

Linux im Netzwerk - ein "Mini"-Tutorial (Teil 1)

[Jana Jaeger](#)

Eigentlich war dieser Artikel als eine Einführung in das Thema Linux-Firewalls geplant - dann stellte sich aber heraus, daß die schönsten Erklärungen zu Firewalls & Co. ohne die entsprechenden Grundlagen zum Thema Netzwerke wenig wert sind. Deshalb also unsere kleine Reihe zu Linux im Netzwerk. Wo wir nicht in die Tiefe gehen wollen/können, werden wir uns bemühen, auf die entsprechenden Websites oder Bücher hinzuweisen. Hier kommen dann unsere Leser ins Spiel: Wir werden sicher nicht alles abdecken können, deshalb sind wir für Erweiterungen dieses Tutorials oder weitere Links etc. sehr dankbar.

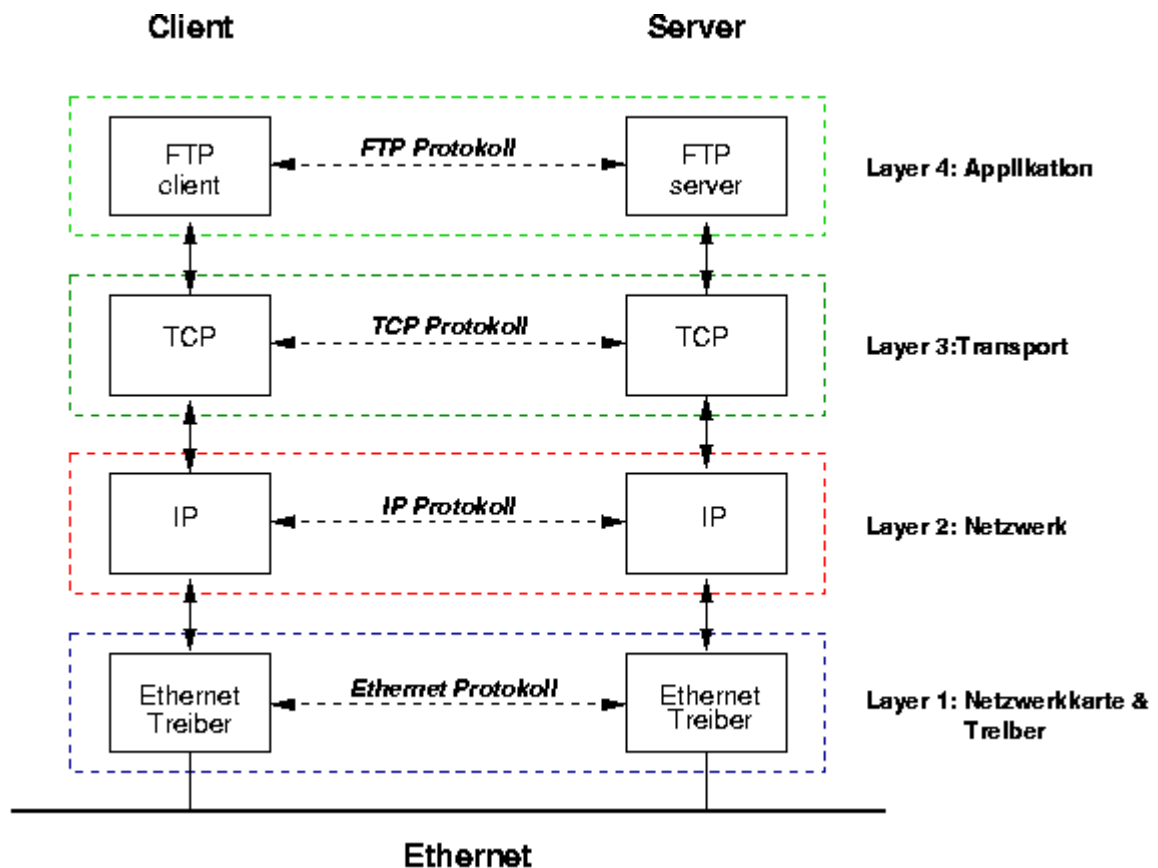
In diesem Sinne - viel Spaß mit unserem Mini-Tutorial!

