Linux-Server an ein hochverfügbares Netzwerk anbinden

# Doppelt reißfest

Wenn in einem hochverfügbaren Netzwerk eine Komponente ausfällt, darf dies die Funktionstüchtigkeit des Netzes nicht beeinträchtigen. Der Bonding-Kerneltreiber auf Servern schaltet dafür automatisch zwischen mehreren Netzwerk-Schnittstellen des Servers um, die an jeweils anderen Switches hängen. Clifford Wolf


Knack! – und ein Port am Switch im lokalen Netz fällt aus. Der normale Effekt: Bis der Fehler bemerkt ist und ein Techniker ihn lokalisiert und behebt, vergeht viel wertvolle Zeit. Eine redundante physikalische Verbindung zwischen Server und Switch hätte die Folgen des Hardwaredefekts automatisch abgefangen und kein Netzwerk-Server oder -Client hätte etwas bemerkt.

Zur Verwaltung so einer hochverfügbaren Netzwerkanbindung enthält der Kernel den Bonding-Treiber, der den Traffic automatisch auf eine zweite Ethernet-Netzwerkschnittstelle umleitet – und das sogar transparent für alle angeschlossenen Geräte. Ein Shellskript hilft den Bonding-Verbund überwachen.

## Verkabeln für hochverfügbares Netzwerk

In einem hochverfügbarem Netzwerk übernehmen zwei Switches die Funktion eines einzelnen. Jeder Rechner ist über einen weiteren Netzwerk-Controller mit einem Patchkabel am zweiten Switch angebunden. Dabei entscheidet jeder Rechner auch, welchen Switch er verwendet. Ausgehende Pakete senden die Rechner nur an diesen aktiven Switch. Ankommende Pakete finden auf beiden Interfaces Einlass.

Fällt ein Switch zum Beispiel wegen eines Defekts im Netzteil aus, migrieren alle Server ihre aktiven Interfaces auf den verbleibenden Switch und die Server können weiterhin über das Netzwerk Daten austauschen. Fällt jedoch ein Interface aus, ist das oben beschriebene Netzwerk nicht mehr vollständig: Der betroffene Rechner kann keine Pakete von den anderen entgegennehmen, die mit ihren aktiven Interfaces ebenfalls an diesem Switch hängen. Daher verbindet ein Crossover-Kabel die beiden Switches und bildet einen Switch-Interconnect.

Vorher ist unbedingt zu prüfen, ob die Switches schon über die Uplinks miteinander verbunden sind. Diese Verbindung führt zu einer Schleife, die bei der Verkabelung per Ethernet nicht erlaubt ist.

Wer glaubt, dass der Switch diesen Fehler per Spanning-Tree-Protokoll (STP) [1] abfängt, wiegt sich in trügerischer Sicherheit. STP sollte eine überflüssige Verbindung zwar ignorieren und die Verfügbarkeit des Netzwerks damit erhöhen. In der Praxis bringt jedoch häufig schon ein einziges defektes Gerät im Baum das Protokoll aus dem Tritt und lässt dabei gleich das ganze Rechenzentrum offline gehen.

## Bonding-Treiber aktivieren

Im Kernel implementiert der »Bonding«-Treiber die Funktionalität, um das aktive Netzwerk-Interface zu wechseln. Er legt sich zwischen die Netzwerkkarten-Treiber und TCP/IP (Abbildung 1). Die Dokumentation ist in »linux/Documentation/networking/bonding.txt« zu finden und auch online einzusehen, unter anderem auf [2].

