intern.saarmetall.example, funktioniert nicht. Lena meint, seit der Domänencontroller ausgetauscht wurde.*

Auftrag: Repariere die Nameserver-Gruppe ns-intern, sodass nb-buchhaltung erp01.intern.saarmetall.example auflöst. Die Gruppe soll weiterhin nur für die interne Domain gelten.

- ☐ nb-buchhaltung löst erp01.intern.saarmetall.example auf
- ☐ ns-intern zeigt auf den Domänencontroller dc-sb-01
- ☐ ns-intern gilt nur für die interne Domain (Split DNS)

Worum es geht: Nameserver-Gruppen verteilen DNS-Server an die Peers, wahlweise für alle Anfragen oder nur für bestimmte Domains (Split DNS). Zeigt eine Gruppe auf eine Adresse, die es nicht mehr gibt, laufen Anfragen ins Leere. Der DNS-Server braucht außerdem eine Richtlinie, die UDP 53 erlaubt. Im Buch: Kapitel 12.2 und 12.3.

Mission 9: Posture Checks

Mission 9: Gesperrt durch den Posture Check

Buch: Kapitel 10.1 und 10.5 Von: Martin Scholl (IT-Leitung)

> *Seit heute früh kommt die gesamte Verwaltung nicht mehr ins ERP. Netzwerk und Server sind in Ordnung. Sabine hat gestern die Gerätechecks „verschärft". Bitte reparieren, aber den Check nicht einfach entfernen, er ist eine Auflage aus dem Audit.*

Auftrag: Finde den Posture Check, der die Verwaltung aussperrt, und setze die Mindestversion auf einen Wert, den die ausgerollten Clients erfüllen. Der Check bleibt an der Richtlinie.

- ☐ Die Buchhaltung erreicht das ERP wieder auf TCP 443
- ☐ „Verwaltung auf ERP" ist weiterhin an den Posture Check gebunden
- ☐ Die Mindestversion ist nicht niedriger als 0.70.0

Worum es geht: Posture Checks prüfen das Gerät, bevor eine Richtlinie greift. Eine Mindestversion über der ausgerollten Version sperrt alle aus. Wer verschärft, prüft vorher, welche Versionen im Feld sind, und kündigt Änderungen an. Im Buch: Kapitel 10.1 und 10.5.

Mission 10: Der fremde Peer

Mission 10: Der fremde Peer

Buch: Kapitel 18.1 und 19.8 Von: Sabine Weiler (Informationssicherheit)

> *Alarm aus der Auswertung des Aktivitätsprotokolls: Heute Nacht hat sich ein Peer angemeldet, den niemand kennt. Er ist in einer Servergruppe gelandet. Vermutlich ist ein Setup-Key aus einem alten Skript im Umlauf. Das ist ein Sicherheitsvorfall nach unserem Notfallplan.*

Auftrag: Finde den fremden Peer und den Setup-Key, mit dem er sich angemeldet hat. Entferne den Peer, widerrufe den Schlüssel und halte beides in ~/berichte/vorfall-netbird.txt fest (Peer-Name, Schlüsselname, Uhrzeit der Anmeldung).

- ☐ Der fremde Peer ist aus dem Konto entfernt
- ☐ Der Setup-Key server-saarbruecken-alt ist widerrufen oder gelöscht

☐ erp01 und die übrigen Server laufen weiter

☐ Der Vorfallbericht nennt Peer, Schlüssel und Uhrzeit

Worum es geht: Bei einem kompromittierten Schlüssel zählt die Reihenfolge: Spur sichern, Schlüssel widerrufen, damit keine weiteren Geräte nachkommen, fremde Peers entfernen, dann aufarbeiten. Wiederverwendbare Keys ohne Ablaufdatum sind das eigentliche Risiko. Zeiten im Aktivitätsprotokoll stehen in UTC. Im Buch: Kapitel 18.1 und 19.8.