Teil 1: Wie redet mein Rechner mit der Welt?

Die Sprache, in der sich alle Computer dieser Welt über Hardware- und Betriebssystemgrenzen hinweg verständigen, heißt TCP/IP. Genau genommen ist TCP/IP nicht ein einziges Protokoll, sondern eine ganze Familie von Protokollen. Seinen Ursprung hat TCP/IP in einigen Projekten des US-Verteidigungsministeriums Ende der 60er Jahre. Seit Anfang der 90er hat es sich zum mittlerweile meistverwendeten Netzwerkprotokoll überhaupt gemausert.

TCP/IP - Ein Überblick

Die Protokollfamilie TCP/IP besteht aus mehreren "Familienmitgliedern", die in einer mehrschichtigen Struktur organisiert sind. Diese Schichten werden oft auch neudeutsch "Layer" genannt.



Die Abbildung stellt - für das LAN vereinfacht, die vier Layer der TCP/IP-Familie und den Ablauf einer FTP-Anfrage dar. Auf unterster Ebene (Layer 1) geht es um die reine Kommunikation auf Hardwareebene. Die Netzwerkkarten sowie die im Betriebssystem enthaltenen Treiber fallen in dieses Layer (im englischen Sprachraum existieren auch die Bezeichnungen "link layer", "data link layer" oder "network interface layer"). Hier unterhält sich die Hardware miteinander.

Die Ebene darüber wird als Netzwerk-Layer bezeichnet. Hier wird das "Herumschicken" von Daten im Netz geregelt. Der Datenverkehr ins Internet wird auch auf dieser Ebene abgewickelt (geroutet). Die wichtigen Protokolle:

- **IP** Internet Protocol. Es sorgt für die korrekte Adressierung und stellt einige wichtige Informationen bereit, die ein richtiges Routen der Pakete gewährleisten. Mehr zu IP später in Abschnitt "[Adressen im Internet](#)".
- **ICMP** Internet Control Message Protocol. Per ICMP können beispielsweise Datenpakete versandt werden, die ihren alleinigen Zweck darin haben, Fehler oder andere wichtige Informationen, die die Informationsübermittlung via IP betreffen, an den Versender der Daten zurückzumelden. Beispielsweise bedienen sich die beiden zur Netzwerkanalyse sehr wichtigen Tools *ping* und *traceroute* bestimmter ICMP-Funktionen.
- **IGMP** Internet Group Management Protocol.

Im dritten Layer finden sich die eigentlichen "Transporteure", die für den Datenfluß zwischen zwei Rechnern und den entsprechenden Applikationen sorgen. Die beiden wichtigen Kandidaten auf dieser Ebene:

- **TCP** Transmission Control Protocol. TCP sorgt für einen reibungslosen Transport der Daten vom Absender zum Empfänger. Es stückelt die von einer Applikation bereitgestellte Datenmenge in vernünftige Happen und kontrolliert, ob sie am anderen Ende auch wirklich ankommen etc. Die Applikation, die auf TCP aufbaut, braucht sich folglich um diese Details nicht mehr zu kümmern.
- **UDP** User Datagram Protocol. UDP ist auf schnelle Datenübertragung hin optimiert. Aus diesem Grund entfällt die bei TCP implementierte Kontrolle auf Korrektheit des Datenversands. Solche Überprüfungen müssen, falls sie gewünscht werden, bei Verwendung von UDP auf

