NEU! Linux-Expertenwissen für Profis

LINUX PROFESSIONELL

Linux fürs Business

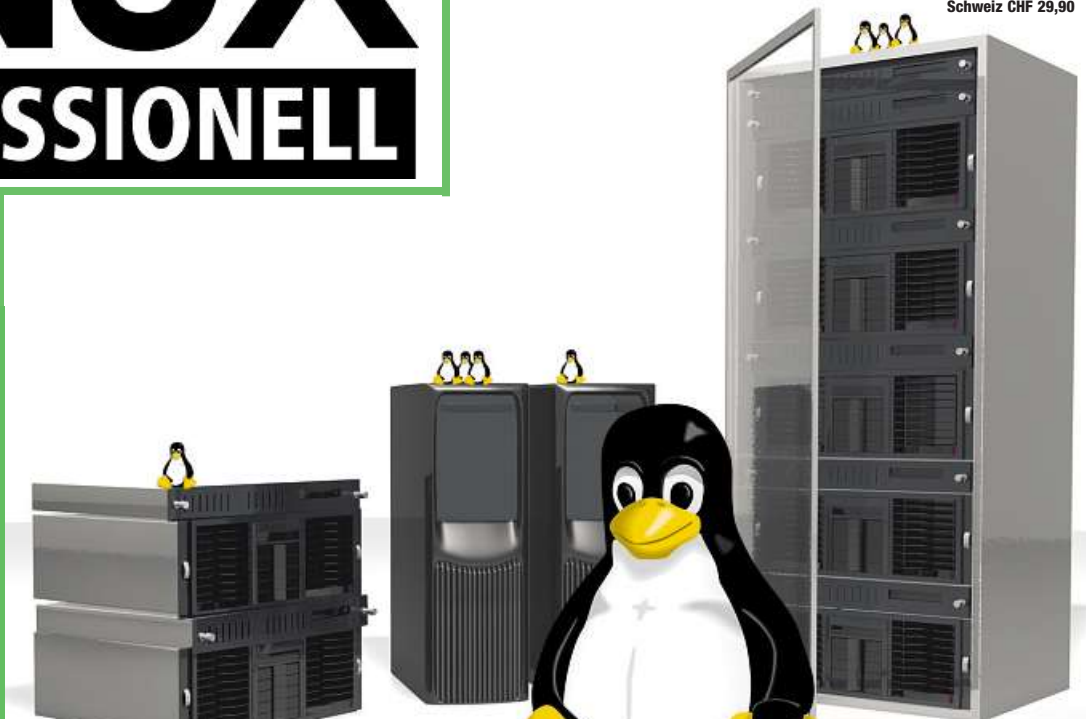
Erfahrungswerte: Hochverfügbarkeits-Server S. 14

vnu business publications deutschland

03|05 €14,95

Österreich € 16,45
Benelux € 17,20
Schweiz CHF 29,90

Erstmals auf einer **Heft-DVD!**



LINUX PROFESSIONELL

Zur Homepage
www.linux-professionell.net

Übersicht Praxis Workshops als PDF

Linux Professionell Übersicht aller Ausgaben

Suse Linux Enterprise Server 9 mit SP 1*

*Ohne Support, Updates für 30 Tage nach Registrierung

Server im Vergleich

Red Hat Enterprise Server, Debian GNU/Linux, Mandriva Corporate Server 3.0, Suse Linux Enterprise Server 9 S. 20

Server-Technik

Virtualisierung mit ZEN, Linux auf 64-Bit-CPUs, Network File System, Global File System S. 37

Server-Praxis

VoIP-Gateway mit Asterisk, Postfix-Mailserver, File-, Web-Proxy- und FTP-Server, VPN-Gateway, Firewall und SSH S. 45

Plus Profi-Wissen

- ☑ Grundlagen
- ☑ Workshops
- ☑ Insider-Tipps
- ☑ Software

Red Hat, Suse & Co.
Linux-Server

Zentrale Schaltstelle im LAN
Zuverlässig, sicher & schnell

Tuning: High-Performance-Cluster

Server zusammenschließen – Geschwindigkeit potenzieren S. 37

Trends: Neue Power-Prozessoren

IBM 64-Bit-Power5- und Cell-CPU für Linux-Server S. 10

Telefonieren: VoIP-Gateway

Standorte verbinden – kostenlose Gespräche führen S. 45

03
4 196788 914953

Im Tunnel durchs Netz

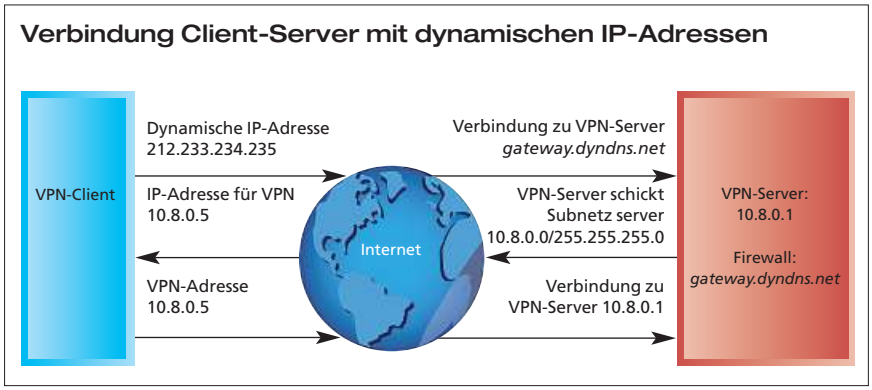
Mit OpenVPN 2.0 erhalten Anwender eine kostenlose VPN-Implementierung auf Basis des Secure Socket Layers. Das System ist leicht zu konfigurieren, da keine Änderungen am Kernel vorzunehmen sind. Linux Professionell zeigt die Einrichtung des Servers und die Verbindung per Client-PC.

