Schweizerische
Fachschule

TEKO

Linux Projekt

VPN-Gateway mit WireGuard

Verfasser:
Christian Rybovic

Studiengang:
Dipl. Informatiker HF

Klasse:
L-TIP-24-Do-a

Webseite:
neo.christianrybovic.ch

Datum der Abgabe:
05. Dezember 2025

Management Summary

In diesem Projekt wird ein sicheres Zugangssystem eingerichtet, mit dem Nutzer von aussen auf ein privates Netzwerk zugreifen können (zum Beispiel von unterwegs aus). Dazu wird als Serverbetriebssystem die Linux-Distribution Debian verwendet und als VPN-Gateway kommt WireGuard zum Einsatz.

Das VPN-Gateway stellt die Verbindung zwischen dem privaten Netzwerk und dem Internet her. Nur autorisierte Personen können über eine sichere Verbindung auf die gewünschten Daten oder Geräte im Netzwerk zugreifen.

Debian wurde ausgewählt, weil es sehr stabil, zuverlässig und sicher ist. WireGuard überzeugt durch seine hohe Geschwindigkeit, moderne Technik und einfache Verwaltung. Der Zugriff auf den Server erfolgt über eine Anmeldung mit Schlüsselpaaren statt Passwörtern.

Um den Schutz des Systems zu erhöhen, werden zusätzliche Sicherheitsmassnahmen umgesetzt:

- Änderung der Standard-Ports von WireGuard, um Angriffe zu erschweren
- Nur ausgewählte Zugriffe mithilfe von Firewall-Regeln erlauben
- Hardwareabsicherung durch die Deaktivierung von Bluetooth und WLAN im BIOS sowieso setzen eines BIOS-Passworts
- Absicherung des SSH-Zugangs durch Fail2Ban gegen automatische Login-Versuche

Ein einfaches Backup-Konzept sorgt dafür, dass alle wichtigen Einstellungen gesichert und bei einem Ausfall schnell wiederhergestellt werden können.

Das Projekt zeigt, wie sich ein sicherer und effizienter VPN-Dienst auf Basis freier Softwarelösungen realisieren lässt. Es bietet eine praxisnahe Grundlage für den sicheren Remote-Zugriff und verdeutlicht den professionellen Umgang mit Linux-Servern, Netzwerksicherheit und Systemadministration.

Inhaltsverzeichnis

1. Ausgangslage

In der bisherigen Infrastruktur wurden zur externen Erreichbarkeit verschiedener interner Dienste (darunter Überwachungskameras, ein Media-Streaming-Server, sowie weitere Systeme) mehrere Portfreigaben direkt auf dem Internetrouter eingerichtet. Dieses Vorgehen galt lange als gängige Praxis, birgt jedoch zunehmend sicherheitstechnische Risiken. Jede Portfreigabe erweitert die potenzielle Angriffsfläche und eröffnet Möglichkeiten für unautorisierte Zugriffe, Schwachstellenausnutzung oder automatisierte Angriffe aus dem Internet.

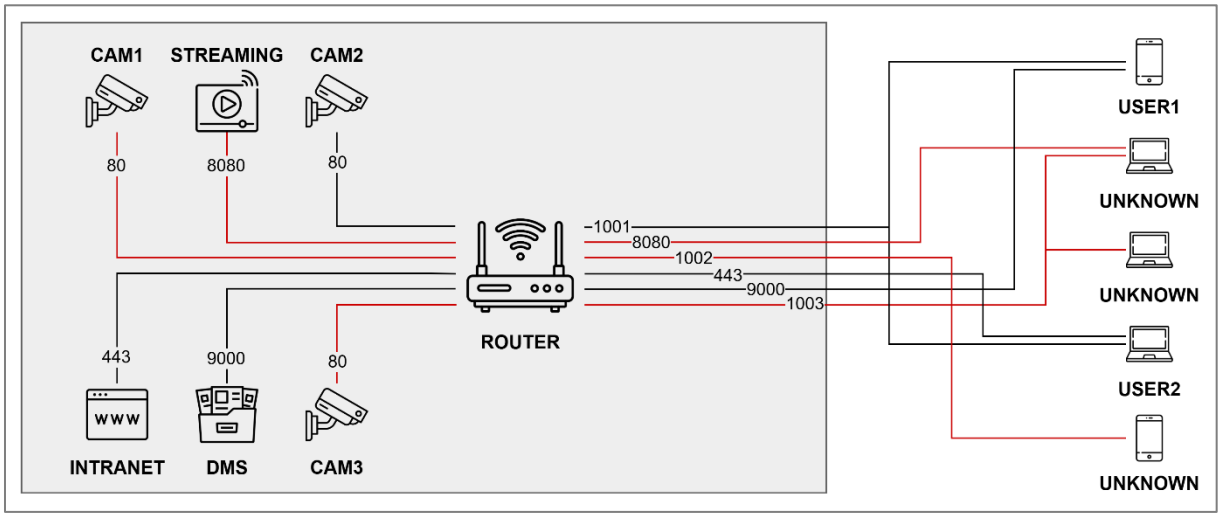

Abbildung 1: Netzwerk IST-Situation

Um diesen Risiken zu begegnen, soll ein zentrales VPN-Gateway eingeführt werden, das künftig als einzige öffentlich erreichbare Komponente fungiert. Sämtliche externen Zugriffe auf interne Dienste erfolgen dadurch ausschließlich über eine authentifizierte und verschlüsselte VPN-Verbindung. Die interne Netzwerkinfrastruktur bleibt vollständig vor direktem Internetzugang geschützt und auf dem Router muss lediglich ein einzelner Port für das VPN-Gateway offen bleiben.