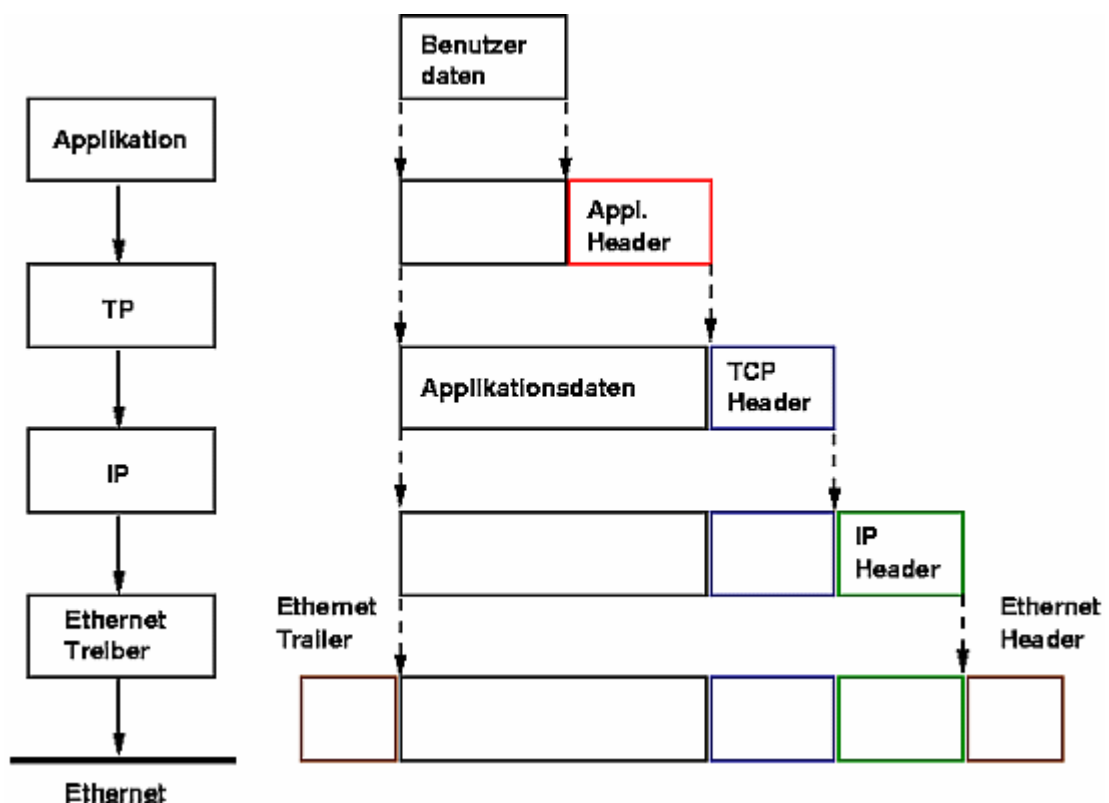
Anwendungsebene vorgenommen werden.

Das vierte Layer ist schließlich die Anwendungs- oder Applikationsebene. Einige der wichtigsten TCP/IP-Applikationen seien hier kurz vorgestellt:

- **FTP** File Transfer Protocol. FTP ist für die Dateiübertragung zwischen zwei Rechnern gedacht.
- **SNMP** Simple Network Management Protocol. SNMP dient zur Bündelung netzwerkrelevanter Daten (Anomalien, Ports nicht erreichbar, etc.).
- **Telnet** Protokoll zum Einloggen auf externen Rechner.
- **X Window System** Eine Sammlung von Programmen, Protokollen und Routinen zur Verwaltung einer grafischen Benutzeroberfläche.
- **NFS** Network File System. Per NFS können Dateisysteme über das gesamte Netzwerk gemountet werden.
- **SMTP** Simple Mail Transfer Protocol. E-Mail Zustellung
- **DNS** Domain Name Service. Zur Übersetzung von Netzwerkknoten in konkrete IP-Adressen.

So, und nun zur zweiten Aussage der Abbildung. Wie läuft eine FTP-Anfrage ab?

Ganz grob unterteilt sich die Abbildung in Client- und Serverseite. Für jedes Protokoll in den einzelnen Layers auf Clientseite existiert auf Serverseite das entsprechende Gegenstück (engl. *peer*). Sendet jetzt der Client eine FTP-Anfrage, reicht die Applikation "FTP client" diese Anfrage an die Transportebene weiter. Hier wird das Datenpaket um eine weitere Informationseinheit erweitert, den TCP Header, in dem wichtige Informationen (Absender, Empfänger, Quell-/Zielpport, etc.) zur Weiterleitung der Anfrage bzw. für den Datentransport enthalten sind.