Der Treiber liegt in der Kernel-Config unter »Network device support | Bonding

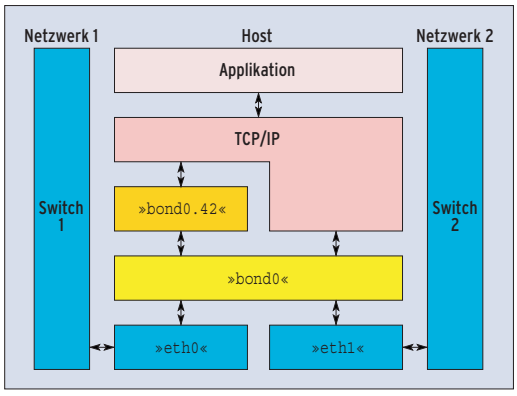


**Abbildung 1: Der Bonding-Treiber fungiert als Vermittlungsschicht zwischen dem Treiber der Netzwerkkkarte und TCP/IP.**

driver support«. Er muss unbedingt als Modul vorliegen, da der Treiber einige Einstellungen als Modulparameter direkt übergeben bekommt. Bei einem monolithisch gebauten Kernel müsste das System bei jeder Konfigurationsänderung des Bonding-Treibers neu starten, was natürlich als Downtime zählt und in einem hochverfügbarem Umfeld generell abträglich ist.

Der wichtigste Parameter für das Bonding-Modul ist der Bonding-Modus, der per »mode = *N*« in der Datei »/etc/modules« hinterlegt ist. Mögliche Werte für *N* sind 0 für Round Robin (Default), 1 für Active Backup und 2 für XOR (siehe Kasten „Die Bonding-Modi"). In einem hochverfügbaren Netzwerk kommt der Modus 1 (Active Backup) zum Einsatz.

## ARP ergänzt MII

Normalerweise erkennt der Bonding-Treiber am MII-Status (Media Independent Interface), ob ein Interface funktioniert. Steckt man das Kabel ab oder schaltet sich der Switch aus, wechselt der MII-Status in den Down-Status und der Bonding-Treiber migriert auf das andere Interface. Das funktioniert jedoch nicht, wenn die Switch-Backplane defekt ist. Die Interfaces tauschen dann zwar keine Daten mehr aus, funktionieren aber noch, also meldet der MII-Status, dass alles in Ordnung sei.

## ARP-Pings konfigurieren

Dieses Problem lässt sich mit einer zweiten Überwachung abfangen: Wenn der Switch für eine gewisse Zeitspanne keine Daten vom Interface empfängt, gilt es als down. Um dabei zu verhindern, dass das Interface den Down-Status auch dann erhält, wenn im Netzwerk zufällig kein Traffic auftritt, schickt der Bonding-Treiber regelmäßig ARP-Anfragen (Address Resolution Protocol) an eine Liste von IP-Adressen. Die entsprechenden ARP-Antworten stellen sicher, dass immer zumindest ein minimaler Traffic am Interface eingeht.

Als Ziel-IPs für die ARP-Pings verwendet der Administrator die Management-IPs der Switches. Es kann aber nicht schaden, auch noch weitere Hosts in die Liste mit aufzunehmen. Die Ziel-IPs kommen zusammen mit der Liste der Modulparameter in die Datei »/etc/modules.conf« (Listing 1). Der Bonding-Treiber erstellt daraufhin ein virtuelles Netzwerk-Interface namens »bond0«, das dann wie eine normale Netzkarte zu konfigurieren ist.

Erst nachdem das Bonding-Device eine IP-Adresse hat, lassen sich ihm die physikalischen Interfaces zuordnen. Diese Reihenfolge hat einen einfachen Hintergrund: Solange das Bonding-Device keine IP-Adresse hat, kann es auch keine ARP-Requests erzeugen. Dann würde der Kernel melden, dass alle dem Bonding-Device zugeordneten Interfaces down wären.

## Interfaces zuordnen

Das Zuordnen der physischen Interfaces erledigt der Verwalter mit dem Tool »ifenslave«, das er am besten selbst kompiliert. Die Distributionsversionen sind in der Regel veraltet oder das Tool fehlt gänzlich (siehe Kasten „Ifenslave kompilieren").