KAPITEL 6

Wenn etwas nicht klappt

Symptom	Ursache und Abhilfe
Die Installation bricht beim Laden der Images ab	Die VM erreicht Docker Hub nicht. Internetzugang, Proxy und DNS der VM prüfen, dann `install` erneut aufrufen. Das Skript setzt dort fort, wo es abgebrochen ist.
`permission denied ... docker.sock`	Ihr Benutzer ist noch nicht in der Gruppe docker aktiv. Ab- und wieder anmelden oder `sudo docker ...` verwenden.
Ein Peer bleibt getrennt	`docker logs NAME` zeigt die Meldung des Clients. `sudo saarmetall-lab reset` meldet getrennte Peers neu an.
API antwortet mit 401	Token fehlt oder ist falsch eingefügt. `echo $TOKEN` muss mit nbp_ beginnen.
API antwortet mit 422	Ein Pflichtfeld fehlt oder eine Gruppe steht als Name statt als ID im Objekt. Die Fehlermeldung nennt das Feld.
Eine Änderung wirkt nicht sofort	Die Peers erhalten neue Richtlinien innerhalb weniger Sekunden. Zehn Sekunden warten, dann erneut prüfen.
Der Name netbird.saarmetall.example ist unbekannt	Der Eintrag in /etc/hosts fehlt. `install` ergänzt ihn wieder.

Tabelle 6.1: Häufige Probleme im Labor

Wenn gar nichts mehr geht: `sudo saarmetall-lab reset` stellt den Ausgangszustand her, ohne die Instanz neu aufzusetzen. Für einen vollständigen Neuanfang spielen Sie den Schnappschuss der VM zurück.

Versionshinweis

Das Labor verwendet NetBird 0.81.3. Neuere Versionen können Felder der API ergänzen oder Ausgaben von `netbird status` verändern. Die Prinzipien aus dem Buch bleiben gleich. Eine aktualisierte Fassung des Skripts steht im Downloadbereich bereit.

ANHANG A

Spickzettel REST-API

```
TOKEN=$(cat ~/.netbird-token)
API=https://netbird.saarmetall.example/api
H=(-H "Authorization: Token $TOKEN" -H "Content-Type: application/
json")
```

Aufgabe	Aufruf
Peers anzeigen	`curl -s "${H[@]}" $API/peers | jq '.[] | {name, ip, connected}'`
Peer löschen	`curl -s "${H[@]}" -X DELETE $API/peers/ID`
Gruppen-ID ermitteln	`curl -s "${H[@]}" $API/groups | jq -r '.[] | select(.name == "srv-erp") | .id'`
Setup-Keys anzeigen	`curl -s "${H[@]}" $API/setup-keys | jq '.[] | {name, type, state, expires}'`
Setup-Key anlegen	`curl -s "${H[@]}" -X POST $API/setup-keys -d '{"name":"...","type":"one-off","expires_in":86400,"auto_groups":["GID"],"usage_limit":1}'`
Setup-Key widerrufen	`curl -s "${H[@]}" -X PUT $API/setup-keys/ID -d '{"revoked":true,"auto_groups":[...]}'`
Richtlinien anzeigen	`curl -s "${H[@]}" $API/policies | jq '.[] | {name, enabled}'`
Richtlinie anlegen	`curl -s "${H[@]}" -X POST $API/policies -d @richtlinie.json`
Richtlinie ändern	`curl -s "${H[@]}" -X PUT $API/policies/ID -d @richtlinie.json`
Networks und Router	`$API/networks`, `$API/networks/ID/routers`, `$API/networks/ID/resources`
Nameserver-Gruppen	`$API/dns/nameservers`, ändern per PUT mit allen Feldern
Posture Checks	`$API/posture-checks`, ändern per PUT mit name, description und checks
Aktivitätsprotokoll	`curl -s "${H[@]}" $API/events/audit | jq '.[0:10]'`

Tabelle A.1: Die wichtigsten Aufrufe für die Missionen

Aufbau einer Richtlinie

```
{
  "name": "Verwaltung auf ERP",
  "description": "ERP-Weboberfläche für die Verwaltung",
  "enabled": true,
  "source_posture_checks": [],
  "rules": [{
    "name": "Verwaltung auf ERP",
    "enabled": true,
    "action": "accept",
    "bidirectional": false,
    "protocol": "tcp",
    "ports": ["443"],
    "sources": ["ID von nb-verwaltung"],
    "destinations": ["ID von srv-erp"]
  }]
}
```

Die Felder entsprechen dem Dialog unter Access Control, Policies im Dashboard (Kapitel 9.3 des Buchs). Die Zeiten im Aktivitätsprotokoll stehen in UTC.