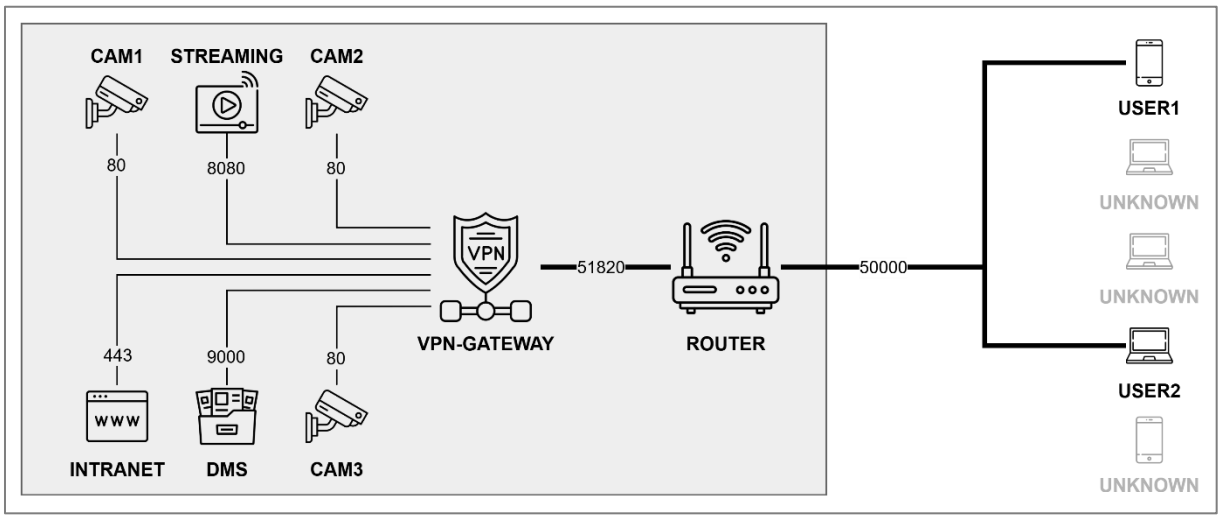

Abbildung 2: Netzwerk SOLL-Situation

Durch diese Umstellung wird die Angriffsfläche erheblich reduziert, die Zugangskontrolle verbessert und die Kommunikation insgesamt deutlich sicherer gestaltet.

2. Hardware

Als Hardware für das VPN-Gateway kommt ein **MINIX NEO Z100-0dB** zum Einsatz. Das Gerät überzeugt durch seine kompakte Bauweise, den geräuschlosen Betrieb dank passiver Kühlung, sowie einen geringen Stromverbrauch, was es ideal für den Dauerbetrieb macht.

Abbildung 3: Hardwaredetails

Trotz der kleinen Grösse bietet er ausreichend Leistung für ein VPN-Gateway. Der Intel N100 Quad-Core-Prozessor bietet genug Rechenleistung für verschlüsselten Netzwerkverkehr und parallele VPN-Tunnel. Zudem unterstützt das Gerät Features wie Wake-on-LAN und RTC-Wake, was eine flexible Automatisierung und Einbindung in bestehende Netzwerkumgebungen erleichtert.

Im BIOS des Gerätes wird vor der Installation noch ein Passwort gesetzt und WLAN sowie Bluetooth deaktiviert. Da diese nicht benötigt werden, kann mit der Deaktivierung gleichzeitig auch die Sicherheit des Systems erhöht werden.

Hersteller	MINIX
Modell	NEO Z100-0db
Type	NUC/Mini PC
CPU	Intel N100 3.4GHz
RAM	8GB DDR4-3200
Datenträger	256GB M.2 NVMe

Tabelle 1: Hardwarespezifikationen

Beim eingesetzten Router handelt es sich um eine FRITZ!Box 7590. Für den Zugriff auf das VPN-Gateway aus dem Internet, muss auf dem Router eine Portfreigabe eingerichtet werden. Hierfür wird der Port 50000 konfiguriert. Die Verwendung eines Nicht-Standard-Ports dient als zusätzliche Verschleierungsebene, damit automatisierte Angriffe, die auf Standardports zielen, nicht mehr funktionieren.

3. Betriebssystem

3.1. Wahl der Linux Distribution

Es gibt verschiedene Linux Distributionen, welche sich für dieses Projekt eigenen würden. Da das System rund um die Uhr im Einsatz steht, wurden bei der Auswahl ausschliesslich Serverbetriebssysteme berücksichtigt. Für die richtige Wahl wurden die vier Distributionen **Debian**, **openSUSE Leap**, **RHEL** und **Ubuntu Server** im Rahmen einer Nutzwertanalyse miteinander verglichen (siehe Anhang 1).

Bewertet wurden dabei acht Kriterien, wobei der Schwerpunkt klar auf den Bereichen Sicherheit, Stabilität, sowie geringer Ressourcenverbrauch lag. Diese Aspekte sind für ein VPN-Gateway besonders entscheidend, da das System direkt aus dem Internet zugänglich sein wird.

Das Ergebnis dieser Analyse zeigt, dass Debian durch seine konservative Update-Politik, hohe Stabilität und sehr schlanke Grundinstallation insgesamt die höchsten Werte erzielt und am geeignetsten für dieses Projekt ist.

3.2. Installation Debian

Über die Webseite von Debian kann das Image heruntergeladen werden. Zum Erstellen des bootfähigen USB-Sticks ist die Software Rufus zu empfehlen. Rufus ist OpenSource, unter GNU GPL lizensiert und benötigt keine Installation. In Rufus dann den USB-Stick und das ISO angeben und auf «Start» klicken.

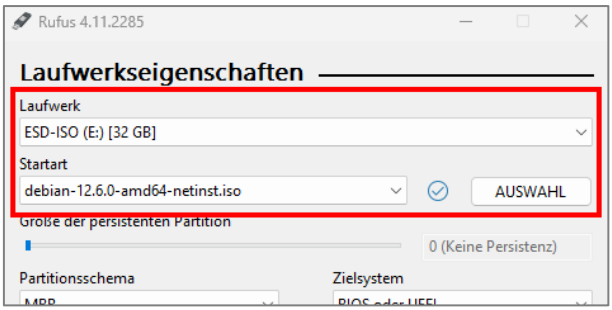

Abbildung 4: Bootstick mit Rufus erstellen

Wie auf der Abbildung zu erkennen ist, wird nicht das neuste Debian installiert, sondern die Version 12.6.0 von Juni 2024. Es wurde bewusst eine ältere Version verwendet, damit die Aktualisierung auf eine neuere Version auch praktisch getestet werden kann.