JÖRG REITTER

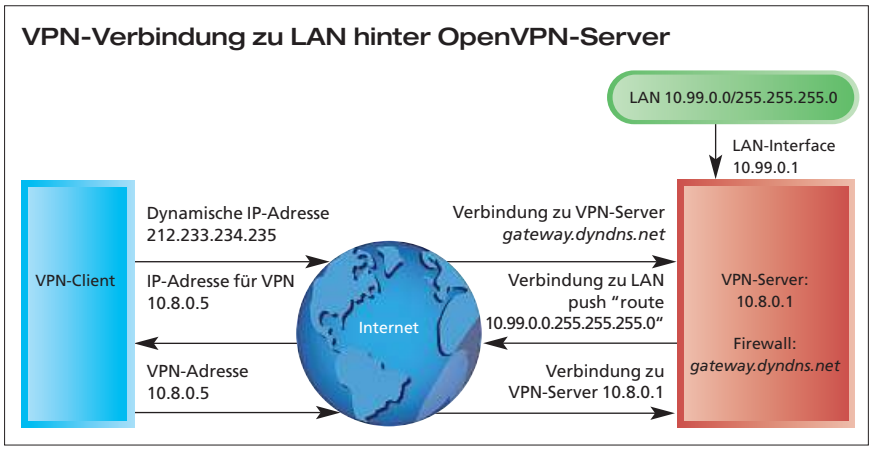
Für Linux gibt es bereits einige VPN-Implementierungen. Keine lässt sich allerdings so einfach und komfortabel im Netzwerk etablieren wie OpenVPN 2.0 *(http://openvpn.net/)*. Die Software basiert auf SSL (Secure Socket Layer), wodurch der Administrator keinerlei Änderung am Linux-Kernel vornehmen muss – so wie es bei den IPsec-Implementierungen von FreeS/WAN oder Openswan notwendig ist. Zur Authentifikation lassen sich Zertifikate oder normale Zugangs-Accounts nutzen. Sogar Smartcards unterstützt das System. Die Zugriffskontrolle funktioniert mit Policies für bestimmte Benutzer oder ganze Gruppen.

Ein großer Vorteil gegenüber anderen VPN-Produkten ist zudem, dass OpenVPN keine Probleme mit Network Address Translation (NAT) respektive dynamischen IP-Adressen hat. So kann ein Benutzer aus dem Firmen-LAN den VPN-Server zu Hause kontaktieren, ohne dass die Firmen-Firewall etwas davon bemerkt. Zudem ist das System nicht auf Linux als Plattform beschränkt. Auch Solaris, BSD-Varianten, Mac OS X sowie Windows 2000/XP lassen sich als OpenVPN-Server oder -Client einsetzen. Damit bekommt der Verwalter eine flexible Lösung, die er unternehmensweit etablieren kann.

Wer allerdings eine Hardware-Appliance als Tunnel-Endpunkt einsetzt, dürfte darin häufig keine SSL/TLS-Unterstützung haben. Somit lässt sich auf der anderen Seite, etwa einer Außenstelle, OpenVPN nicht einsetzen, da die Software nicht mit IPsec-, PPTP- oder L2TP-Verfahren zusammenarbeitet.



Der VPN-Client bekommt das virtuelle Subnetz vom Server zugewiesen



Der VPN-Server schickt dem Client die Route zu dem hinter ihm liegenden lokalen Netzwerk

OpenVPN installieren

Vor der Installation von OpenVPN stellen Sie sicher, dass die Pakete *OpenSSL 0.9.6* für Zertifkate, *Lzo* für die Kompression des Traffics sowie *PAM* zur Sicherung der Authentifikation auf dem System vorhanden sind. Alle Pakete sind Bestandteil einer üblichen Distribution. Suse 9.3 Professional enthält zudem das OpenVPN-Paket 2.0. Anwender von Red Hat Fedora müssen das Paket besorgen und installieren. Wer auf Debian Sarge setzt, bekommt die aktuelle Version per:

```
apt-get install openvpn
```

Für ein RPM-basiertes Linux Fedora Core besorgen Sie die Binär-Pakete von *http://openvpn.net* und installieren diese mit:

```
rpm -ivh openvpn.rpm
```

Die Konfigurationsdateien befinden sich dann je nach Distribution im Verzeichnis */usr/share/doc/packages/openvpn* oder */usr/share/doc/openvpn-2.0*. Das eigentliche Konfigurationsverzeichnis legt der Installer im Folder */etc/openvpn* an.

VPN planen

Virtual Private Networks sind dazu gedacht, komplette Netzwerke an verschiedenen Standorten über das Internet zu verbinden. Die Computer in den lokalen Netzwerken (LANs) sind mit privaten IP-Adressen konfiguriert, die Internet-Router nicht weiterleiten. Was für einzelne Netzwerke sehr angenehm ist, kann beim Zusammenschluss mehrerer LANs über ein VPN zum Problem werden, wenn die Netzwerke dieselben IP-Blöcke benutzen. Bevor der Administrator sich an die Konfiguration des OpenVPN macht, müssen die beteiligten Netzwerke auf solche konfliktträchtigen Einstellungen abgeklopft werden. Das OpenVPN-How-to auf *http://openvpn.net/howto.html* empfiehlt daher, auf Subnetze in selten benutzten Blöcken wie *10.0.0.0/8* zurückzugreifen. Vorteil: Der Verwalter umgeht Konflikte mit den häufig eingesetzten Blöcken *192.168.0.0/24* und *172.16.0.0/12* bereits im Vorfeld. Auch Benutzer, die sich von WLAN-Hotspots oder Internet-Cafés im VPN einloggen möchten, bekommen keine Probleme mit den dortigen Subnetzen, die meist eine Adresse aus *192.168.0.0/24* einsetzen.

Certificate Authority benutzen

Ein wesentlicher Baustein der verschlüsselten Kommunikation in einem VPN sind digitale Zertifikate, mit denen sich eine Public-Key-Infrastruktur (PKI) bauen lässt. OpenVPN 2.0 unterstützt die bidirektionale Authentifikation, bei der Server und Client das jeweilige Zertifikat des anderen erst bestätigen müssen, bevor sich der Tunnel etabliert. Dazu benötigt der Anwender eine Certificate Authority (CA), die ein Master-Zertifikat ausstellt und einen Schlüssel, mit dem sich die Server- und Client-Zertifikate signieren

lassen. Diese Zertifikate entsprechen dem öffentlichen Schlüssel (Public Key). Dazu kommt noch ein privater Schlüssel (Private Key) für den Server und alle Clients.