Das »ifenslave«-Tool verlangt den Namen des Bonding-Interface als ersten Parameter und den Namen des physischen Interface als zweiten. Fährt der Admin

### Die Bonding-Modi

Der Bonding-Treiber hat drei Betriebsmodi: Round Robin, Active Backup und XOR. Alle drei akzeptieren Pakete an allen Interfaces. Der Betriebsmodus regelt lediglich, über welches Interface Pakete das System verlassen. Der Round-Robin-Modus – also 0 – ist in Kombination mit Switches, die Link-Aggregation unterstützen, interessant. Er bündelt die Kapazität mehrerer Interfaces zu einer dicken Verbindung, die entsprechend mehr Bandbreite bereitstellt. In einem derartigen Setup sind alle zu einem Bonding-Interface gehörenden physischen Interfaces mit dem gleichen Switch verbunden, zusätzlich ist Link-Aggregation am Switch für die entsprechenden Ports aktiviert. Im Round-Robin-Modus verteilen sich die ausgehenden Pakete gleichmäßig auf alle Interfaces.

Der Active-Backup-Modus (1) ist für hochverfügbare Umgebungen geeignet. Hier ist immer ein Interface das aktive. Das andere akzeptiert zwar Pakete, sendet jedoch keine. Dadurch lassen sich die Interfaces mit unterschiedlichen Switches verbinden, ohne dabei die Forwarding-Tabellen der Switches dadurch zu verwirren, dass Pakete vom selben Rechner ständig aus unterschiedlichen Richtungen kommen. Wenn der Bonding-Treiber das aktive Interface ändert, sieht es für die Switches aus, als wäre der Rechner mit einem anderen Switch verbunden.

Der XOR-Modus (2) verwendet das Resultat einer XOR-Verknüpfung zwischen den Hardware-Adressen des Absenders und des Empfängers eines Pakets, um zu entscheiden, über welches Interface das Paket zu verschicken ist. Der Modus erlaubt es, etwas Ähnliches wie Link-Aggregation mit Switches zu implementieren, die Link-Aggregation nicht direkt unterstützen.

**Listing 1: »/etc/modules.conf«**

```
01 alias bond0 bonding
02 options bond0 mode=1 miimon=0 arp_interval=2000
   arp_ip_target=192.168.100.1,192.168.100.2,192.168.100.3
```

**Listing 2: »/etc/network/interfaces«**

```
01 iface bond0 inet static
02 up ifenslave bond0 eth0
03 up ifenslave bond0 eth1
04 post-down modprobe -r bond0 eth0 eth1
05 address 192.168.100.65
06 netmask 255.255.255.0
07 gateway 192.168.100.1
08 broadcast 192.168.100.255
```

das Bonding-Interface herunter, hebt der Kernel automatisch alle mit ihm verbundenen Verknüpfungen zu physischen Interfaces auf.
Das Problem dabei: Der Bonding-Treiber setzt auf allen ihm zugeordneten physischen Interfaces die gleiche MAC-Adresse. Deshalb sollten sich die Treiber der physischen Interfaces bei einem Shutdown des Bonding-Interface selbst entfernen. Nur so lässt sich sicherstellen, dass das System die physischen Interfaces beim nächsten Setup reinitialisiert und diese wieder die ursprünglichen MAC-Adressen bekommen. Einen entsprechenden Eintrag in der Datei »/etc/network/interfaces« (hier bei Debian) zeigt Listing 2.