Nach dem Booten ab USB-Stick wird die Installation über den Menüpunkt "Install" gestartet. Um Übersetzungsfehler oder gar Softwarebugs aufgrund der ausgewählten Sprache zu vermeiden, wird das System auf Englisch installiert. So lassen sich dann beispielsweise bei Problemen auch Fehlermeldungen schneller im Internet finden.

Um die Einrichtung weiter nicht unnötig komplizierter zu machen, wurde bei der Installation bewusst nur ein einfaches Passwort gewählt - dieses wird vor dem produktiven Einsatz noch durch ein komplexes Passwort geändert. Die Installationsschritte im Detail können dem Anhang 2 entnommen werden.

3.3. Systemupdates und -upgrades

Aktualisierungen innerhalb der gleichen Version können mit dem Befehl `apt update && apt upgrade` durchgeführt werden. Dadurch werden zuerst die Paketliste und anschliessend die installierten Pakete aktualisiert. Nun kann das System mit `apt full-upgrade` auf den neusten Stand gebracht werden. Wenn in den Paketquellen unter "/etc/apt/source.list" überall "bookworm" durch "trixie" ersetz und diese mit `apt update` nochmal neu eingelesen werden, führt das System bei `apt upgrade –without-new-pkgs` ein minimales und mit `apt full-upgrade` ein vollständiges Upgrade auf Debian 13.2 durch.

4. Software

4.1. Wahl der VPN-Software

Auch für die Wahl der VPN-Software wurde eine Nutzwertanalyse erstellt (siehe Anhang 3). Hierbei wurden **strongSwan**, **OpenVPN** und **WireGuard** miteinander verglichen. Der Schwerpunkt wurde auch hier wieder auf Sicherheit und Performance gelegt.

WireGuard erzielte, unter Berücksichtigung der fünf Bewertungskriterien, das höchste Resultat. Die moderne Kryptographie, die geringe Codebasis und die sehr einfache Einrichtung reduzieren sowohl potenzielle Angriffsflächen als auch langfristige Betriebs- und Wartungskosten. Durch die gute Performance eignet sich WireGuard zudem besonders für Umgebungen mit vielen gleichzeitigen Clients oder hohen Durchsatzanforderungen.

4.2. Installation WireGuard

Die VPN-Software WireGuard kann mit dem Befehl `apt install wireguard` installiert werden. Um den Datenverkehr zwischen dem Netzwerk und VPN weiterleiten zu können, benötigt es noch IP-Forwarding. Dieses kann bei Debian im Kernel über "sysctl" aktiviert werden. Die Datei "/etc/sysctl.conf" sollte hierfür aber nicht direkt bearbeitet werden, da sie bei einem Systemupgrade geändert werden könnte oder auch den Upgradeprozess unterbricht, wenn nämlich gefragt wird, ob die aktuelle Konfigurationsdatei durch eine neuere Version überschrieben werden soll. Deshalb wird mit dem Befehl `echo 'net.ipv4.ip_forward=1' | tee -a /etc/sysctl.d/60-wireguard.conf` stattdessen eine neue Konfigurationsdatei mit der gewünschten Option unter "/etc/sysctl.d/" angelegt. Alle Konfigurationsdateien werden nach ihrem Dateinamen in lexikografischer Reihenfolge sortiert und ausgeführt. Wenn mehrere Dateien dieselbe Option angeben, hat der Eintrag in der Datei mit dem lexikografisch neuesten Namen Vorrang. Aus diesem Grund wird empfohlen, allen Dateinamen eine zweistellige Zahl und einen Bindestrich voranzustellen (um die Sortierung zu vereinfachen) und für Konfigurationsdateien im Ordner "/etc/" den Bereich 60-90 zu verwenden (um sicherzustellen, dass diese Dateien immer Vorrang vor den vom Betriebssystemhersteller mitgelieferten Konfigurationsdateien haben).

Die Anpassungen können mit dem Befehl `sysctl -p /etc/sysctl.d/60-wireguard.conf` oder einem Neustart des Systems übernommen werden. Das Installationsprotokoll befindet sich im Anhang 4. Sollte es einmal eine neuere Version geben, kann zuerst die Paketliste aktualisiert und anschliessend nochmal der Befehl für die Installation ausgeführt werden.

4.3. Konfiguration WireGuard

Im nächsten Schritt müssen Schlüssel generiert werden. Diese werden sowohl für die Authentifizierung als auch die Verschlüsselung genutzt. Der Server und alle Peers benötigen einen eigenen Schlüssel. Sollte ein Benutzer mehrere Geräte haben, dann sollte für jeden Endpunkt ein separater Schlüssel generiert werden. Ein Schlüsselpaar (bestehend aus einem Public- und einem Private-Key) kann mit dem Befehl `wg genkey | tee SERVER.PRIVATE.KEY | wg pubkey > SERVER.PUBLIC.KEY` im Ordner "/etc/wireguard" generiert werden.

WireGuard benötigt noch eine Konfigurationsdatei. In der neu erstellten Datei "wg0.conf" werden allgemeine Einstellungen sowie die Endpunkte angegeben (siehe Anhang 5). Die Einstellungen sind grundsätzlich selbsterklärend. Wichtig ist, dass alle generierten Schlüssel korrekt angegeben werden, da ansonsten kein VPN-Tunnel aufgebaut werden.

Sobald alle Schlüssel und die Konfigurationsdatei erstellt wurden, können noch die Dateiberechtigungen mit `chmod 600 *` eingeschränkt werden. So ist nur noch der User root berechtigt, die Dateien zu lesen oder zu bearbeiten. Und mit `systemctl enable wg-quick@wg0` wird WireGuard beim Systemstart automatisch mit unserer neuen Konfigurationsdatei ausgeführt.