Die Authentifikation verläuft dann nach folgendem Schema: Server und Client authentifizieren sich gegenseitig, indem sie prüfen, ob die Certificate Authority das jeweilige Zertifikat signiert hat. Anschließend testen beide Seiten, ob die Informationen im Zertifikat-Header, wie beispielsweise die Zertifikat-Typen, zusammenpassen.

Wichtiger ist, dass der Server nur solchen Clients die Verbindung erlaubt, deren Zertifikate von der CA signiert wurden. Das ist ein großer Sicherheitsgewinn, denn der Server benötigt für die Prüfung keinen Zugriff auf den privaten Schlüssel der CA. Dieser muss sich nicht einmal auf der VPN-Maschine befinden, wodurch das sensibelste Teil der PKI in Sicherheit ist.

Auch wenn ein privater Schlüssel kompromittiert ist, zeigt dieses Verfahren seine Stärke. Das Zertifikat lässt sich ganz einfach deaktivieren, indem es der Administrator auf einer Rückrufliste (Certificate Revocation List) platziert. Dadurch muss der Server nur das betreffende Zertifikat zurückweisen, aber nicht die gesamte Public-Key-Infrastruktur neu aufbauen.

Master-Zertifikat anlegen

Das Zertifikat der Master-CA sowie den Schlüssel legen Sie zuerst an. Dies ist das Fundament, auf dem das VPN steht. Wir generieren je ein Zertifikat respektive einen Schlüssel für die Master-CA, den Server sowie einen Client. Das OpenVPN-Paket enthält diverse Scripts, um die Erstellung zu erleichtern.

Nach einer RPM-Installation befinden sich Scripts und Beispieldateien im Verzeichnis */usr/share/doc/packages/openvpn* oder */usr/share/doc/openvpn-2.0*. Kopieren Sie das OpenVPN-Verzeichnis zuerst nach */etc/openvpn* und nehmen dort die Konfiguration auf. Wechseln Sie dann ins Verzeichnis */etc/openvpn/easy-rsa* und laden die Datei *vars* in einen Editor. Darin tragen Sie die Daten für *KEY_COUNTRY*, *KEY_PROVINCE*, *KEY_CITY*, *KEY_ORG* und *KEY_EMAIL* ein. Den Servernamen geben Sie während des Scriptlaufs an. Anschließend initialisieren Sie die PKI mit folgenden Kommandos:

```
. ./vars
./clean-all
./build-ca
```

Wichtig: Den ersten Befehl leitet ein Punkt ein, gefolgt von einem Leerzeichen. So lässt sich eine Datei ausführen, die wie im Fall von *vars* eine Umgebungsvariable enthält und diese mit *export* setzt. Das Script *clean-all* löscht den Inhalt eines existierenden *keys*-Verzeichnisses. Schließlich generiert *build-ca* das Master-CA-Zertifikat *ca.crt* sowie den Schlüssel *ca.key* und legt diese in */etc/openvpn/easy-rsa/keys* ab.

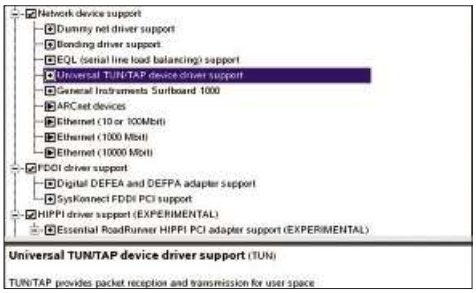
Als Nächstes steht die Generierung des Server-Zertifikats und des Schlüssels an. Diese erstellen Sie mit dem Kommando:

```
./build-key-server srv150
```

Die Fragen beantworten Sie wie vorher bei der CA. Lediglich den Common Name ändern Sie auf *srv150*. Zum Schluss bejahen Sie die Fragen *Sign the certificate?* und *1 out of 1 certificate requests certified, commit?* Zertifikat *srv150.crt* und Schlüssel *srv150.key* liegen im Verzeichnis *keys*. Erstellen Sie das Zertifikat und den Schlüssel für den Client:

```
./build-key wks99
```

Diesen Befehl muss der Administrator für jeden einzelnen Client wiederholen, da jeder Client ein eigenes Zertifikat und einen Schlüssel braucht. Als zusätzliche Sicherheitsmaßnahme lässt sich der Schlüssel auch per Passwort sichern. Benutzen Sie dafür das Script *build-key-pass*.



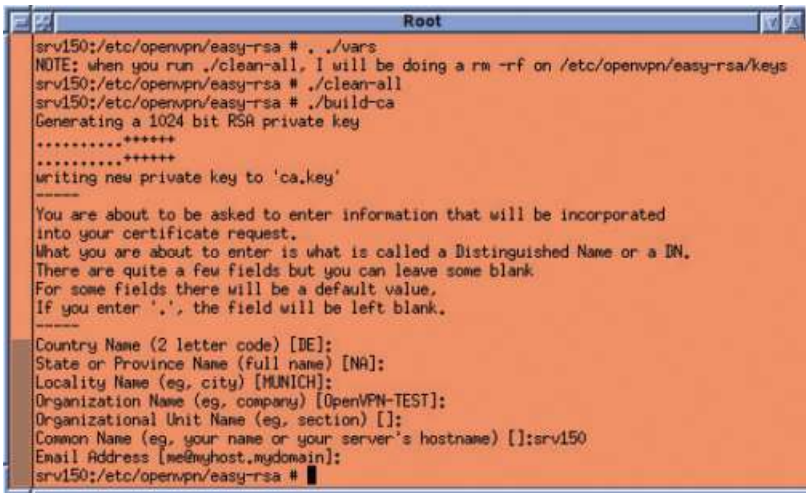
Der Treiber der TUN/TAP-Devices liegt auf vielen Distributionen als ladbares Modul vor

Da nun das Zertifikat vorliegt, erstellt der Anwender Parameter des Diffie-Hellman-Verschlüsselungsalgorithmus für den Server:

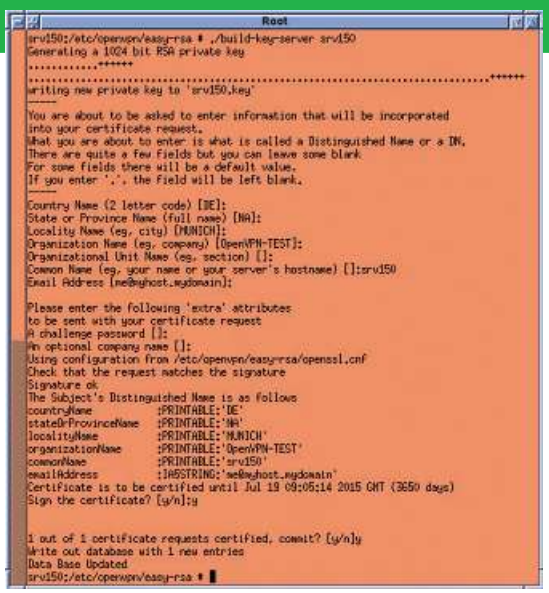
```
./build-dh
```

Listing 1: /etc/server.conf

```
# OpenVPN 2.0 server.conf
# Explizites Setzten des Server-Ports
port 1194
# UDP als Transportprotokoll
proto udp
# Routing-Modus mit /dev/tun
dev tun
# Name und Pfad der Zertifikate/Schlüssel-Dateien
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/srv150.crt
key /etc/openvpn/easy-rsa/keys/srv150.key
# Schlüssellänge des Diffie-Hellman: 1024
dh /etc/openvpn/easy-rsa/keys/dh1024.pem
# VPN-Subnetz für Clients; Server benutzt 10.8.0.1
server 10.8.0.0 255.255.255.0
# Zuordnung von Client-IP-Adressen zu VPN-Adressen  in ipp.txt
  führen. Wichtig für Re-Connects.
ifconfig-pool-persist ipp.txt
# In ccd liegen Client-spezifische Konfigurationen
client-config-dir ccd
# Route zum Subnetz hinter dem Client setzen für Multi-Client-Zugriff
route 192.168.4.0 255.255.255.0
# Ein Ping ähnlicher Mechanismus prüft alle 10 Sekunden, ob die
  Gegenseite noch da ist. Nach 120 Sekunden gilt die Verbindung als
  getrennt.
keepalive 10 120
# Clients im selben Subnet können auch andere Clients sehen
client-to-client
# Kompression benutzen
comp-lzo
# Anonyme Benutzer/Gruppe nobody mit UID/GID 65534
user nobody
group nobody
# Bei unerwartetem Verlust der Verbindung:  Schlüssel behalten und
  Interface nicht herunterfahren.
persist-key
persist-tun
# Protokolliere alle momentanen Verbindungen
status openvpn-status.log
# Meldungen an Syslog mit Verbose-Level 3
verb 3
```



OpenVPN unterstützt standardmäßig digitale Zertifikate. Möchte der Administrator dafür eine eigene Certification Authority anlegen, können dies die OpenVPN beiliegenden Scripts leicht und komfortabel erledigen



Jeder Server und Client benötigt ein eigenes Zertifikat und einen Schlüssel. Scripts rufen die passenden »openssl«-Befehle zur Generierung auf

Das Script benutzt standardmäßig eine Schlüssellänge von 1024 Bit und erzeugt die Datei *dh1024.pem*. Möglich sind auch 2048 Bit. Die Sicherheit verlangt, dass die Dateien nun noch über einen sicheren Kanal auf die beteiligten Maschinen gelangen. Dies lässt sich etwa mit *Scp* (Secure-Copy) erreichen.

Server konfigurieren

Benutzen Sie die mitgelieferten Beispieldateien, haben Sie eine recht unkomplizierte Konfiguration vor sich. Die Beispiele befinden sich im Verzeichnis */etc/openvpn/sample-config-files*. Die Konfigurationsdateien heißen konsequenterweise *server.conf* (Listing 1) und *client.conf* (Listing 2).

Zuerst konfiguriert der Administrator den Server. Dieser soll am UDP-Port *1194* auf Verbindungswünsche warten und aus dem virtuellen Subnetz *10.8.0.0/24* Adressen an die VPN-Clients verteilen. Für die effizientere Verbindung per Routing sorgt das virtuelle TUN-Interface *(dev tun)*. Es stellt User-Space-Programmen wie OpenVPN die zwei Schnittstellen */dev/tunX* (Character-Device) und *tunX* (virtuelles Punkt-zu-Punkt-Gerät) zur Verfügung. Das TUN/TAP-Device ist als optionaler Treiber im Kernel enthalten und liegt auf den meisten Distributionen als Modul vor. Weitere Informationen hierzu lesen Sie im Kasten »Routing oder Bridging – TUN oder TAP« auf der nächsten Seite.

Bevor Sie *server.conf* produktiv einsetzen können, passen Sie die Pfade zum Root-Zertifikat *ca*, dem Server-Zertifikat *cert*, dem Server-Schlüssel *key* sowie den DH-Parametern *dh* an. Zudem sollten Sie die Direktiven *user nobody* und *group nobody* benutzen. Damit mappt der VPN-Server jeden Benutzer auf die Pseudo-User *nobody* mit der UID *65534* respektive jede Gruppe auf *nobody* mit GID *65534*. Dadurch verliert Benutzer *root* seine Privilegien.

Client konfigurieren

Die Konfiguration des Clients ist mit noch weniger Aufwand verbunden. Passen Sie in */etc/openvpn/client.conf* ebenfalls die Pfade bei *ca*, *cert* und *key* an. Denken Sie daran, dass jeder Client ein eigenes Zertifikat beziehungsweise Schlüssel-Paar benötigt. Das vorher angelegte Zertifikat für *wks99* lässt sich nicht auf andere Maschinen übertragen. Lediglich das Root-CA-Zertifikat *ca* ist im ganzen OpenVPN gültig.

Füllen Sie anschließend die Anweisung *remote* mit der IP-Adresse und Port-Nummer des OpenVPN-Servers aus. Sofern der VPN-Server keine statische öffentliche IP-Adresse besitzt respektive hinter einer Firewall oder einem NAT-Gateway steht, gehen Sie so vor: Tragen Sie die Adresse des NAT-Gateways ein und leiten Sie den Port *1194/udp* zum dahinter stehenden VPN-Server (Forwarding).