## Bonding-Treiber per SNMP überwachen

Der Status des aktiven Bonding-Interface ist in der Datei »/proc/net/bonding/bond0« (Abbildung 2) einzusehen. Natürlich ist es wichtig, dass die Administratoren von dem Ausfall einer Komponente so schnell wie möglich erfahren. Daher nimmt man den Bonding-Status in den SNMP-Tree (Simple Network Management Protocol) auf [3].
SNMP eignet sich ideal, da es die meisten Monitoring-Lösungen unterstützen. Um das Linux-Bonding-Device SNMP-fähig zu machen, genügen das Shellskript »snmp_bond« (Listing 3) und der Net-SNMP-Agent [4], den viele Linux-Distributionen bereits enthalten.
Sobald das Shellskript als »snmp_bond« in »/usr/local/bin/« platziert ist, sorgt eine einzige Zeile am Ende der SNMP-Konfigurationsdatei »/etc/snmp/snmpd.conf« dafür, dass der Bonding-Status in den SNMP-Tree einfließt:

```
exec .1.3.6.1.4.1.2021.170 ➥
/bin/bash /usr/local/bin/snmp_bond
```

Damit ist sofort unter der Objekt-ID ».1.3.6.1.4.1.2021.170.101.1« zu erkennen, ob in letzter Zeit (der Net-SNMP-Daemon hält ein Resultat eine gewisse Zeit lang in einem Cache vor) eine Zustandsänderung aufgetreten ist. Hinter der Zeile »*.2« ist das aktuell aktive Interface und hinter »*.3« der momentane Up- oder Down-Status für alle Interfaces zu sehen (Abbildung 3).

### Ifenslave kompilieren

**Viele Distributionen enthalten entweder kein »ifenslave« oder das Tool ist nur in einer veralteten Version vorhanden. Deshalb sollte man es immer selbst kompilieren. Die Sourcen sind Teil der Kernelquellen. Beim Übersetzen ist vor allem der Pfad zu den richtigen Headern wichtig:**

```
gcc -o /usr/local/sbin/ifenslave ➥
-I /usr/src/linux/include /usr/src/➥
Documentation/networking/ifenslave.c
```

**Der Parameter »-I /usr/src/linux/include« ist deshalb von großer Bedeutung, da der Compiler sonst die oft veralteten Kernelheader aus »/usr/include/linux« verwendet, was zu Problemen im Betrieb führen kann. Enthält die Distribution ein »ifenslave«-Binary, sollte es der Administrator löschen oder umbenennen. Somit stellt er sicher, dass immer das selbst kompilierte »ifenslave« zum Zug kommt.**

## VLAN-Tagging bei zu wenig Ports nutzen

Nun gilt es, nur noch ein Problem zu lösen: Die Anzahl der zur Verfügung stehenden Netzwerk-Interfaces in den Rechnern geschickt aufteilen. Im Projekt enthalten die Rechner bereits Vier-Port-Netzwerkkarten. Die Rechner selbst sind paarweise hochverfügbar ausgelegt und ein Netzwerk-Interface an jedem Rechner ist bereits für die Direktverbindung zum jeweils anderem Rechner im Clusterpaar reserviert.
Die drei übrigen Ports reichen für die Verbindungen zu den beiden Switches aus. Allerdings sind die Firewalls mit bis zu drei Netzen verbunden, was mit der Direktverbindung zwischen den Cluster-Nodes sieben Netzwerk-Interfaces notwendig macht – zu viele.
Um mit den vorhandenen Schnittstellen auszukommen, stellt der Verwalter die einzelnen Netze von physisch getrennten Netzen auf virtuelle Netze mit VLAN-Tagging (IEEE 802.1Q) um. In dieser Konfiguration lassen sich dann mehrere getrennte Netze über nur ein Netz-

