Wiki / netcat

Dieser Artikel wurde für die folgenden Ubuntu-Versionen getestet:

- **Ubuntu 12.04** Precise Pangolin

Zum Verständnis dieses Artikels sind folgende Seiten hilfreich:

1. **Ein Terminal öffnen**
2. **Root-Rechte**

Inhaltsverzeichnis

Netcat, kurz **nc**, ist ein universelles Kommandozeilenwerkzeug. Es kann im Terminal oder in Skripten für die Kommunikation über **TCP** [http://de.wikipedia.org/wiki/Transmission_Control_Protocol]- und **UDP** [http://de.wikipedia.org/wiki/User_Datagram_Protocol]-Netzwerkverbindungen (**Internet Protocol** [http://de.wikipedia.org/wiki/Internet%20Protocol] Version 4 und **Version 6** [http://wiki.ubuntuusers.de/IPv6]), aber auch lokale **UNIX Domain Sockets** [http://de.wikipedia.org/wiki/Unix_Domain_Socket] genutzt werden. Damit eignet es sich für zahlreiche Anwendungsfälle zum Thema Netzwerkverbindungen.

Installation

Das Paket **netcat-openbsd** ist Bestandteil jeder Ubuntuinstallation, kann aber ansonsten über das Paket:

- **netcat-openbsd**

installiert werden. Alternativ können auch die Pakete **netcat-traditional** [http://packages.ubuntu.com/netcat-traditional] oder **netcat6** [http://packages.ubuntu.com/netcat6] verwendet werden. Die bevorzugte Variante kann dann mit dem **Alternativen-System** [http://wiki.ubuntuusers.de/Alternativen-System] eingestellt werden.

Aufruf

Netcat wird mit dem Befehl [1]:

```
nc [OPTIONEN] HOST PORT
```

aufgerufen, um sich mit einem bereits laufenden Server zu verbinden. Um stattdessen selbst an einem bestimmten Port auf eingehende Verbindungen zu warten, verwendet man:

```
nc [OPTIONEN] -l PORT
```

In diesem Fall können Portnummern unter 1024 nur mit Root-Rechten [2] genutzt werden.

Die bestimmten Varianten von Netcat aus den unterschiedlichen Paketen können auch ganz explizit aufgerufen werden, falls man ein anderes Programm als den eingestellten Standard benutzen möchte:

```
nc.openbsd       # aus netcat-openbsd
nc.traditional   # aus netcat-traditional
nc6              # aus netcat6
```

nc liest jetzt zu sendende Daten von der Standardeingabe *(stdin)* und schreibt empfangene Daten auf die Standardausgabe ("stdout"). Meldungen werden auf die Standardfehlerausgabe ("stderr") geschrieben. So kann nc interaktiv genutzt, aber auch für **Shell-Umleitungen** [http://wiki.ubuntuusers.de/Shell/Umleitungen] eingesetzt werden. **Beispiele** dazu sind weiter unten zu finden.

Die folgende Tabelle gibt einige der Optionen für nc.openbsd an. **Unterschiede** zu den anderen Versionen sind weiter unten zu finden.

Optionen von nc	
Option	**Beschreibung**
-4 / -6	Erzwingt die Nutzung von IP Version 4 bzw. 6.
-u	Nutzt UDP statt TCP.
-U	Nutzt Unix Domain Sockets. Es gelten die **unten beschriebenen Besonderheiten**.
-l	nc wartet auf eingehende Verbindungen ("listen"), anstatt selbst eine Verbindung zu einem anderen Host aufzubauen. Die Angabe eines HOSTs entfällt (siehe oben). Erlaubt die Option -k zu verwenden, sollte aber nicht zusammen mit -p, -s oder -z verwendet werden.
-k	nc wartet nach Verbindungsende auf eine neue Verbindung. Nur in Verbindung mit -l möglich.
-z	nc scannt nur nach lauschenden ("listening") Diensten (Portscan), anstatt Daten zu senden.

-s IP	Gibt an, über welche Netzwerkschnittstelle die Daten gesendet werden sollen.
-p PORT	Gibt den lokalen Port an, von dem aus nc die Verbindung aufbauen soll.
-q SEKUNDEN	nc wartet die angegebene Zeit nach einem **End of File** [http://de.wikipedia.org/wiki/End%20of%20File]-Zeichen (EOF) auf der Standardeingabe und beendet sich dann. Bei einem negativen Wert wartet nc unendlich lange (bis die Netzwerkverbindung getrennt wird).
-w TIMEOUT	Verbindungen, die nicht aufgebaut werden können oder nicht genutzt werden, werden nach TIMEOUT Sekunden beendet. -w hat keine Auswirkung, wenn -l ("listen") verwendet wird.
-v	Aktiviert ausführlichere Ausgaben.

Mehr Optionen findet man in der **Manpage** [http://wiki.ubuntuusers.de/Manpage] von nc.

Unterschiede und Kompatibilität

Die jeweils akzeptierten Optionen unterscheiden sich teilweise. Die oben angegebene Tabelle gilt für nc.openbsd.