Bekommt das NAT-Gateway die IP-Adresse dynamisch zugewiesen, wie das häufig in kleinen Büros oder im privaten Netzwerk der Fall ist, greifen Sie am besten auf die

Listing 2: /etc/client.conf

```
# OpenVPN 2.0 client.conf
# Die Maschine ist Client
client
# UDP als Transportprotokoll
proto udp
# Routing-Modus mit /dev/tun
dev tun
# IP-Adresse oder FQDN des OpenVPN-Servers sowie Port 1194
remote gateway.dyndns.net 1194
# Unendlich versuchen, den Hostnamen des Servers zu ermitteln.
# Hilfreich bei nur temporär mit dem Internet verbundenen Rechnern.
resolv-retry infinite
# Keine Bindung an lokale Portnummer
nobind
# Anonyme Benutzer/Gruppe nobody mit UID/GID 65534
user nobody
group nobody
# Bei unerwartetem Verlust der Verbindung:  Schlüssel behalten
  und Interface nicht herunterfahren.
persist-key
persist-tun
# Name und Pfad der Zertifikate/Schlüssel-Dateien
ca /etc/openvpn/easy-rsa/keys/ca.crt
cert /etc/openvpn/easy-rsa/keys/wks99.crt
key /etc/openvpn/easy-rsa/keys/wks99.key
# Kompression benutzen
comp-lzo
# Meldungen an Syslog mit Verbose-Level 3
verb 3
```

Dienste eines kostenlosen Dyn-DNS-Providers zurück. Dieser synchronisiert die momentane öffentliche IP-Adresse beispielsweise mit dem Host *gateway.dyndns.net* und trägt diesen temporär in einen DNS-Record ein. So lässt sich das Gateway immer über *gateway.dyndns.net* erreichen, obwohl sich die IP-Adresse bei jeder Einwahl beim Provider ändert. Dann benutzen Sie wieder das Forwarding, um den VPN-Server anzubinden. Zum Schluss prüfen Sie, ob die Einträge in der Datei *client.conf* mit denen der Server-Konfiguration identisch sind. Besonders wichtig hierbei sind das Device *tun* sowie das Transportprotokoll *UDP*.

Server starten

Nun ist der Server prinzipiell bereit für den ersten Start. Allerdings sollten Sie vorher probieren, ob sich der Server mit einem Ping über das Internet erreichen lässt. Falls das Ping fehlschlägt, liegt es an einer falschen Konfiguration einer Firewall. Entweder passt die Forwarding-Regel nicht oder die Firewall lässt *UDP* am Port *1194* nicht zu.

Möglicherweise ist auch eine Firewall auf dem VPN-Server oder -Client Schuld. Diverse Distributionen installieren standardmäßig Firewalls, die jeden eingehenden Verkehr blocken, insbesondere neue Netzwerk-Schnittstellen wie TUN. In diesem Fall fügen Sie neue Regeln hinzu, die das entsprechende Szenario abdecken. Passende Regeln findet der Verwalter in den OpenVPN-FAQs auf *http://openvpn.net/faq.html#firewall*.

Ist die Server-Maschine schließlich erreichbar, starten Sie den OpenVPN-Dienst mit folgendem Kommando:

```
openvpn
```

Der Server schreibt seine Startmeldungen in das System-Log in */var/log/messages*. Diese Aktion sollten Sie während der weiteren Tests in Echtzeit verfolgen:

```
tail -f /var/log/messages
```

Klappt etwas nicht wie erwartet, können die Einträge in der Protokolldatei einen Hinweis auf die Fehlerursache geben.

Verbindung testen

Schließt der Server den Startvorgang von OpenVPN mit der Meldung *Initialization Sequence Completed* erfolgreich ab, lässt sich auch der Client für die erste Verbindung auf der Kommandozeile starten:

```
openvpn
```

Zuerst probiert der Anwender, wieder ein Ping an den Server durchzubringen. Läuft dies erfolgreich, steht die Verbindung über den verschlüsselten Tunnel.

Um OpenVPN-Client oder -Server automatisch beim Systemstart zu laden, benutzen Sie das Init-Script. Dieses liegt dem heruntergeladenen Paket bei. Der Administrator aktiviert dieses im Verzeichnis */etc/init.d* für die Runlevel mit Netzwerk-Unterstützung 3, 4 und 5. Komfortabel funktioniert dies beispielsweise mit einem grafischen Programm wie *system-config-services* (Red Hat) oder dem Novell/Suse-Runlevel-Editor, der sich in Yast in der Rubrik System aufhält. Kommt OpenVPN 2.0 von der Novell/Suse-Distribution 9.3 Professional, existiert bereits ein fertiges Init-Script, das sich direkt starten lässt. Der Befehl lautet:

```
/etc/init.d/openvpn start
```

Zugriff auf Server im VPN-Subnetz

Mit dem Zugriff auf den OpenVPN-Server dürften sich nur die wenigsten Anwender zufrieden geben. Schließlich will der Benutzer Daten von Fileservern oder Datenbanken einsehen oder auf den Client transferieren. Hierzu muss der VPN-Server die Clients auf Hosts im Sub-Netzwerk umleiten und den Zugriff kontrollieren. Im Routing-Modus mit */dev/tun* lässt sich der Server so konfigurieren, dass er die Route zu seinem lokalen Netzwerk den Clients bekannt macht. Dazu ist die Direktive *push* vorgesehen:

```
push "route
  10.99.0.0 255.255.255.0"
```

Danach setzt der Verwalter eine Route auf dem LAN-Gateway, um die Clients aus *10.8.0.0* zum OpenVPN-Server zu leiten.

Zugriff von mehreren Clients

Um die Konfiguration zu komplettieren, sollen auch mehrere Clients aus einem entfernten Subnetz Zugriff auf die Server hinter dem OpenVPN-Server bekommen. Das folgende Szenario routet alle Clients im LAN *192.168.4.0/24* über das lokale VPN-Gateway zum entfernten VPN-Server. Hier ist vorher wieder besonders darauf zu achten, dass das Client-LAN eine Netzwerk-Adresse hat, die im restlichen Unternehmen nicht mehr vorkommt. Jeder Client muss zudem mit einem einzigartigen Common Name im Zertifikat aufgeführt sein, um eindeutig identifiziert zu werden. Der zweite Client benutzt ein Zertifikat mit dem Common Name *wks2*.