4.4. Firewall

Als Firewall kommt nftables zum Einsatz. Im Vergleich zu anderen Firewalls wie ufw ist nftables etwas komplizierter aufgebaut, erlaubt aber eine flexiblere Gestaltung der Konfiguration.

In der Datei "/etc/nftables.conf" werden alle Firewall-Regeln hinterlegt (siehe Anhang 6). Um die Konfiguration zu vereinfachen, werden am Anfang der Datei zuerst drei Variablen erstellt. Anschliessend wird für WireGuard eine separate Forward-Chain "wg-forward" erstellt. Dadurch sind alle Regeln, welche WireGuard betreffen, zusammengefasst und von anderen Regeln sauber getrennt. Für jeden Peer (bezogen auf die IP-Adresse) können so unterschiedliche Regeln erstellt werden. Es können alle Zugriffe für die Peers individuell gesteuert werden. Für dieses Projekt gibt es im Moment nur zwei Peers, welche auch die gleichen Zugriffe erhalten. Um die Risiken weiter zu minimieren, werden nur immer exakt die Ports freigegeben, welche tatsächlich benötigt werden.

Auch kann bei nftables angegeben werden, was passieren soll, wenn eine Verbindung abgelehnt wird. Solche Anfragen können beispielsweise mit dem klassischen "host unreachable" beantwortet werden. Je nach dem um was für ein VPN-Gateway es sich handelt, kann es aber auch Sinn machen, die Anfragen mit einem "administratively prohibited" zu beantworten (im Problemfall ist so schneller klar, dass der Zugriff über die Firewall eingeschränkt wurde).

Da es sich beim aufgebauten VPN-Gateway grundsätzlich um ein eigenständiges Netzwerk handelt und der WireGuard-Anschluss nur virtuell ist, muss zum Schluss in der Postrouting-Chain noch "masquerade" aktiviert werden. Nur so ist ein Datenverkehr zwischen den Netzwerken erst möglich.

Die Firewall befindet sich standardmässig im deaktivieren Zustand. Mit dem Befehl `systemctl enable nftables` kann diese aktiviert und mit `systemctl start nftables` gestartet werden. Der Status kann mit `systemctl status nftables` überprüft werden.

4.5. Fail2Ban

Fail2Ban kann mit dem Befehl `apt install fail2ban` installiert werden. Da auch hier die Konfigurationsdatei nicht direkt angepasst werden darf, kann diese mit `cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local` wieder zuerst kopiert werden. In der Konfigurationsdatei werden einerseits die Sperrzeit und Fehlversuche sowie die "bannaction" und "bannaction_allports" auf "nftables" angepasst. Im Moment genügt es, nur SSH zu überwachen. Die Standardeinstellungen sind für dieses Projekt soweit in Ordnung. Für Debian muss aber in der Datei "paths-debian.conf" das "sshd_backend" noch ergänzt werden, sonst kann der Service nicht gestartet werden. Der aktuelle Status kann mit dem Befehl `fail2ban-client status sshd` angezeigt werden. Das Installationsprotokoll und die Konfigurationsdatei für Fail2Ban befinden sich im Anhang 7.

4.6. Passwortfreies SSH-Login

Für das passwortfreie SSH-Login benötigt man ein Schlüsselpaar. Das Schlüsselpaar kann mit dem Befehl `ssh-keygen -t rsa -b 4096` generiert werden (alternativ kann man das Schlüsselpaar auch direkt über PuTTY erstellen). Mit den Parametern kann man die Schlüsselstärke und weitere Einstellungen steuern. Der Standardpfad passt und eine Passphrase wird nicht gesetzt (so kann das Login auch für Automatisierungen einfach verwendet werden kann). Der private Schlüssel kann dann im PuTTY hinterlegt werden und wird so automatisch für das Login verwendet.

Der Public-Key kann nun mit `cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys` auf dem Server zu den berechtigten Authentifizierungsmethoden hinzugefügt werden. Standardmässig ist die Einstellung "PubkeyAuthentication" bei Debian bereits aktiviert (kann über die Konfigurationsdatei trotzdem noch explizit aktiviert werden), zusätzlich wird aber zur bessern Sicherheit noch "PasswordAuthentication" deaktiviert. Das gesamte Installationsprotokoll und die Konfigurationsdatei sind im Anhang 8 vorhanden.

5. Backup

Grundsätzlich sollte über ein Backup alles das gesichert werden, was im Zuge der Inbetriebnahme entweder angepasst oder generiert wurde. Auf ein automatisiertes und komplexes Backup-Konzept wird in diesem Projekt jedoch verzichtet - es werden keine Dateien zur Laufzeit automatisch angepasst und Anpassungen generell sind nur selten. Die Sicherung sollte bei jeder Anpassung gemacht werden, damit keine Konfigurationen verloren gehen.

Mit dem Befehl `tar -cJf backup.tar.xz /etc/sysctl.d/60-wireguard.conf /etc/wireguard/ /etc/nftables.conf /etc/fail2ban/jail.conf /etc/fail2ban/paths-debian.conf` werden daher alle notwendigen Dateien in ein TAR-Archiv gepackt, sodass diese bei einem Ausfall auf einem Ersatzsystem einfach und schnell wieder entpackt werden können. Die Dateien werden beim Packen komprimiert (wenn auch diese nicht sehr gross sind). Unter Beibehaltung des Pfades werden somit folgende Dateien gesichert:

Pfad	Datei	Beschreibung
/etc/	nftables.conf	Firewall-Konfigurationen für nftables
/etc/fail2ban/	jail.conf	Konfiguration für Fail2Ban
/etc/fail2ban/	paths-debian.conf	Debianspezifische Konfiguration für Fail2Ban
/etc/sysctl.d/	60-wireguard.conf	IP-Forwarding-Konfiguration für WireGuard
/etc/wireguard/	wg0.conf	Konfigurationsdatei für WireGuard
/etc/wireguard/	SERVER.PRIVATE.KEY	Privater Schlüssel des Servers
/etc/wireguard/	SERVER.PUBLIC.KEY	Öffentlicher Schlüssel des Servers
/etc/wireguard/	USER01.PRIVATE.KEY	Privater Schlüssel vom Benutzer 1
/etc/wireguard/	USER01.PUBLIC.KEY	Öffentlicher Schlüssel vom Benutzer 1
/etc/wireguard/	USER02.PRIVATE.KEY	Privater Schlüssel vom Benutzer 2
/etc/wireguard/	USER02.PUBLIC.KEY	Öffentlicher Schlüssel vom Benutzer 2