**Listing 3: »snmp_bond«**

```
01 #!/bin/bash
02
03 oldstat=/var/run/snmp_bond0_stat
04 exec < /proc/net/bonding/bond0
05
06 active="notfound"
07 main_status="notfound" interface=main
08 eth0_status="notfound" eth0_fcount=-1
09 eth1_status="notfound" eth1_fcount=-1
10
11 while read line; do
12       case "$line" in
13         "Currently Active Slave: "*)
14             active="${line##Currently Active Slave: }" ;;
15         "Slave Interface: "*)
16             interface="${line##Slave Interface: }" ;;
17         "MII Status: "*)
18             eval "${interface}_status=${line##MII Status: }" ;;
19         "Link Failure Count: "*)
20             eval "${interface}_fcount=${line##Link Failure Count: }" ;;
21       esac
22 done
23
24 if [ -f $oldstat ]; then
25       read old_active old_eth0_fcount old_eth1_fcount < $oldstat
26 else
27       old_active="$active"
28       old_eth0_fcount=$eth0_fcount
29       old_eth1_fcount=$eth1_fcount
30 fi
31
32 echo $active $eth0_fcount $eth1_fcount > $oldstat
33
34 ch=none
35
36 [ $eth0_fcount -ne $old_eth0_fcount ] && ch="$ch eth0_new_failure"
37 [ $eth1_fcount -ne $old_eth1_fcount ] && ch="$ch eth1_new_failure"
38 [ $old_active != $active ] && ch="$ch new_active=$active"
39
40 echo "${ch#none }"
41 echo "$active"
42 echo "bond0:$main_status eth0:$eth0_status eth1:$eth1_status"
```

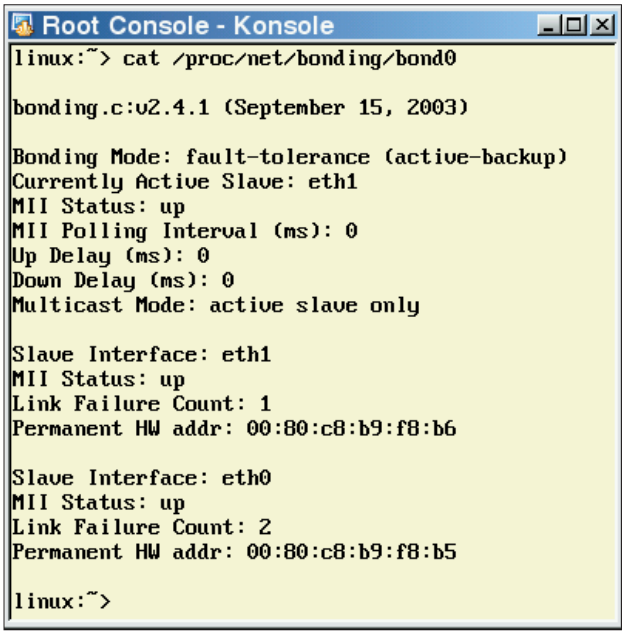


**Abbildung 2:** Über den Status der Bonding-Interfaces informieren die Dateien im Proc-Dateisystem unter »/proc/net/bonding«.

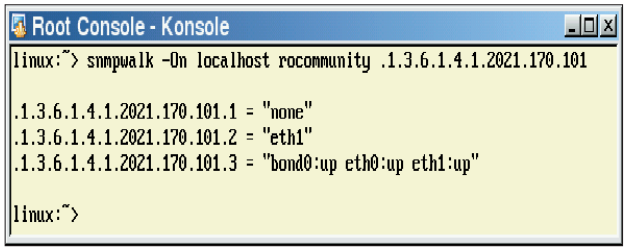


**Abbildung 3:** Der Befehl »snmpwalk« gibt dem Administrator Aufschluss über den aktuellen Zustand des Bonding-Verbunds.

werk-Interface und -Kabel ansprechen. Die Trennung der Netze erledigt der Switch virtuell durch eine entsprechende Erweiterung des Ethernet-Protokolls. Der Switch hängt dazu jedem Ethernet-Paket ein VLAN-Tag an, das die Nummer des virtuellen Netzes enthält. Der Switch muss lediglich wissen, an welchen Ports welche VLAN-Tags erlaubt sind. Damit lässt sich am Switch ein virtuelles Netz definieren und zudem festlegen, welcher Rechner in welchen Netzen hängt.

Am Switch selbst braucht der Administrator die VLAN-Tags nie zu ändern. Wenn zwei Rechner aus unterschiedlichen VLANs miteinander kommunizieren sollen, müssen alle Pakete über einen dritten Host gehen, der in beiden Netzen präsent ist – als wären die VLANs unabhängige Netze.