- Optionen, die nc.traditional nicht kennt: -4 / -6, -U, -k
- Optionen, die nc6 nicht kennt: -U, -k (aber stattdessen --continuous)

Laut der **Manpage** [http://wiki.ubuntuusers.de/Manpage] für nc.openbsd sollte -p nicht in Verbindung mit -l verwendet werden, aber es scheint dennoch zu funktionieren. Bei nc.traditional und nc6 muss die -p-Option angegeben werden, wenn ein "listen"-Port bestimmt werden soll. Außerdem bricht nc.openbsd ab, wenn es ein **End of File** [http://de.wikipedia.org/wiki/End%20of%20File]-Zeichen (EOF) über die Standardeingabe liest, während dies nc.traditional und nc6 nicht tun. Bei allen dreien kann der Abbruch bei EOF mit -q 0 erzwungen und mit -q -1 deaktiviert werden.

Unix Domain Sockets

nc.traditional und nc6 beherrschen den Umgang mit Unix Domain Sockets nicht. Die Syntax für Unix Domain Sockets ist etwas anders:

```
nc -U [-l] [-u] [WEITERE_OPTIONEN] SOCKET-DATEI
```

nc verbindet sich so mit einem bestehenden Socket bzw. erstellt mit -l einen Socket, auf dem es lauscht. Es verwendet dazu einen stream socket (stateful = zustandsbehaftet, ähnlich TCP). Mit -u werden stattdessen datagram sockets (stateless = zustandslos, ähnlich UDP) genutzt, doch das funktioniert nicht in allen Ubuntu-Versionen.

Beispiele

Hier ein paar simple Beispiele zum Einsatz von nc. Sie sollen die vielfältigen Einsatzmöglichkeiten aufzeigen, jedoch gibt es dabei auch (fast) immer spezialisierte Programme, die bestimmte Aufgaben wesentlich besser lösen (z.B. Datei- oder Webserver).

Simpler Dateitransfer

Wie man schnell ein Dateiarchiv über ein Netzwerk (bzw. zwischen Host und Gast einer Virtuellen Maschine) übertragen kann, ist im **QEMU-Artikel** [http://wiki.ubuntuusers.de/QEMU#Einzelne-Dateien] beschrieben. Ein anderes Beispiel soll hier noch gegeben werden. Dazu führt man folgenden Befehl auf dem Rechner aus, der die Datei empfangen soll:

```
nc -l 8080 > ZIELDATEI
```

Nun auf dem Computer, der die Datei senden soll. Dateinamen, IP-Adresse und ggf. auch die Portnummern müssen dabei passend ersetzt werden.

```
nc 192.0.2.123 8080 < QUELLDATEI
```

Zu beachten bleibt, dass die Datei ohne Verschlüsselung oder andere Sicherheitsmaßnahmen übertragen wird. Dafür sollten andere Werkzeuge wie z.B. **scp** [http://wiki.ubuntuusers.de/SSH#scp] verwendet werden.

Manuell mit einem Server "sprechen"

Möchte man genau sehen, was ein Server antwortet, um z.B. Fehler zu beheben, kann man auch dazu nc benutzen. Beispiel Webserver:

```
echo -e "GET / HTTP/1.1\nHost: ubuntuusers.de\n" | nc ubuntuusers.de 80
```

Hier das Versenden einer Mail per SMTP und **Postfix** [http://wiki.ubuntuusers.de/Postfix]. Die Zeilen, die mit einer Zahl beginnen, sind Serverantworten:

```
nc localhost 25
```

```
220 A ESMTP Postfix (Ubuntu)
HELO localhost
250 A
MAIL FROM:<benutzer@localhost>
250 2.1.0 Ok
RCPT TO:<benutzer@localhost>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Dies ist eine Test-E-Mail.
```

Portscan

Auch simple Portscans sind mit nc möglich. Diese Beispiele zeigen auch, wie man mehrere Ports gleichzeitig angibt.

- Port-Bereich scannen:

```
nc -zv localhost 22-25
```

```
nc: connect to localhost port 22 (tcp) failed: Connection refused
nc: connect to localhost port 23 (tcp) failed: Connection refused
nc: connect to localhost port 24 (tcp) failed: Connection refused
Connection to localhost 25 port [tcp/smtp] succeeded!
```

- Zwei einzelne Ports scannen:

```
nc -zv localhost 53 631
```

```
nc: connect to localhost port 53 (tcp) failed: Connection refused
Connection to localhost 631 port [tcp/ipp] succeeded!
```

Links

- **Wikipedia** [http://de.wikipedia.org/wiki/Netcat]
- **Shell/Befehlsübersicht** [http://wiki.ubuntuusers.de/Shell/Befehls%C3%BCbersicht] - Übersicht über verschiedene Shellbefehle

---

Diese Revision [http://wiki.ubuntuusers.de/netcat?rev=720072] wurde am 2. Mai 2014 00:06 von **ubot** erstellt.
Die folgenden Schlagworte wurden dem Artikel zugewiesen: **Netzwerk** [http://wiki.ubuntuusers.de/Wiki/Tags?tag=Netzwerk], **Shell** [http://wiki.ubuntuusers.de/Wiki/Tags?tag=Shell]