Zuerst verweist der Administrator im File *server.conf* auf ein Client-Konfigurations-Verzeichnis, in dem sich weitere Anweisungen befinden. In diesem Fall steht darin nur die Route zum Server. OpenVPN legt mit *ccd* dazu ein Beispielverzeichnis unter */etc/openvpn* an. Auf dieses referenzieren Sie mit:

```
client-config-dir ccd
```

Verbindet sich ein Client zum OpenVPN-Server, prüft der Server-Daemon, ob eine Datei existiert, die den Common Name des Clients enthält. Findet der Daemon so eine Datei, sucht er darin nach weiteren Anweisungen. Der Administrator kann somit eine besondere Konfiguration für Gruppen anlegen, beispielsweise für Außendienst- oder Telearbeiter. Auch einzelne Clients wie wks2 lassen sich speziell konfigurieren.

Für den Client wks2 legt der Administrator folglich die Datei *wks2* im Verzeichnis *ccd* an. Diese enthält lediglich die Route zum Subnetz *192.168.4.0/24*:

```
iroute 192.168.4.0 255.255.255.0
```

Fast dieselbe Zeile muss auch die *server.conf* aufweisen. Lediglich der Routing-Befehl heißt darin anders:

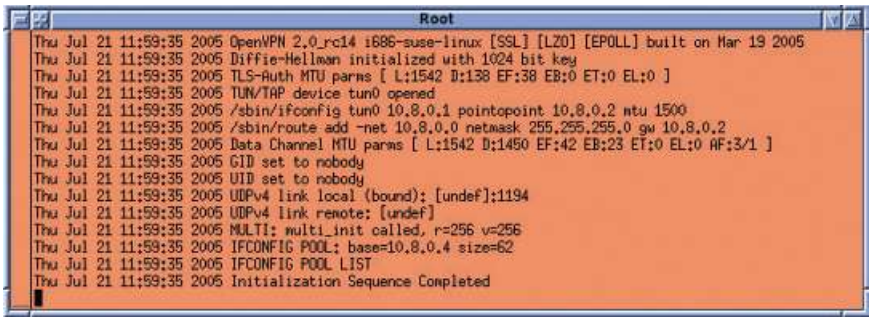
```
route 192.168.4.0 255.255.255.0
```

Routing oder Bridging – TUN oder TAP

Der TUN/TAP-Treiber des Linux-Kernels stellt die zwei Modi Routing und Bridging zur Verfügung, für die sich virtuelle Schnittstellen anlegen lassen. OpenVPN 2.0 kann mit beiden umgehen. Die zwei Modi gleichzeitig auf einer Maschine einzusetzen, funktioniert jedoch nicht.

Ein VPN, das mit IP-Routen arbeitet, benutzt die TUN-Schnittstelle. Diese Konfiguration ist einfach herzustellen und reicht für die meisten Anforderungen aus. Darüber hinaus ist das geroutete VPN effizienter und skaliert besser, da lediglich Routing-Tabellen gesetzt werden. Je spezieller die Zugriffe der Clients sein sollen, desto mehr Konfigurationsaufwand steht jedoch ins Haus. Wer den Clients aus verschiedenen Subnetzen Zugriff auf einen zentralen SMB-Fileserver geben möchte, benötigt gleichzeitig einen WINS-Server.

Wenn Software von Broadcasts (Windows-Netze) abhängt, kommt der Verwalter hingegen nicht um den Einsatz des Bridging-Modus herum. Auch wer Unterstützung für andere Protokolle abseits von IP benötigt oder wem der OpenVPN-Server gleichzeitig als WLAN-Hotspot dient, benutzt das TAP-Interface. Der Administrator erhält dadurch ein virtuelles Ethernet, das sich bei Bedarf mit einem realen Ethernet koppeln lässt (Bridge). Nachteil des Bridging-Modus ist der erhöhte Bandbreitenbedarf aufgrund des größeren Overheads, da die Verwaltung des Software-Ethernets mehr Daten produziert. Allerdings ist damit auch ein recht einfacher Zugriff auf Samba- oder Windows-Shares möglich, da Broadcasts durch das VPN laufen. Wie sich beispielsweise WLAN-Clients per TAP-Schnittstelle als VPN-Clients konfigurieren lassen, erklärt die Anleitung auf *www.closeconsultants.com/~peter/m0n0-ovpn-wifi.html*.



Während der OpenVPN-Server startet, gibt er im Terminal diverse Einstellungen und Informationen auf »nobody« aus, beispielsweise das UID/GID-Mapping

Die Befehle *iroute* und *route* spielen bestimmte Rollen. Das Kommando *iroute* ist für das Routing vom OpenVPN-Server zu den Clients verantwortlich, während *route* die Routing-Tabelle des Kernels des OpenVPN-Servers setzt. Optional kann der Verwalter noch erlauben, dass sich die VPN-Clients untereinander austauschen dürfen. Dazu dient die Anweisung *client-to-client* in *server.conf*:

```
client-to-client
push "route
  192.168.4.0 255.255.255.0"
```

Der OpenVPN-Server übernimmt über das Statement *push* das Setzen der Route zum Subnetz der Clients.

VPN-Server gleich Gateway

Der OpenVPN-Server steht im Unternehmen hinter der Firewall. In kleinen Netzwerken laufen Internet-Gateway (Router) und Firewall meist auf einer Maschine. Außer einer geringeren Ausfallsicherheit spricht nichts dagegen, auch den OpenVPN-Server auf dem Gateway laufen zu lassen. Diese Konfiguration ist mit weniger Aufwand verbunden, da der Verwalter keine Route zwischen VPN-Server und Gateway setzen muss.

Laufen Gateway und VPN-Server auf verschiedenen Maschinen, ist explizit eine Route auf dem Gateway zu setzen, die den VPN-Traffic aus *192.168.4.0/24* an den OpenVPN-Server weiterreicht. Fehlt die Route, weiß das Gateway nicht, was es mit IP-Paketen aus *192.168.4.0/24* anfangen soll und verwirft diese. Was in Richtung zum VPN-Server zwingend ist, muss auch andersherum konfiguriert werden. Ist der VPN-Client nicht gleichzeitig das Gateway für die anderen Clients im LAN, muss das Gateway eine Route haben, die selbsttätig alle virtuellen Subnetze zum entsprechenden Client leitet.

