HTBLuVA Wiener Neustadt
Höhere Lehranstalt für Informatik

NVS PROJEKT

WireGuard

Ausgeführt im Schuljahr 2019/20 von:

WireGuard für Unterrichtseinsatz aufbereiten
Karlo PERANOVIC 5BHIF-18

Lehrer:

Dipl.-Ing. Dr. Günter Kolousek

Wiener Neustadt, am 11. April 2020

Abgabevermerk: Übernommen von:

Inhaltsverzeichnis

Kapitel 1

WireGuard

Abbildung 1.1: WireGuard Logo

1.1 Einführung

WireGuard ist ein extrem einfaches und dennoch schnelles und modernes VPN-Protkoll, welches eine sichere Lösung für das VPN-Tunneling bieten soll. Es ist darauf ausgelegt, leistungsfähiger, einfacher und nützlicher als die Konkurrenz z.B. IPsec, OpenVPN zu sein. WireGuard ist als Allzweck-VPN konzipiert, das sowohl auf eingebetteten Schnittstellen als auch auf Supercomputern ausgeführt werden kann und für viele verschiedene Umstände geeignet ist.

Ursprünglich wurde WireGuard für den Linux-Kernel veröffentlicht, ist jedoch nun plattformübergreifend (Windows, MacOS, BSD, iOS, Android) weitgehend einsetzbar. Derzeit wird WireGuard stark weiterentwickelt, aber kann jetzt schon als die sicherste, benutzerfreundlichste und einfachste VPN-Lösung in der Branche angesehen werden.

1.2 Installation

WireGuard kann wie in Abschnitt 1.1 beschrieben, auf vielen Betriebssystemen eingesetzt werden. Die Installation wird in weiterer Folge für das Betriebssystem Linux erklärt.

Unter Ubuntu $\geq$ 19.10 erfolgt die Installation durch:

```
1 $ sudo apt install wireguard
```

Ubuntu $\leq$ 19.04:

```
1 $ sudo add-apt-repository ppa:wireguard/wireguard
2 $ sudo apt-get update
3 $ sudo apt-get install wireguard
```

Debian:

```
1 # apt install wireguard
```

Arch:

```
1 $ sudo pacman -S wireguard-tools
```

1.3 Verwendung

Im folgenden Abschnitt werde ich auf alle Schritte eingehen, die zur Installation von WireGuard notwendig waren. Klar, es gibt immer mehrere Wege zu einem Ziel.

1.3.1 Vorbereitung

Um eine sinnvolle Funktionalität von WireGuard zu demonstrieren ist sowohl ein Server, als auch ein Client notwendig. Dazu habe ich in der Oracle Virtualbox zwei virtuelle Maschinen aufgesetzt. Auf einer VM lief ein Ubuntu Server mit der Version 18.04.4, auf der anderen VM lief ein Ubuntu Desktop System mit der Version 19.10. Der Vorgang zum Aufsetzen, erfolgt wie üblich. Ich empfehle lediglich dem Client mehr als 1GB RAM zuzuweisen, da er sonst während der Installation abstürzen kann. Zusätzlich sollte man die neueste Version von VirtualBox installieren, da in älteren Versionen offiziell Bugs bei der Kommunikation zwischen den VMs bestehen. Dies kann sonst einige Stunden Aufwand kosten :) .
Da zur Verwendung von WireGuard eine Kommunikation zwischen Client und Server und zum Internet notwendig ist, müssen ein paar Konfigurationen in der VirtualBox vorgenommen werden. Dazu muss bei beiden VMs auf den Netzwerk Adaptern *NAT* ausgewählt sein (default). Beim Ubuntu Server muss man einen zusätzlichen Netwerkadapter aktivieren und ihn als *Host-only* Adapter einstellen.

