

# Hochverfügbare Systeme unter Linux

[Jana Jaeger und Lars Marowsky-Bree](#)

## Inhaltsverzeichnis

- [Warum werden hochverfügbare Systeme eingesetzt?](#)
  - [Warum HA-Lösungen unter Linux einsetzen?](#)
- [Was bedeutet "hochverfügbar"?](#)
- [Wie erreicht man Hochverfügbarkeit?](#)
  - [High Availability Clustering](#)
  - [Load Balancing - mit dem Linux Virtual Server](#)
  - [DRBD](#)
- [Zum Schluß](#)

Dieser Artikel soll einen ersten Überblick über den Begriff hochverfügbarer Systeme geben. Mittlerweile sind Linux-Lösungen in diesen Markt eingedrungen, weshalb die entsprechenden Produkte und Strategien kurz angerissen werden.

## Warum werden hochverfügbare Systeme eingesetzt?

Jeder, der einen rund um die Uhr erreichbaren Server betreiben muß, sollte den Einsatz von HA-Lösungen erwägen. Bereits ein unvorhergesehener Ausfall für von wenigen Minuten kann den Betreiber eines Dienstes in große finanzielle Schwierigkeiten bringen. Krassestes Beispiel hierfür ist der Bankrott von 145 der 450 im World Trade Center ansässigen Unternehmen innerhalb eines Jahres nach dem Bombenattentat von 1993. Sie mußten aufgeben, weil sie keine redundante IT-Struktur hatten, die es ihnen ermöglicht hätte, in kürzester Zeit den Betrieb wieder aufzunehmen. Jede Minute, jede Stunde und jeder Tag des Ausfalls ließ die finanziellen Lasten ins Unermeßliche steigen. Hinzu kam noch ein nicht näher zu beziffernder Imageverlust. Ein Unternehmen im Zeitalter des *e*-Business kann es sich nicht leisten, für einen spürbaren Zeitraum vom Netz zu sein. Hierbei ist die Unternehmensgröße unerheblich, auch kleine Unternehmen werden empfindlich getroffen, wenn ihr E-Mail-System, die Internetanbindung oder der Fileserver ausfallen. In solchen Fällen reicht oft schon der Einsatz einer kleinen, handlichen und vor allem kostengünstigen Linux-Lösung, die den Ausfall des Systems abpuffern kann.

Jeder Verantwortliche muß sich nun überlegen, ob die Vorteile eines HA-Systems nicht bei weitem gegenüber seinen Nachteilen überwiegen. Bei einem Ausfall kommt es nicht mehr nur auf einen gut ausgebildeten Systemadministrator an, der im rechten Moment seinen Notfallplan in die Tat umsetzt. An seiner Stelle übernimmt die HA-Software mittels einer Anzahl ausgeklügelter Skripte, Tools und Protokolle das Regiment. Der Administrator muß im günstigsten Fall nur noch die ausgefallenen Hardwarekomponenten ersetzen. Auf den ersten Blick mag der Kostenaufwand für die Einrichtung eines HA-Systems mit entsprechender Hard- und Softwareausstattung groß erscheinen, aber die Verluste nach einem Ausfall übertreffen sie bei weitem.

## Warum HA-Lösungen unter Linux einsetzen?

Nicht zu unterschätzender Vorteil aller Linux-HA Lösungen ist ihre Transparenz durch die Offenlegung des Quellcodes. Softwarefehler bei Closed Source Lösungen zu suchen (und zu beheben) ist ein

aussichtsloses Vorhaben. Open Source Lösungen mit ihren offengelegten Quellen bieten hingegen die Möglichkeit, den Fehler zu diagnostizieren und selbst eine Problemlösung auszuarbeiten. Diese Flexibilität ist ein weiterer Pluspunkt, wenn es um die Realisierung spezieller Kundenwünsche geht, da am freien Quellcode selbstverständlich Anpassungen vornehmbar sind.

## Was bedeutet "hochverfügbar"?

Die Größe, von der in diesem Zusammenhang am meisten die Rede ist, ist die der prozentualen Verfügbarkeit eines Dienstes (Wartungsfenster für die betroffenen Server sind hierbei ausgenommen). Die folgende Tabelle vermittelt ein Gefühl für die tatsächlichen Ausfallzeiten, die hinter den prozentualen Verfügbarkeitsangaben stecken.