## VLAN-Tagging mit Vconfig aktivieren

Die VLANs lassen sich mit dem Tool Vconfig [5] konfigurieren. Im folgenden Beispiel bekommt das Interface »bond0« ein getaggtes VLAN mit der VLAN-ID 42 zugewiesen:

```
vconfig add bond0 42
```

Das neue virtuelle Interface ist »bond0.42«. Die Konfiguration unterscheidet sich nicht von der eines ganz normalen Netzwerk-Interface. Zu beachten ist nur, dass das VLAN-Interface beim Shutdown mit »vconfig rem *Interface*« immer sauber aus dem System zu entfernen ist. Die fertige Konfiguration für das Interface »bond0.42« nimmt (auf Debian) wieder die Datei »/etc/network/interfaces« auf (siehe Listing 4).

## Fehler bei der VLAN-Konfiguration vermeiden

Probleme kann das VLAN-Tag im Zusammenhang mit der MTU (Maximum Transfer Unit) verursachen. Grund: Der Ethernet-Header vergrößert sich durch das Tag um 4 Byte. In manchen Netzwerkkarten-Treibern ist jedoch die maximale Gesamtgröße eines Ethernet-Pakets mit einer MTU von 1500 fix einprogrammiert. Daher funktionieren sie mit VLANs und einer MTU von 1500 nicht. Das Hindernis lässt sich umgehen, indem man die MTU für das VLAN-Interface um mindestens 4 Byte verringert oder die entsprechende Konstante in den Treiberquellen um 4 Byte erhöht.

Wer VLANs auf Bonding-Devices einrichtet, sollte noch beachten, dass die IP-Adressen für die ARP-Pings ohne VLAN-Taggs erreichbar sein müssen. Darüber hinaus besteht kein nennenswerter Unterschied im Umgang mit VLANs auf einzelnen Interfaces und VLANs auf Bonding-Interfaces.

In einem Projekt des Autors machen sowohl der Linux-Bonding-Treiber als auch der Linux-VLAN-Treiber eine gute Figur. Abgesehen von den Problemen, die durch eine hart kodierte MTU von 1500 mit VLANs auf manchen Netzwerkkarten-Treibern entstanden sind, traten keine weiteren Schwierigkeiten auf und das System hat den ersten Praxistest, bei dem sich einer der Switches für eine halbe Minute tot gestellt hatte, mit Bravour bestanden. *(jre)* ■

### Der Autor

Clifford Wolf ist Mitarbeiter der Firma Linbit Information Technologies GmbH [http://www.linbit.com] und betreut dort unter anderem das in dem Artikel beschriebene hochverfügbare Netzwerk. Zudem fungiert er als Maintainer von

Rock Linux [http://www.rocklinux.org] und einigen anderen Open-Source-Projekten. Die Adresse seiner privaten Homepage lautet: [http://www.clifford.at]

### Infos

[1] Spanning-Tree: [http://www.javvin.com/protocolSTP.html]
[2] Bonding: [http://kernel.kernelnotes.de]
[3] Dirk von Suchodolez, Achim Leitner, „Ferndiagnose“: Linux Magazin 09/02, S. 34, und [http://www.linux-magazin.de/Artikel/ausgabe/2002/09/snmp/snmp.html]
[4] Net-SNMP: [http://net-snmp.sourceforge.net]
[5] Vconfig: [http://www.candelatech.com/~greear/vlan.html]

**Listing 4: VLAN-Tag mit »vconfig« setzen**

```
01 iface bond0.42 inet static
02 pre-up vconfig add bond0 42
03 pre-up ip l set bond0.42 mtu 1400
04 post-down vconfig rem bond0.42
05 address 192.168.200.65
06 netmask 255.255.255.0
07 broadcast 192.168.200.255
```