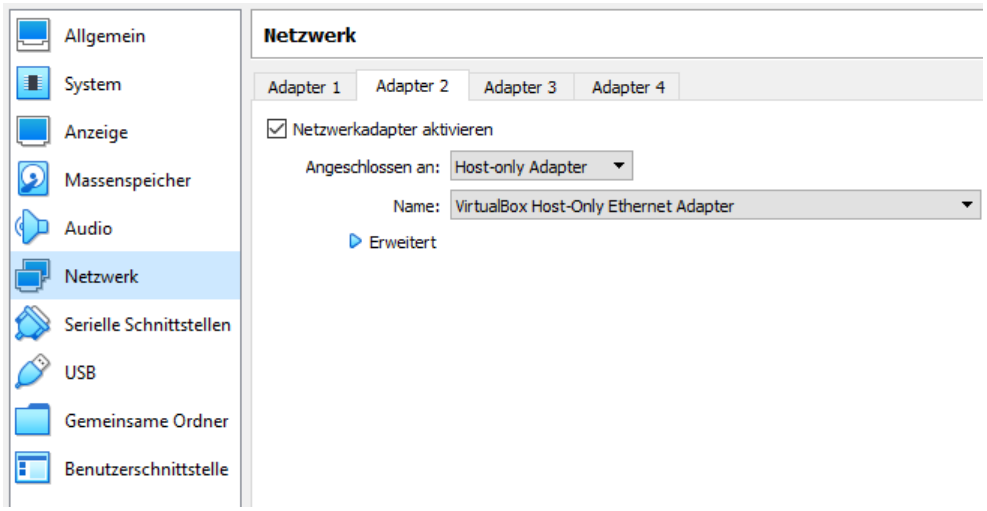

Abbildung 1.2: Host-only Adapter

Es wird davon abgeraten ein eigenes NAT-Network zu konfigurieren, da in VirtualBox somit zwar eine Kommunikation zwischen den VMs funktioniert, jedoch keine Kommunikation zum Internet, wordurch keine Installation von WireGuard möglich ist.

Anschließend muss diesem neu angelegten Adapter eine IP-Adresse zugewiesen werden. Der Name des Adapters kann mit dem Befehl *ifconfig -a* identifiziert werden. Unser gewünschter Adapter ist derjenige, der keine IP Adresse besitzt. In meinem Fall war das *enp0s8*. Dazu kann mit dem Befehl

```
1 $ vi /etc/network/interfaces
```

dem Adapter folgenderweise eine IP-Adresse zugewiesen werden:

```
1 auto enp0s8
2 iface enp0s8 inet static
3 address 192.168.56.10
4 netmask 255.255.255.0
```

Zuletzt kann wieder der Befehl *ifconfig* ausgeführt werden, der Folgendes zurückliefert:

```
karlo@server:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe51:61de  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:51:61:de  txqueuelen 1000  (Ethernet)
        RX packets 1117  bytes 894071 (894.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 354  bytes 31037 (31.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.103  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe2d:9e9a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:2d:9e:9a  txqueuelen 1000  (Ethernet)
        RX packets 4  bytes 1488 (1.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 1328 (1.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Abbildung 1.3: IP Adressen der Netzwerkadapter

1.3.2 Server

Sind nun die Schritte aus Abschnitt 1.3.1 ausgeführt, kann die Installation von WireGuard beginnen. Dazu sind der Server und der Client zu starten. Um WireGuard am Server zu installieren ist am Client folgender Befehl auszuführen:

```
1 $ ssh <username>@192.168.56.103
```

Anschließend wurde mit folgenden Befehlen WireGuard am Server installiert:

```
1 add-apt-repository ppa:wireguard/wireguard
2 apt-get update
3 apt-get install wireguard-dkms wireguard-tools linux-headers-$(uname -r)
```

Danach wurde mit folgenden Befehlen ein private und public Key generiert:

```
1 umask 077
2 wg genkey | tee server_private_key | wg pubkey > server_public_key
```

Um die Konfigurationen langfristig zu speichern kann mit dem Befehl

```
1 $ vi /etc/wireguard/wg0.conf
```

eine Konfigurationsdatei erstellt werden, die folgenden Inhalt enthält:

```
1 [Interface]
2 Address = 10.100.100.1/24
3 SaveConfig = true
4 PrivateKey =
5 ListenPort = 51820
6 PostUp = iptables -A FORWARDls -i %i -j ACCEPT; iptables -A FORWARD -o %i -j ACCEPT;
      iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
7
8 PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j ACCEPT;
      iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE
9
10 [Peer]
11 PublicKey =
12 AllowedIPs = 10.100.100.2/32
```

In Zeile 4 tragen wir den generierten Private Key des Servers ein. Zeile 11 bleibt vorerst so wie sie ist.

Anschließend muss noch IPv4 forwarding erlaubt werden, um den Zugriff auf das gesamte LAN zu ermöglichen. Dazu ist folgender Befehl auszuführen:

```
1 $ vi /etc/sysctl.conf
```

und in folgender Zeile den Befehl:

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

auszukommenntieren.

Um diese Änderungen zu übernehmen kann entweder der Server mit dem Befehl *reboot* neugestartet werden, oder mit folgenen Befehlen ohne eines Neustarts übernommen werden:

```
1 sysctl -p
2 echo 1 > /proc/sys/net/ipv4/ip_forward
```

Soweit so gut. Der Server ist nun konfiguriert.

WireGuard kann nun mit folgenden Befehl gestartet werden:

```
1 $ wg-quick up wg0
```

Möchte man, dass WireGuard automatisch gestartet wird, sobald der Server gestartet wird, kann man das mit folgenden Befehl erreichen:

```
1 $ systemctl enable wg-quick@wg0.service
```

1.3.3 Client

Zuerst muss WireGuard installiert werden. Dieser Vorgang ist in Abschnitt 1.2 erklärt. Für Ubuntu 19.10 habe ich folgenden Befehl ausgeführt:

```
1 $ sudo apt install wireguard
```

Anschließend wird für den Client ein private und public Key generiert.