ANHANG B

Lösungswege

Die folgenden Wege führen zum Ziel. Andere sind ebenso richtig, solange die Prüfung alle Ziele als erfüllt meldet. In den Beispielen sind `TOKEN`, `API` und `H` wie in Anhang A gesetzt.

Mission 1: Der Agent schläft

```
netbird status
systemctl status netbird
sudo systemctl enable --now netbird
netbird status
```

Mission 2: Anmelden am eigenen Management

```
sudo netbird up --management-url https://netbird.saarmetall.exampl
e \
  --setup-key "$(sudo cat /root/netbird-setup-key.txt)"
sudo rm /root/netbird-setup-key.txt
netbird status
```

Mission 3: Wer fehlt?

```
netbird status -d
docker ps -a
docker start zbx01
sudo saarmetall-lab check 3 zbx01
```

Mission 4: Setup-Key abgelehnt

```
docker logs nb-buchhaltung
curl -s -H "Authorization: Token $TOKEN" $API/setup-keys | jq '.[]
| {name, state}'
NV=$(curl -s -H "Authorization: Token $TOKEN" $API/groups | jq -r
'.[] | select(.name == "nb-verwaltung") | .id')
KEY=$(curl -s -X POST -H "Authorization: Token $TOKEN" $API/setup-
keys \
  -d "{\"name\":\"nb-buchhaltung\",\"type\":\"one-off\",\"expires_
in\":86400,\"auto_groups\":[\"$NV\"],\"usage_limit\":1}" | jq -r .
key)
docker exec nb-buchhaltung netbird up --setup-key $KEY
docker exec nb-buchhaltung nc -zv -w 3 erp01.netbird.selfhosted 44
3
```

Mission 5: Default-Allow muss weg

```
curl -s -H "Authorization: Token $TOKEN" $API/policies | jq '.[] |
{name, enabled}'
curl -s -H "Authorization: Token $TOKEN" $API/policies | jq '.[] |
select(.name == "Default")' > default.json
jq '{name, description, enabled, source_posture_checks, rules: [.r
ules[] | .sources |= map(.id) | .destinations |= map(.id)]} | .ena
bled = false' default.json > default-aus.json
curl -s -X PUT -H "Authorization: Token $TOKEN" -d @default-aus.js
on $API/policies/$(jq -r .id default.json)
V=$(curl -s -H "Authorization: Token $TOKEN" $API/groups | jq -r
'.[] | select(.name == "nb-verwaltung") | .id')
E=$(curl -s -H "Authorization: Token $TOKEN" $API/groups | jq -r
'.[] | select(.name == "srv-erp") | .id')
curl -s -X POST -H "Authorization: Token $TOKEN" $API/policies -d
"{\"name\":\"Verwaltung auf ERP\",\"enabled\":true,\"rules\":[{\"n
ame\":\"Verwaltung auf ERP\",\"enabled\":true,\"action\":\"accept
\",\"bidirectional\":false,\"protocol\":\"tcp\",\"ports\":[\"443
\"],\"sources\":[\"$V\"],\"destinations\":[\"$E\"]}]}"
docker exec nb-buchhaltung nc -zv -w 3 erp01.netbird.selfhosted 44
3
docker exec dl-cnc nc -zv -w 3 erp01.netbird.selfhosted 443
```

Mission 6: Zu viel für den ERP-Partner

```
curl -s -H "Authorization: Token $TOKEN" $API/policies | jq '.[] |
select(.name == "ERP-Dienstleister")' > p.json
jq '{name, description, enabled, source_posture_checks, rules: [.r
ules[] | .sources |= map(.id) | .destinations |= map(.id) | .proto
col = "tcp" | .ports = ["3389"] | .bidirectional = false]}' p.json
> neu.json
curl -s -X PUT -H "Authorization: Token $TOKEN" -d @neu.json $API/
policies/$(jq -r .id p.json)
docker exec dl-erp nc -zv -w 3 erp01.netbird.selfhosted 3389
docker exec dl-erp nc -zv -w 3 erp01.netbird.selfhosted 443
```