Was sonst noch geht

Wenn VPN-Clients dynamische IP-Adressen verwenden, stellt dies kein Problem dar. Anders beim VPN-Server. Dessen IP-Adresse benötigt der Client über die Direktive *remoteip* in *client.conf*, um den Server zu finden. Nutzt der Server eine dynamische IP-Adresse, findet der Client nach einem Refresh der Adresse den Server nicht mehr. Dies lässt sich mit einem Dynamischen-DNS-Client verhindern. Dieser legt einen temporären DNS-Record für eine IP-Adresse an und aktualisiert den Verweis, sobald sich die IP-Adresse ändert. Viele Soho-Router enthalten dynamisches DNS. Für einen Eigenbau-Router lässt sich der Vorgang über einen Dyn-DNS-Client wie *ddclient (http://ddclient.sourceforge.net/)* steuern.

Der Verwalter kann Clients auch zu einem Samba-Server leiten, um Dateien auszutauschen. Verwendet der Administrator */dev/tun*, setzt er eine Route zum Samba-Server und konfiguriert diesen so, dass die Clients aus dem VPN-Subnetz Zugriff auf den Server erhalten. Dazu sind die Samba-Direktiven *interfaces* sowie *host allow* entsprechend anzupassen. Kürzer geht es mit */dev/tap*, da sich der Server in dieser Konfiguration meist im gleichen Ethernet-Subnetz befindet wie die Clients.

Höhere Sicherheit garantiert das verbindungslose UDP als Transportprotokoll. Es erschwert DoS-Attacken sowie Port-Scans. Zur Sicherung des Host-Systems lässt sich der OpenVPN-Server in eine *chroot*-Umgebung einsperren. Root-Privilegien deaktiviert der Verwalter, indem er Benutzer und Gruppe auf den Pseudo-User *nobody* mappt.

Droht Gefahr durch OpenVPN?

OpenVPN schützt nicht nur die Kommunikation eines Unternehmens. Der Anwender kann es auch konfigurieren, um Kontakt zu einem entfernten OpenVPN-Server aufzunehmen, ohne dass die Firmen-Firewall (und damit der Administrator) etwas davon bemerkt und die Verbindung sperrt. Der Benutzer untertunnelt dazu die Firewall.

Die Konfiguration setzt TCP als Transportprotokoll voraus und lässt den Client die VPN-Verbindung aufbauen. Wichtig dabei ist, die Bandbreite des Upstream-Kanals auf einen geringen Wert, beispielsweise 64 KBit/s (*MAXRATE=8000*) zu begrenzen, um nicht durch den verursachten Traffic aufzufallen. Ebenfalls nicht zu vergessen ist die Option *—persist-tun*, um das Schließen des TUN-Interfaces zu verhindern, wenn der Tunnel gekappt wird. Ansonsten müssten alle Routen neu gesetzt werden. Untertunnelt wird die Firmen-Firewall dadurch, dass der Anwender eine statische Route zum VPN-Server setzt, wodurch die Verbindung unabhängig von der Default-Route ist.

Das Verfahren eignet sich ideal, um von einem öffentlichen Zugang wie einem Internet-Café auf Daten im heimischen Netzwerk zuzugreifen. Die Konfiguration lässt sich allerdings auch missbrauchen, um Daten unbemerkt aus dem Firmennetz zu schleusen. Der Administrator hat so gut wie keine Chance, den Tunnel zu entdecken: Da der Traffic per SSL verschlüsselt ist, lässt sich der Verkehr nicht analysieren. Einen Anhaltspunkt für illegal hergestellte Tunnel wie oben können die Verbindungsdauer oder die Tageszeit, zu der die Verbindung besteht, liefern.

Weitere Informationen

Der Workshop konnte bei weitem nicht alle Möglichkeiten erfassen, die OpenVPN offeriert. Wer mehr Informationen benötigt, um beispielsweise ganz spezielle Server-Konfigurationen anzupassen, öffnet im Webbrowser das ausgezeichnete How-to auf *http://openvpn.net/howto.html*. Das englische Dokument beschreibt zahlreiche verschiedene Konfigurationen. Deutsche Workshops lassen sich zudem leicht per Suchmaschine auffinden. ■

OpenVPN für Windows und Mac OS X

Als wesentlicher Vorteil gegenüber vielen anderen VPN-Lösungen zeigt sich, dass OpenVPN 2.0 auch mit Windows oder Mac OS X als Server oder Client funktioniert. Für beide steht sogar ein grafischer Client zur Verfügung, der sich auch von weniger versierten Benutzern nach kurzer Einführung bedienen lässt. Der Windows-Anwender verbindet sich über ein Icon im System-Tray mit einem Server. Die Kommunikation mit Windows erfolgt über die TAP-Win32-Schnittstelle.

Die Pakete, Installationsanweisungen sowie spezielle Einstellungen wie Rechte für beide Betriebssysteme sind auf der Website des OpenVPN-Projekts *(http://openvpn.net/)* zu finden. Das How-to geht detailliert auf die Unterschiede zwischen den Unix-Versionen und den Releases für Microsoft respektive Apple ein.

Mit dem Windows-Client für OpenVPN 2.0 stellt der Anwender die Verbindung über ein Menü im System-Tray her

Weitere Interessante Artikel aus der Linux Professionell

Aktuell

Geladen: APC Mobile Power Pack
Mehr Gizmodo-Videos | RSS | bei iTunes abonnieren

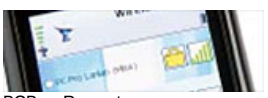
Linux Professionell

LINUX PROFESSIONELL Distributionen
Die Übersicht!
9 Linux Distributionen

PCPro Top-Themen

PCPro-Report
Weihnachten: Online-Bezahlsysteme verstehen

PCPro-Report
Internet Security Report: Unsichere Heim-PCs



PCPro-Report
Basiswissen VoIP: VoIP-Telefon installieren

Remote Linux per X-Server
So fern und doch so nah
Das X-Window-System erlaubt dem Anwender, den Desktop von einem Server auf einen Client-PC zu laden. So kann man beispielsweise beliebige Linux-Programme in einemWindows-Fenster ausführen und Wartungsaufgaben auf dem Server erledigen. **> weiter**