Das um den TCP Header erweiterte Datenpaket wird für das Routing an das IP-Layer weitergereicht und wieder um einen Header mit wichtigen Informationen zum Routing, dem IP-Header, versehen. Schließlich wird das Datenpaket an die Netzwerkschicht (damit dann endlich an die Hardware) weitergereicht und mit konkreten Netzwerkinformationen (Ethernet-Header und Ethernet-Trailer) versehen. So kann das Datenpaket auf dem Zielrechner über Netzwerk-, IP-, TCP- und Applikationslayer korrekt übermittelt werden. Auf die einzelnen Schichten bezogen sieht es für die beiden "peers" so aus, als würden sie beide miteinander direkt über ihr eigenes Protokoll kommunizieren. Implizit ist aber hierbei ein Durchwandern der jeweils darunter liegenden Layer gemeint.

Adressen im Internet - IP, und was damit zu tun hat

Der Empfänger der Datenpakete bekommt die Sendungen an seine IP-Adresse (IPv4 bedeutet 32 Bit-Adressraum, IPv6 128 Bit). Eine IPv4 Netzwerkadresse enthält vier Byte Daten. In der konventionellen, etwas besser lesbaren Darstellung sind die vier Byte jeweils durch Punkte getrennt und in Dezimalschreibweise angegeben. Ein Beispiel:

```
192.168.27.1
11000000 10101000 00011011 00000001
```

Historisch wurden die IP-Adressen bis Mitte der 90er Jahre in mehrere Unterklassen aufgeteilt. Die Adresse kann grob in zwei Teile gegliedert werden, die Netzwerknummer und die Hostnummer des Rechners im lokalen Netz. Je nach Klasse machten das erste (A), die ersten beiden (B) oder die ersten drei Byte (C) die Netzwerknummer aus. Demnach gab es nur einige wenige Klasse A-Netzwerke (0-127), die dann aber entsprechend eine riesengroße Anzahl von Hosts im lokalen Netzwerk anbinden konnten. Die restlichen Klassen erlaubten dementsprechend eine größere Anzahl der Netze, allerdings mit weniger Hosts.

Mittlerweile hat sich diese Art der Klassifizierung von Netzwerken aber überholt. Man spricht mittlerweile von klassenlosen Adressen (engl. "classless addresses") und die oben beschriebene Aufteilung ist somit hinfällig. Eine klassenlose IP-Adresse besteht aus einer variablen Anzahl fester Bits für die Angabe des Netzwerks, die restlichen Bits bezeichnen die einzelnen Hosts. Die alte Notierung in Klassen fügt sich als Spezialfall in diese Regelung ein. Netzmasken für die Einbindung von Subnetzen ins gesamte Internet werden einfach durch die Anzahl der fixen Bits spezifiziert. Klassenlose IP-Adressen (IPv4) werden in der altbekannten Zahlengruppen- und Punktschreibweise angegeben, wobei am Ende der Adresse durch einen Slash abgetrennt, die Anzahl der fixen Bits (prefix) notiert wird. Die neuere IP-Version, IPv6 benutzt ausschließlich klassenlose Adressen und folgt etwas anderen Schreibkonventionen als die herkömmlichen IPv4-Adressen. Eine IPv6 Adresse besteht aus acht 16-Bit Hexadezimalzahlen, die durch ":" voneinander getrennt werden. Eine Folge von Nullen wird hierbei als "::" Als Beispiel für eine "alte" IPv4-Adresse in der neuen Schreibweise muß wieder unser Beispiel von oben herhalten:

```
IPv4-Schreibweise      : 192.168.27.1
IPv6 (ausgeschrieben) : 0:0:0:0:0:FFFF:192.168.27.1
IPv6 (vereinfacht)    : ::FFFF:192.168.27.1
```

Noch unterstützen längst nicht alle Programme IPv6, weshalb ein Nebeneinander von alter und neuer Konvention gewährleistet bleiben muß. Für die Konvertierung einer IPv4 in eine IPv6-Adresse bedeutet dies, daß die ersten 80 Bit auf Null gesetzt sind, die darauf folgenden 16 Bit auf Eins gesetzt sind, und die verbleibenden 32 Bit von der alten Adresse eingenommen werden.