Tabelle 2: Dateien für Backup

Nach der Installation des Betriebssystems erfordert eine Wiederherstellung folgende Schritte:

1. Benötigte Software mit `apt -y install wireguard fail2ban` installieren
2. Mit `tar -xJf backup.tar.xz -C /` die Konfigurationen wiederherstellen
3. WireGuard und Firewall mit `systemctl enable wg-quick@wg0` und `systemctl enable nftables` aktivieren
4. Server mit `systemctl reboot` neustarten

Die Widerherstellung dauerte (ohne Betriebssystem) nur wenige Minuten, bis dieses wieder vollständig betriebsbereit ist. Was aus Sicherheitsgründen bewusst nicht gesichert wurde, sind die Schlüssel für die passwortfreie Authentifizierung und entsprechend auch die Konfigurationen für SSH.

6. Reflexion

Die Umsetzung des VPN-Gateways unter Debian mit WireGuard war eine sehr lehrreiche Erfahrung, die mir sowohl technische als auch organisatorische Erkenntnisse gebracht hat. Ein besonders kritischer Punkt war die Konfiguration der Firewall. Es ist hierbei extrem wichtig darauf zu achten, dass man sich nicht selbst aussperrt, beispielsweise indem man vergisst, den SSH-Port freizugeben.

Die Wahl von nftables anstelle von einfacheren Tools wie ufw stellte zunächst eine Herausforderung dar. Nftables wirkte zunächst umständlich und komplex, insbesondere wegen der detaillierten Syntax und der Vielzahl an Konfigurationsmöglichkeiten. Gleichzeitig bietet nftables jedoch auch eine deutlich größere Flexibilität, was zukünftige Erweiterungen und Anpassungen betrifft. Nach einer Eingewöhnungsphase wurde klar, dass der Mehraufwand gerechtfertigt ist, da man damit langfristig eine robustere und skalierbarere Firewall-Architektur erhält.

Ein weiteres Lernfeld war das Verständnis von WireGuard selbst. Das Konzept, dass Public- und Private-Keys gleichzeitig für Authentifizierung und Verschlüsselung genutzt werden, ist anfangs etwas ungewohnt und kann verwirrend wirken (auch welcher Key nun genau wo hinterlegt werden soll). Im Gegensatz zu klassischen VPN-Lösungen mit getrennten Passwörtern oder Zertifikaten muss man hier die Schlüsselverwaltung genau verstehen. Sobald der Aufbau klar war, liess sich WireGuard jedoch effizient konfigurieren und überzeugte durch seine einfache und schnelle Verbindungstechnik.

Was die Features betrifft, zeigt sich, dass WireGuard im Vergleich zu anderen Lösungen relativ minimalistisch ist. Besonders eine grafische Oberfläche für die Benutzerverwaltung fehlt. Eine solche Oberfläche würde die Verwaltung von Benutzern deutlich vereinfachen, gerade in größeren Umgebungen. Allerdings ist unklar, ob eine sichere, offiziell unterstützte Lösung existiert, sodass man derzeit auf manuelle Konfiguration über Konfigurationsdateien angewiesen ist.

Insgesamt bin ich mit der Umsetzung meines Projekts sehr zufrieden. Die Arbeit verlief weitgehend nach Plan, und das Ergebnis entspricht meinen Erwartungen. Etwas, das ich beim nächsten Mal anders handhaben würde, ist die Zeitplanung, insbesondere für die Dokumentation. Gerade das sorgfältige Festhalten von Schritten, Problemen und Lösungen in einer benutzerfreundlichen Formatierung sollte man nicht unterschätzen, da es viel Zeit in Anspruch nehmen kann.

Abschliessend lässt sich sagen, dass dieses Projekt nicht nur meine praktischen Fähigkeiten in Linux-Systemadministration, Firewall-Konfiguration und VPN-Technologien gestärkt hat, sondern auch ein besseres Verständnis für Sicherheitsaspekte und die langfristige Wartbarkeit von Systemen vermittelt hat.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BIOS ..... Basic Input/Output System
CPU ..... Central Processing Unit
RAM ..... Random-Access Memory
TAR ..... Tape Archive/Tarball
USB ..... Universal Serial Bus
VPN ..... Virtual Private Gateway

Quellenverzeichnis

Icons in den Grafiken von Freepik. (2. Dezember 2025). Von https://flaticon.com abgerufen

Anhang 1: Nutzwertanalyse Linux Distributionen

			Debian		openSUSE Leap		RHEL		Ubuntu Server	
Nr.	**Bewertungskriterium**	**Gewichtung [%]**	**Bewertung**	**Punkte**	**Bewertung**	**Punkte**	**Bewertung**	**Punkte**	**Bewertung**	**Punkte**
1	Stabilität (Langzeitbetrieb)	20	5	1.00	4	0.80	5	1.00	4	0.80
2	Release-/Support-Zyklus	10	5	0.50	4	0.40	5	0.50	4	0.40
3	Sicherheitskonzept (SELinux/AppArmor, Patches)	20	4	0.80	4	0.80	5	1.00	4	0.80
4	Paketverfügbarkeit	10	5	0.50	4	0.40	4	0.40	5	0.50
5	Minimal installierbar / geringer Ressourcenverbrauch	20	5	1.00	3	0.60	4	0.80	4	0.80
6	Kompatibilität/Hardware-Support	5	4	0.20	4	0.20	5	0.25	5	0.25
7	Community & Dokumentation	10	5	0.50	4	0.40	4	0.40	5	0.50
8	Administration/Benutzerfreundlichkeit	5	3	0.15	5	0.25	4	0.20	4	0.20
	Nutzwert	**100**		**4.65**		**3.85**		**4.55**		**4.25**