Linux als Terminal-Server
Remote-Computing
Mehr als einen PC mit Netzwerkkarte benötigt man nicht, umalte Rechner mit dem Projekt Linux Terminal Server als Thin-Clients zu konfigurieren. Die Rechenleistung spielt keine Rolle, denn das eigentliche Computing wird dem Server überlassen. **> weiter**

Linux als SSH-Server
Verbindungen sichern
Um eine sichere Verbindung zu einem Server aufzubauen, ist reines TCP der falscheWeg. Besser ist eine verschlüsselte Kommunikation über die Secure Shell. **> weiter**

Server-Tools und -Dienste
Die Red-Hat-Familie
Rund um Enterprise Linux hat der Hersteller Red Hat mit ergänzenden Werkzeugen und Diensten ein komplexes Portfolio für den Profi im Programm. Linux Professionell stellt die wichtigsten Komponenten dieser Open-Source- Architektur vor. **> weiter**

Linux als Proxy
Schnelle Webseiten

Evolution des Filesharing
Netzwerk-Dateisysteme

Linux als Mail-Server
Post vom Pinguin

Global File System
Verteilte Datei-Systeme

FTP-Lieferservice mit Selbstbedienung
Linux als FTP-Server

Virtualisierung mit Xen
Parallele Systeme

> Zur Praxis Gesamtübersicht

Google-Anzeigen

Monitoring On Demand
Ohne Konfigurationsaufwand inkl. DMZ Funktionalität
www.2xperts.de

The Groupware Linux
Email, Kalender & Groupware Server In 30 Minuten einsatzbereit.
www.instantOGo.com

Netzwerk mit WLAN sichern
Einbruchsicheres Netzwerk: W-LAN Tipps für Administratoren.
www.computerwissen.de

Linux als Proxy: Schnelle Webseiten
Mit Squid zwischenWebserver und Browser lassen sichWebseiten schneller ausliefern, als wenn der Client denWebserver direkt anspricht. Der Proxy behält die Seiten im Cache und entlastet somit denWebserver.

Evolution des Filesharing: Netzwerk-Dateisysteme
Die Version 4 des Netzwerk-Dateisystems NFS läutet eine neu Ära beim Datenaustausch ein. NFSv4 erlaubt die sichere Kommunikation im Intra- und Internet, vereinfacht die Verwaltung mit einem Pseudodateisystem und beschleunigt den Datentransfer durch Client-Caching.

Linux als Mail-Server: Post vom Pinguin
Mit den Programmen und Modulen Postfix, Fetchmail und Imapd lässt sich im Handumdrehen ein Post- Office für kleine Unternehmen bis 50 Mitarbeiter aufsetzen.

Global File System: Verteilte Datei-Systeme
Der Zusammenschluss mehrerer Linux-Server zu einem Cluster erhöht die Skalierbarkeit und Ausfallsicherheit. Das freie Global File System von Red Hat verwaltet die Zugriffe und sorgt dafür, dass die Daten allen verbundenen Rechnern zeitgleich und direkt zur Verfügung stehen.

FTP-Lieferservice mit Selbstbedienung: Linux als FTP-Server
Die unsicheren Zeiten von FTP sind längst passé. Die Server-Anwendung ProFTPd verschlüsselt Account und Nutzdaten und lässt den Firewall-Administrator dank des Passive Modes gezielt Ports öffnen. Mit einem FTP-Client können Anwender Daten auf und vom FTP-Server transferieren.

Virtualisierung mit Xen: Parallele Systeme
Mit Xen können Administratoren mehrere Betriebssysteme inklusive aller benötigten Dienste abgeschottet voneinander auf derselben Maschine betreiben. Die kostenlose Virtualisierungs- Engine schickt sich an, kommerzielle virtuelle Maschinen wie VMware abzulösen.

Linux als VoIP-Server: Kostenersparnis durch IP-Telefonie
Mit Asterisk kommunizieren Unternehmen per IP-Telefon innerhalb des lokalen Netzwerks. Ausgehende Anrufe routet Asterisk an einen VoIP-Provider oder in das Festnetz. Rein IP-basierte Gespräche sind kostenlos - für Firmen mit mehreren Standorten eine riesige Kostenersparnis.

Linux-Fileserver: Linux und Windows im Duett
Samba erweitert einen Linux-Rechner zumDatei- und Print-Server. Linux- und Windows-Anwender tauschen darüber Dateien aus oder nutzen gemeinsam einen Drucker im Netzwerk.

DHCP für Subnetze konfigurieren: Virtuell getrenntes LAN
DHCP-Server können verschiedene Subnetze mit dynamischen oder statischen IP-Adressen versorgen, wodurch der Administrator die Anschaffung eines teuren VLAN-Switches vermeidet.

Contentfilter für Linux: Saubere Inhalte
Contentfilter sind heute für jedes Unternehmen unerlässlich - egal ob als Appliance- oder Software- Lösung. Die IT-Netze bleiben damit vor unerwünschtenWebsites und fragwürdigen Inhalten geschützt. Wir stellen Ihnen die wichtigsten Produkte für Linux vor.

Im Tunnel durchs Netz: Linux als VPN-Server
Mit OpenVPN 2.0 erhalten Anwender eine kostenlose VPN-Implementierung auf Basis des Secure Socket Layers. Das System ist leicht zu konfigurieren. Linux Professionell zeigt die Einrichtung des Servers und die Verbindung per Client-PC.

Linux-Server im Unternehmen: Linux Anwenderberichte
Verstärkt werden Linux-Server auch in Unternehmensbereichen genutzt, bei denen maximale Datensicherheit und -verfügbarkeit geboten sein muss. In drei Praxisbeispielen zeigt Linux Professionell, was Linux-Server leisten können - und wie Unternehmen gleichzeitig Kosten sparen.

Neu:
Die Profi-Line
jetzt auch im Abo!

Expertenwissen für Profis!

Ihre Abo-Vorteile:

- Sie sparen Geld.
- Sie verpassen keine Ausgabe.
- Lieferung bequem frei Haus
- **Ohne Risiko:** Nach den ersten zwei Ausgaben jederzeit kündbar.

2 Ausgaben testen,

über 33% sparen!

Wählen Sie jetzt unter

www.pcpro.de/profi

die für Sie richtige Professionell-Reihe aus!