Wie das Routing (Weiterleiten) anhand von Netzmasken und IP-Adressen im Einzelnen geschieht, soll an folgendem Beispiel verdeutlicht werden. Der Einfachheit halber werden hier IPv4-Adressen verwendet, das Prinzip bleibt für IPv6 das gleiche ;-):

```
Absender 192.168.27.1/24 = 11000000 10101000 00011011 00000001
Netzmaske 255.255.255.0  = 11111111 11111111 11111111 00000000

Empfänger 192.140.6.2/26 = 11000000 10001100 00000110 00000010
Netzmaske 255.255.255.192 = 11111111 11111111 11111111 11000000
```

Der Router hat intern eine Reihe wichtiger Informationen in einer Routingtabelle gespeichert. Hierbei besonders wichtig: die Netzmaske und die IPs der angebotenen Rechner. Kommt ein Paket mit oben genannter Absender- und Empfängeradresse an, muß der Router anhand der Netzmaske entscheiden, ob sich der Empfänger möglicherweise im gleichen Netz befindet wie der Absender, oder ob das Paket weitergereicht und ins Internet verschickt werden soll. Das sieht im einzelnen so aus, daß Netzmaske und Adresse mit einem logisches UND verknüpft werden, wie im Beispiel zu sehen:

```

Empfänger 192.140.6.2      = 11000000 10001100 00000110 00000010
Netzmaske 255.255.255.192 = 11111111 11111111 11111111 11000000
-----
Ergebnis der Verknüpfung: 11000000 10001100 00000110 00000000
" in Dezimalpunktdarstellung: 192.      140.      6.      0

Absender 192.168.27.1     = 11000000 10101000 00011011 00000001
Netzmaske 255.255.255.0   = 11111111 11111111 11111111 00000000
-----
Ergebnis der Verknüpfung: 11000000 10101000 00011011 00000000
" in Dezimalpunktdarstellung: 192.      168.      27.      0

```

Nach dieser Operation wird dem Router "klar", daß sich Empfänger und Absender nicht im gleichen Netz befinden, und das Paket weiter nach außen geleitet werden muß.

Theoretisch stehen bei Verwendung von IPv4 mit einem Adressraum von 32 Bit 2^{32} (ca. 4 Milliarden) Adressen zur Verfügung. Berücksichtigt man hierbei aber zum einen die festgelegte Bedeutung mancher Adressen und die Aufteilung in Netze, wird der Adressraum deutlich kleiner. Hinzu kommt noch, daß Entwicklungen wie die Anbindung von Handys, Kühlschränken o.ä. ans Internet ebenfalls zur Verknappung von IP-Adressen beiträgt. Noch ganz davon zu schweigen, daß es mittlerweile mehr Menschen als (mit IPv4) mögliche Adressen gibt, denen das Internet als Kommunikationsmedium zunehmend wichtiger wird.

Aus dieser Not entstand ein wichtige Konvention, die besonders auch aus Sicherheitsaspekten interessant ist - das **Masquerading**. Das Prinzip dahinter ist vereinfacht geschildert, daß nur ein Rechner mit seiner IP-Adresse dem großen weiten Internet bekannt ist, während sich die anderen Rechner des dahinter angeschlossenen LANs quasi hinter seinem Rücken verstecken und ein eigenes privates Netzwerk bilden. Die so "maskierten" Maschinen tragen IP-Adressen, die nicht nach außen hin bekannt sind (also auch nicht den Adressraum der weltweiten IPs verringern). Das ursprünglich unbeabsichtigte Plus an Sicherheit durch ein derartiges Maskieren von IP-Adressen wird dadurch gewonnen, daß Rechner, deren IP nach außen hin nicht bekannt ist, auch nicht zum Angriffsziel einer gezielten Attacke werden können. Ist allerdings ein Angreifer erst einmal in das lokale Netz eingedrungen, stehen diese Rechner schutzlos da.