Bewertungsskala: 1 (schlecht) – 5 (sehr gut)

Anhang 2: Debian Installationsprotokoll

```
Select a language:
        Language: English

Select your location:
        Country, territory or area: other
        Continent or region: Europe
        Country, territory or area: Switzerland

Configure locales:
        Country to base default locale settings on: United States

Configure the keyboard:
        Keymap to use: Swiss German

Configure the network:
        Hostname: SV-VPN-001
        Domain name: [NONE]

Set up users and passwords:
        Root password: [NONE]
        Re-enter password to verify: [NONE]
        Full name for the new user: alpha
        Username for your account: alpha
        Choose a password for the new user: Luzern6000
        Re-enter password to verify: Luzern6000

Partition disks:
        Partitioning method: Guided - use entire disk
        Select disk to partition: SCSI1 (0,0,0) (sda)
        Partitioning scheme: All files on one partition
        Finish partitioning and write changes to disk
        Write the changes to disks? Yes

Configure the package manager:
        Scan extra installation media? No
        Debian archive mirror country: Switzerland
        Debian archive mirror: deb.debian.org
        HTTP proxy information: [NONE]

Configuring popularity-contest:
        Participate in the package usage survey? No

Software selection:
        Choose software to install:
                [*] SSH server
                [*] standard system utilities

Configuring grub-pc:
        Install the GRUB boot loader to your primary drive? Yes
        Device for boot loader installation: /dev/sda

Installation complete
```

Anhang 3: Nutzwertanalyse VPN-Software

			strongSwan		OpenVPN		WireGuard	
Nr.	**Bewertungskriterium**	**Gewichtung [%]**	**Bewertung**	**Punkte**	**Bewertung**	**Punkte**	**Bewertung**	**Punkte**
1	Performance	30	4	1.20	3	0.90	5	1.50
2	Sicherheit	30	3	0.90	5	1.50	5	1.50
3	Administrationsaufwand	10	2	0.20	3	0.30	4	0.40
4	Kompatibilität	20	4	0.80	5	1.00	4	0.80
5	Komplexität	10	2	0.20	3	0.30	4	0.40
	Nutzwert	**100**		**3.30**		**4.00**		**4.60**

Bewertungsskala: 1 (schlecht) – 5 (sehr gut)

Anhang 4: WireGuard Installationsprotokoll

```
root@SV-VPN-001:/home/alpha# apt install wireguard
Reading package lists... 0%
Reading package lists... 0%
Reading package lists... 100%
Reading package lists... Done
Building dependency tree... 0%
Building dependency tree... 50%
Building dependency tree... Done
Reading state information... 0%
Reading state information... Done
The following additional packages will be installed:
  wireguard-tools
Suggested packages:
  openresolv | resolvconf
The following NEW packages will be installed:
  wireguard wireguard-tools
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 95.8 kB of archives.
After this operation, 346 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
0% [Working]
Get:1 http://deb.debian.org/debian bookworm/main amd64 wireguard-tools amd64 1.0.20210914-1+b1 [87.6 kB]
3% [1 wireguard-tools 4,086 B/87.6 kB 5%]
83% [Working]
Get:2 http://deb.debian.org/debian bookworm/main amd64 wireguard all 1.0.20210914-1 [8,216 B]
87% [2 wireguard 4,696 B/8,216 B 57%]
100% [Working]
Fetched 95.8 kB in 0s (335 kB/s)
Selecting previously unselected package wireguard-tools.
(Reading database ...
(Reading database ... 20%
(Reading database ... 40%
(Reading database ... 60%
(Reading database ... 80%
(Reading database ... 100%
(Reading database ... 33120 files and directories currently installed.)
Preparing to unpack .../wireguard-tools_1.0.20210914-1+b1_amd64.deb ...
Progress: [  0%] [..........................................................] Progress: [ 11%] [######.....................................................] Unpacking wireguard-tools (1.0.20210914-1+b1) ...
Progress: [ 22%] [#############..............................................] Selecting previously unselected package wireguard.
Preparing to unpack .../wireguard_1.0.20210914-1_all.deb ...
Progress: [ 33%] [####################.....................................] Unpacking wireguard (1.0.20210914-1) ...
Progress: [ 44%] [#########################................................] Setting up wireguard-tools (1.0.20210914-1+b1) ...
Progress: [ 56%] [###############################..........................] wg-quick.target is a disabled or a static unit, not starting it.
Progress: [ 67%] [########################################.................] Setting up wireguard (1.0.20210914-1) ...
Progress: [ 78%] [################################################.........] Progress: [ 89%] [#####################################################.....] Processing triggers for man-db (2.11.2-2) ...
root@SV-VPN-001:/home/alpha# echo 'net.ipv4.ip_forward=1' | tee -a /etc/sysctl.d/60-wireguard.conf
net.ipv4.ip_forward=1
root@SV-VPN-001:/home/alpha# sysctl -p /etc/sysctl.d/60-wireguard.conf
net.ipv4.ip_forward = 1
```

/etc/sysctl.d/60-wireguard.conf

```
net.ipv4.ip_forward=1
```

Anhang 5: WireGuard Konfigurationsdateien

/etc/wireguard/wg0.conf

```
[Interface]
Address    = 10.0.200.1/32
SaveConfig = true
ListenPort = 51820
PrivateKey = WDzKo+U+qufiTnt9IkvlqOY4VB0jY91rb7Ibt+T1nk4=

[Peer]
PublicKey  = YbsAcHQoww4aC25cleh03ojN6B/DHQRlwTxF4mXX8T8=
AllowedIPs = 10.0.200.101/32

[Peer]
PublicKey  = 0Z2EhPrOZO7Z9z/wBXw683kWvcf2sfVSpN7vOTbhPz0=
AllowedIPs = 10.0.200.102/32
```

/etc/wireguard/SERVER.PUBLIC.KEY

```
HtQ3RJ4gSYvgNbeHU2LkUTPrOjjVT5gRY+EGuULerlQ=
```