```
1 wg genkey | tee client_private_key | wg pubkey > client_public_key
```

Nun muss noch in der Server Konfigurationsdatei in Zeile 11 der Public Key des Clients eingetragen werden.

Zuletzt wird wieder eine Konfigurationsdatei angelegt.

```
1 $ vi /etc/wireguard/wg0-client.conf
```

Diese hat folgenden Inhalt:

```
1 [Interface]
2 Address = 10.100.100.2/32
3 PrivateKey =
4
5 [Peer]
6 PublicKey =
7 Endpoint = :51820
8 AllowedIPs = 0.0.0.0/0
9 PersistentKeepalive = 21
```

In Zeile 3 wird der Private Key des Clients eingetragen und in Zeile 6 der Public Key des Servers. Zeile 7 enthält zusätzlich noch die IP Adresse des Servers.
Wireguard ist nun am Client konfiguriert und kann mit folgenden Befehl gestartet werden:

```
1 $ sudo wg-quick up wg0-client
```

Überprüft man nun die IP Adresse des Clients, stellt man fest, dass die IP Adresse des WireGuard Servers zu sehen ist, anstelle der ursprünglichen IP Adresse des Clients.
Führt man den Befehl

```
1 $ sudo wg-quick down wg0-client
```

aus erhält man wieder die ursprüngliche IP Adresse des Clients.

1.4 Technische Funktionalität

1.4.1 VPN

Als VPN (Virtual Private Network) bezeichnet man ein logisches privates Netzwerk, das auf einer öffentlichen Infrastruktur zugänglich ist. Ausschließlich Kommunikationspartner, die zu diesem privaten Netzwerk gehören, können miteinander kommunizieren.
Die Grundsätze von VPNs um eine sichere Kommunikation zu gewährleisten lauten:

- **Authentizität**
  Eigenschaft, die gewährleistet, dass ein Kommunikationspartner tatsächlich derjenige ist, der er vorgibt zu sein.
- **Vertraulichkeit**
  Schutz vor unbefugter Preisgabe von Information. Vertrauliche Daten dürfen ausschließlich den Kommunikationspartnern im privaten Netzwerk zugänglich sein.
- **Integrität**
  Sicherstellung der Korrektheit von Daten.

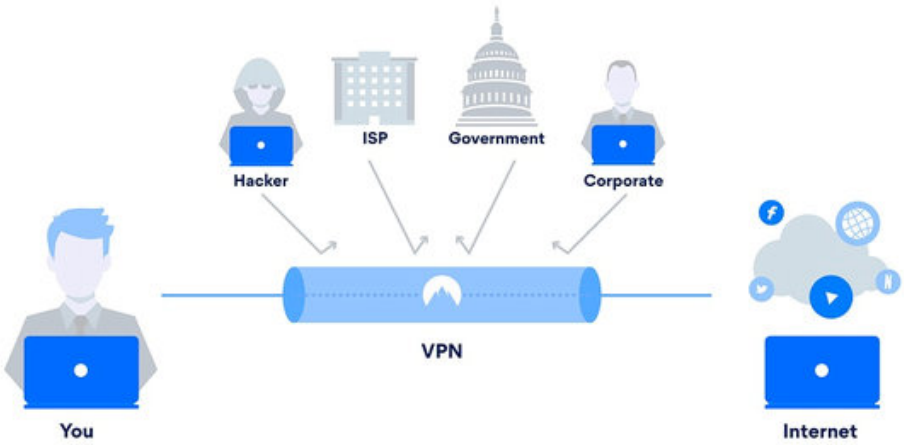

Abbildung 1.4: VPN Funktionsweise

Ablauf VPN-Verbindung

1. Ein VPN-Client stellt eine Verbindung zu einem VPN-Server über ein VPN-Protokoll her.
2. Der VPN-Server überprüft die Zugangsdaten, nimmt die Verbindung dementsprechend an und einigt sich mit dem VPN-Client auf einen Schlüssel zur sicheren Kommunikation.
3. Es entsteht eine verschlüsselte Verbindung zwischen dem VPN-Server und dem VPN-Client, die auch als *Tunnel* bezeichnet wird und eine physische Verbindung simuliert.
4. Der VPN-Client kann auf das lokale Netzwerk des VPN Servers zugreifen, als wäre er vor Ort mit diesem Netzwerk verbunden.