Ports

Jede Internetverbindung kann mit genau vier Nummern eindeutig charakterisiert werden. Diese vier Nummern sind:

1. IP-Adresse des sendenden Teilnehmers (siehe oben)
2. Portnummer des sendenden Teilnehmers (16 Bit)
3. IP-Adresse des Empfängers (siehe oben)
4. Portnummer des Empfängers (16 Bit)

Zwei Rechner können miteinander über insgesamt 65536 (durchnummeriert von 0 bis 65535) Ports miteinander kommunizieren. Die Informationen über Quell- und Zielpport eines Paketes - egal ob TCP oder UDP - werden den Paketen im Header mitgegeben. Jedem der bekannten Netzdienste wird eine eindeutige Portnummer zugewiesen. Die Riesensmenge der vorhandenen Ports wird in zwei Gruppen unterteilt:

1. **privilegierte Ports:** Die Ports von 0 bis 1023 sind mit festen, definierten Diensten belegt (ein Blick in die Datei `/etc/services` verrät, mit welchen). Unter UNIX gibt es die Konvention, daß nur der Superuser (`root`) auf diesen Ports "lauschen" darf oder von dort aus Verbindungen nach außen initiieren darf. Ein kleiner Ausschnitt:

```

ftp          21/tcp      # File Transfer [Control]
fsp          21/udp      # UDP File Transfer
ssh          22/tcp      # SSH Remote Login Protocol

```

```

ssh                22/udp           # SSH Remote Login Protocol
telnet             23/tcp           # Telnet
telnet             23/udp           # Telnet
#                 24/tcp           any private mail system
#                 24/udp           any private mail system
smtp               25/tcp           mail              # Simple Mail Transfer
smtp               25/udp           mail              # Simple Mail Transfer

```

2. **nicht privilegierte Ports:** Alle Ports mit den Nummern 1024 bis 65535, hier tummeln sich andere Dienste, die nicht vom Superuser mit root Rechten aufgesetzt sein müssen. Beispiel: Ein Apache-Webserver auf Port 8080, o.ä.

Wichtige Informationen zum Thema

Tieferen Einblick in das Netzwerkgeschehen und die TCP/IP-Internia geben die folgenden Websites und Bücher.

Bücher

- UNIX Network Programming, Volume 1: Networking APIs - Sockets and XTI von W. Richard Stevens, erschienen bei Prentice Hall; ISBN: 013490012X
- TCP/IP Illustrated, Volume 1 : The Protocols (Addison-Wesley Professional Computing Series) von W. Richard Stevens, erschienen bei Addison-Wesley Pub Co; ISBN: 0201633469
Sehr ausführliche Erläuterung zu TCP/IP- und Netzwerkproblematik. Teilweise wird sie für den Anfänger aber zu technisch.
- [Linux Netzwerke](#) von Dr. Stefan Fischer und Ulrich Walther, erschienen bei SuSE PRESS, ISBN 3934678203. Guter Einstieg in die Thematik Linux im Netzwerk mit wichtigen Tips für den Praktiker.
- [Practical Unix and Internet Security](#) von Simson Garfinkel und Gene Spafford, erschienen bei O'Reilly & Associates; ISBN: 1565921488
Die "Bibel" für alle, die einen groben Überblick über UNIX-Security brauchen.

Websites

- http://www.private.org.il/tcpip_rl.html
Eine sehr (!!!) ausführliche und gut gegliederte Link- und Infosammlung zu TCP/IP.
- <http://www.yale.edu/pclt/COMM/TCPIP.HTM>
Kurz und knapp, die wichtigsten Details zu TCP/IP.
- http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/ip.htm#xtocid2236323
Eine sehr gute und verständliche Dokumentation zu Internet Protokollen (IP, ICMP und TCP). 