/etc/wireguard/SERVER.PRIVATE.KEY

```
WDzKo+U+qufiTnt9IkvlqOY4VB0jY91rb7Ibt+T1nk4=
```

/etc/wireguard/USER01.PUBLIC.KEY

```
YbsAcHQoww4aC25cleh03ojN6B/DHQRlwTxF4mXX8T8=
```

/etc/wireguard/USER01.PRIVATE.KEY

```
wJiL7tAxIc+Q2dYmqZPox2jofX6CvyQAqviDQfJtLU4=
```

/etc/wireguard/USER02.PUBLIC.KEY

```
0Z2EhPrOZO7Z9z/wBXw683kWvcf2sfVSpN7vOTbhPz0=
```

/etc/wireguard/USER02.PRIVATE.KEY

```
cL6ePR9Gq+8xygBRjp1Z7f+4ZmQRJtartG86bHZvLkU=
```

Anhang 6: Firewall-Konfiguration

/etc/nftables.conf

```
#!/usr/sbin/nft -f
flush ruleset

define pub_iface = "enp1s0"
define wg_port   = 51820
define wg_iface  = "wg0"

table inet filter {
    chain input {
        type filter hook input priority 0; policy drop;
        # accept all loopback packets
        iif "lo" accept
        # accept all icmp/icmpv6 packets
        meta l4proto { icmp, ipv6-icmp } accept
        # accept all packets that are part of an already-established connection
        ct state vmap { invalid : drop, established : accept, related : accept }
        # drop new connections over rate limit
        ct state new limit rate over 1/second burst 10 packets drop
        # accept all DHCPv6 packets received at a link-local address
        ip6 daddr fe80::/64 udp dport dhcpv6-client accept
        # accept all SSH packets received on a public interface
        iifname $pub_iface tcp dport ssh accept
        # accept all WireGuard packets received on a public interface
        iifname $pub_iface udp dport $wg_port accept

        reject
    }
    chain forward {
        type filter hook forward priority 0; policy drop;
        # forward all packets that are part of an already-established connection
        ct state vmap { invalid : drop, established : accept, related : accept }
        # filter all packets from WireGuard VPN to Site B via wg-forward chain
        iifname $wg_iface oifname $pub_iface goto wg-forward
        reject with icmpx type host-unreachable
    }
    chain wg-forward {
        # forward all icmp/icmpv6 packets
        meta l4proto { icmp, ipv6-icmp } accept

        # forward all packets for endpoints and ports
        ip saddr 10.0.200.101 ip daddr 10.0.10.14 tcp dport 9000 accept  # DMS
        ip saddr 10.0.200.101 ip daddr 10.0.10.31 tcp dport 80 accept    # CAM1
        ip saddr 10.0.200.101 ip daddr 10.0.10.32 tcp dport 80 accept    # CAM2
        ip saddr 10.0.200.101 ip daddr 10.0.10.33 tcp dport 80 accept    # CAM3
        ip saddr 10.0.200.101 ip daddr 10.0.10.67 tcp dport 443 accept   # INTRANET
        ip saddr 10.0.200.101 ip daddr 10.0.10.70 tcp dport 8080 accept  # STREAMING

        ip saddr 10.0.200.102 ip daddr 10.0.10.14 tcp dport 9000 accept  # DMS
        ip saddr 10.0.200.102 ip daddr 10.0.10.31 tcp dport 80 accept    # CAM1
        ip saddr 10.0.200.102 ip daddr 10.0.10.32 tcp dport 80 accept    # CAM2
        ip saddr 10.0.200.102 ip daddr 10.0.10.33 tcp dport 80 accept    # CAM3
        ip saddr 10.0.200.102 ip daddr 10.0.10.67 tcp dport 443 accept   # INTRANET
        ip saddr 10.0.200.102 ip daddr 10.0.10.70 tcp dport 8080 accept  # STREAMING

        # reject with polite "administratively prohibited" icmp response
        reject with icmpx type admin-prohibited
    }
}

table inet nat {
    chain postrouting {
        type nat hook postrouting priority 100; policy accept;
        # masquerade all packets from WireGuard
        iifname $wg_iface oifname $pub_iface masquerade
    }
}
```