Mission 7: Fernwartung CNC

```
NET=$(curl -s -H "Authorization: Token $TOKEN" $API/networks | jq
-r '.[] | select(.name == "Werk Völklingen") | .id')
curl -s -H "Authorization: Token $TOKEN" $API/networks/$NET/router
s | jq .
curl -s -H "Authorization: Token $TOKEN" $API/networks/$NET/router
s | jq '.[0] | {peer, peer_groups, metric, masquerade, enabled: tr
ue}' > router.json
RT=$(curl -s -H "Authorization: Token $TOKEN" $API/networks/$NET/r
outers | jq -r '.[0].id')
curl -s -X PUT -H "Authorization: Token $TOKEN" -d @router.json $A
PI/networks/$NET/routers/$RT
S=$(curl -s -H "Authorization: Token $TOKEN" $API/groups | jq -r
'.[] | select(.name == "nb-dl-cnc") | .id')
D=$(curl -s -H "Authorization: Token $TOKEN" $API/groups | jq -r
'.[] | select(.name == "res-cnc-hersteller") | .id')
curl -s -X POST -H "Authorization: Token $TOKEN" $API/policies -d
"{\"name\":\"CNC-Fernwartung\",\"enabled\":true,\"rules\":[{\"name
\":\"CNC-Fernwartung\",\"enabled\":true,\"action\":\"accept\",\"bi
directional\":false,\"protocol\":\"tcp\",\"ports\":[\"5900\"],\"so
urces\":[\"$S\"],\"destinations\":[\"$D\"]}]}"
docker exec dl-cnc nc -zv -w 3 10.20.100.21 5900
docker exec dl-cnc nc -zv -w 3 10.20.100.22 5900
```

Mission 8: Interne Namen

```
docker exec nb-buchhaltung nslookup erp01.intern.saarmetall.exampl
e
DC=$(curl -s -H "Authorization: Token $TOKEN" $API/peers | jq -r
'.[] | select(.name == "dc-sb-01") | .ip')
curl -s -H "Authorization: Token $TOKEN" $API/dns/nameservers | jq
'.[] | select(.name == "ns-intern")' > ns.json
jq ".nameservers[0].ip = \"$DC\" | del(.id)" ns.json > ns-neu.json
curl -s -X PUT -H "Authorization: Token $TOKEN" -d @ns-neu.json $A
PI/dns/nameservers/$(jq -r .id ns.json)
docker exec nb-buchhaltung nslookup erp01.intern.saarmetall.exampl
e
```

Mission 9: Gesperrt durch den Posture Check

```
docker exec nb-buchhaltung netbird version
curl -s -H "Authorization: Token $TOKEN" $API/posture-checks | jq
.
PC=$(curl -s -H "Authorization: Token $TOKEN" $API/posture-checks
| jq -r '.[] | select(.name == "Client aktuell") | .id')
curl -s -X PUT -H "Authorization: Token $TOKEN" $API/posture-check
s/$PC -d '{"name":"Client aktuell","description":"Mindestversion d
es NetBird-Clients","checks":{"nb_version_check":{"min_versio
n":"0.80.0"}}}'
docker exec nb-buchhaltung nc -zv -w 3 erp01.netbird.selfhosted 44
3
```

Mission 10: Der fremde Peer

```
curl -s -H "Authorization: Token $TOKEN" $API/peers | jq -r '.[].n
ame'
curl -s -H "Authorization: Token $TOKEN" $API/events/audit | jq '.
[] | select(.activity_code == "peer.setupkey.add")' > vorfall-spu
r.json
R=$(curl -s -H "Authorization: Token $TOKEN" $API/peers | jq -r '.
[] | select(.name == "build-7f3a") | .id')
K=$(curl -s -H "Authorization: Token $TOKEN" $API/setup-keys | jq
-r '.[] | select(.name == "server-saarbruecken-alt") | .id')
curl -s -X PUT -H "Authorization: Token $TOKEN" $API/setup-keys/$K
-d '{"revoked":true,"auto_groups":[]}'
curl -s -X DELETE -H "Authorization: Token $TOKEN" $API/peers/$R
mkdir -p ~/berichte && nano ~/berichte/vorfall-netbird.txt
```

Der Bericht nennt den Peer `build-7f3a`, den Schlüssel `server-saarbruecken-alt` und die Uhrzeit der Anmeldung aus dem Aktivitätsprotokoll. Die Reihenfolge ist wichtig: erst die Spur sichern, dann den Schlüssel widerrufen, damit keine weiteren Geräte nachkommen, dann den Peer entfernen (Kapitel 19.8).