Anhang 7: Fail2Ban Installationsprotokoll und Konfigurationsdateien

```
root@SV-VPN-001:/home/alpha# apt install fail2ban
Reading package lists... 0%
Reading package lists... 100%
Reading package lists... Done
Building dependency tree... 0%
Building dependency tree... 50%
Building dependency tree... Done
Reading state information... 0%
Reading state information... 0%
Reading state information... Done
The following additional packages will be installed:
  python3-pyinotify python3-systemd whois
Suggested packages:
  mailx system-log-daemon monit sqlite3 python-pyinotify-doc
The following NEW packages will be installed:
  fail2ban python3-pyinotify python3-systemd whois
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 589 kB of archives.
After this operation, 2,901 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
0% [Working]
0% [Working]
Get:1 http://deb.debian.org/debian bookworm/main amd64 fail2ban all 1.0.2-2 [451 kB]
2% [1 fail2ban 12.3 kB/451 kB 3%]
66% [Working]
Get:2 http://deb.debian.org/debian bookworm/main amd64 python3-pyinotify all 0.9.6-2 [27.4 kB]
67% [2 python3-pyinotify 1,362 B/27.4 kB 5%]
75% [Waiting for headers]
Get:3 http://deb.debian.org/debian bookworm/main amd64 python3-systemd amd64 235-1+b2 [39.3 kB]
77% [3 python3-systemd 11.4 kB/39.3 kB 29%]
85% [Waiting for headers]
Get:4 http://deb.debian.org/debian bookworm/main amd64 whois amd64 5.5.17 [70.8 kB]
86% [4 whois 6,734 B/70.8 kB 10%]
100% [Working]
Fetched 589 kB in 5s (108 kB/s)
Selecting previously unselected package fail2ban.
(Reading database ...
(Reading database ... 20%
(Reading database ... 40%
(Reading database ... 60%
(Reading database ... 80%
(Reading database ... 100%
(Reading database ... 33201 files and directories currently installed.)
Preparing to unpack .../fail2ban_1.0.2-2_all.deb ...
Progress: [  0%] [..................................................] Progress: [  6%]
[###...............................................] Unpacking fail2ban (1.0.2-2) ...
Progress: [ 12%] [######............................................] Selecting previously
unselected package python3-pyinotify.
Preparing to unpack .../python3-pyinotify_0.9.6-2_all.deb ...
Progress: [ 18%] [##########........................................] Unpacking python3-
pyinotify (0.9.6-2) ...
Progress: [ 24%] [#############.....................................] Selecting previously
unselected package python3-systemd.
Preparing to unpack .../python3-systemd_235-1+b2_amd64.deb ...
Progress: [ 29%] [#################.................................] Unpacking python3-
systemd (235-1+b2) ...
Progress: [ 35%] [####################..............................] Selecting previously
unselected package whois.
Preparing to unpack .../whois_5.5.17_amd64.deb ...
Progress: [ 41%] [########################..........................] Unpacking whois
(5.5.17) ...
Progress: [ 47%] [############################......................] Setting up whois
(5.5.17) ...
Progress: [ 53%] [################################..................] Progress: [ 59%]
[###################################...............] Setting up fail2ban (1.0.2-2) ...
Progress: [ 65%] [#######################################...........] Created symlink
/etc/systemd/system/multi-user.target.wants/fail2ban.service
/lib/systemd/system/fail2ban.service.
Progress: [ 71%] [###############################################...] Setting up python3-
pyinotify (0.9.6-2) ...
Progress: [ 76%] [##################################################.....] Progress: [ 82%]
```

```
[################################################..........] Setting up python3-systemd (235-1+b2)
...
Progress: [ 88%] [#################################################.......] Progress: [ 94%]
[##################################################....] Processing triggers for man-db
(2.11.2-2) ...
root@SV-VPN-001:/home/alpha# cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
root@SV-VPN-001:/home/alpha# echo 'sshd_backend = systemd' >> /etc/fail2ban/paths-debian.conf
root@SV-VPN-001:/home/alpha# systemctl restart fail2ban
```

/etc/fail2ban/jail.conf

```
...

# "bantime" is the number of seconds that a host is banned.
bantime  = 30m

# A host is banned if it has generated "maxretry" during the last "findtime"
# seconds.
findtime  = 10m

# "maxretry" is the number of failures before a host get banned.
maxretry = 3

...

# Default banning action (e.g. iptables, iptables-new,
# iptables-multiport, shorewall, etc) It is used to define
# action_* variables. Can be overridden globally or per
# section within jail.local file
banaction = nftables
banaction_allports = nftables

...
```

/etc/fail2ban/paths-debian.conf

```
# Debian

[INCLUDES]

before = paths-common.conf

after  = paths-overrides.local


[DEFAULT]

syslog_mail = /var/log/mail.log

# control the `mail.warn` setting, see `/etc/rsyslog.d/50-default.conf` (if commented `mail.*`
wins).
# syslog_mail_warn = /var/log/mail.warn
syslog_mail_warn = %(syslog_mail)s

syslog_user  =  /var/log/user.log

syslog_ftp = /var/log/syslog

syslog_daemon =  /var/log/daemon.log

exim_main_log = /var/log/exim4/mainlog

# was in debian squeezy but not in wheezy
# /etc/proftpd/proftpd.conf (SystemLog)
proftpd_log = /var/log/proftpd/proftpd.log

roundcube_errors_log = /var/log/roundcube/errors.log
sshd_backend = systemd
```

Anhang 8 : Pubkey Authentication

```
alpha@SV-VPN-001:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alpha/.ssh/id_rsa):
Created directory '/home/alpha/.ssh'.

Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alpha/.ssh/id_rsa
Your public key has been saved in /home/alpha/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:mHbgtCmyehmy881yGYJ1CnzlRCYK3F/thgWs43AGySU alpha@SV-VPN-001
The key's randomart image is:
+---[RSA 4096]----+
|o oE++..o        |
|...=+o o o       |
|..  * = +        |
|...o.X B o       |
| +oo* O S        |
|o +o.+ .         |
| o.+ o           |
|o.+oo            |
|.+.oo            |
+----[SHA256]-----+
alpha@SV-VPN-001:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

/home/alpha/authorized_keys

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDdEK/OEopKL6EBXTn3eesN45GuciS/vuJBBZB7lwkW6LeHM2tQdn2Ub7PPHtAFk7z3KR
b5BZMscjjCJHEZy9WH8BLHdJOCpausnj/i5J1YQBPGA/zL3pO17M0pkFy4BP2hHBinHN/rwuRDIkluPudgdeFGxLtVE2WaPjvX
z3iROsCK7P/de4e46QXoxqAfzAyB1cEvZCuG8YCtU1AuR64RnuAyXPCr5IA67QqDG0tz2BK8wDyOP8mfNHRw0Wr64ZjQCuVtaV
bZmnv6ReJgIsLmGIEVlT+sR8+NLbeJmJSv/Coe+zpZ84k1i6NmwmtbEhPqDsJSJHwpOFzw7MLd5u0qRLZp2rXVDYIAaDLzZOix
fPjlPBUXXXld9oT1iwIIHLaxr1GVS0GhZV8LzvyXdOjHfJbW0A6TPJ/xhZ7K6NezKAycxoSFiJXfXNTJ8pWwiNzwI/KQ1JiXKz
QUrlWghKDwWY7td3KQjkZrgN85ZT3mfPaMdl2WGeDkUqMEX+UIls5VGBtIUaR9ihdogFVeNyEZYXit2S4jjLVXnCyziPr4Hu2S
zRDOdKqesp11uksl5d9hn6YVIb27XCvoJJ2eUFInSZTHb3bCtUApTqPNAYjvcUH/u8rmcZYCZdARaXiTfyFPmLkqBElBJN7B2L
6Og9IbsiJ+sExA/YT1vqc+zyMccw== alpha@SV-VPN-001
```

/etc/ssh/sshd_config

```
...

PubkeyAuthentication yes

...

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no

...
```