Prozentuale Verfügbarkeit	Größenordnung	tatsächlicher Ausfall
99%	2 Neunen	3,6 Tage
99,9%	3 Neunen	8,76 Stunden
99,99%	4 Neunen	52 Minuten
99,999%	5 Neunen	5 Minuten
99,9999%	6 Neunen	30 Sekunden
99,99999%	7 Neunen	3 Sekunden

Wie sonst im richtigen Leben ist auch hier niemals eine hundertprozentige Sicherheit gewährleistet, aber mit dem entsprechenden Aufwand an Hard- und Software lassen sich die Ausfallzeiten durchaus annehmbar klein halten.

## Wie erreicht man Hochverfügbarkeit?

Zum einen müssen SPOFs (Single Points of Failure) korrekt indentifiziert und anschließend eliminiert werden. SPOFs sind diejenigen Komponenten, deren Ausfall den Komplettausfall des gesamten Dienstes bedeuten würde. Zum anderen muß die Dienstverfügbarkeit bei Ausfall eines einzelnen Systems sichergestellt werden. Ob der konkrete Rechner in diesem Fall erreichbar ist oder nicht, spielt keine Rolle. Wichtig ist in diesem Fall, daß sich im Cluster ein anderer Rechner befindet, der nahtlos dort weiterarbeiten kann, wo sein "Kollege" aufgeben mußte.

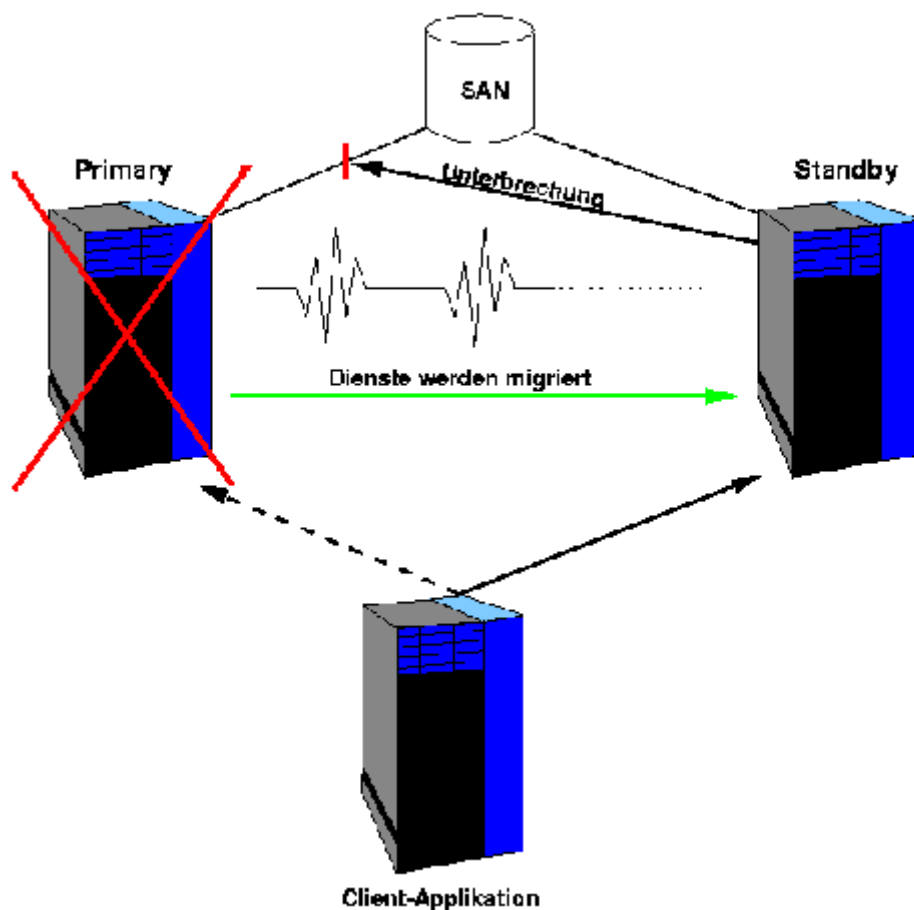
Das Zauberwort im Falle der SPOFs heißt **Redundanz**, einfach gesagt: Jede Komponente, ob Netzteil, Festplatte oder Netzwerkkarte sollte mit einem "Stellvertreter" abgesichert sein, der die Funktion der ausgefallenen Komponente wenn nötig übernimmt. Damit ist der Redundanzen aber noch nicht genug, auch der Server als Ganzes sollte nach hinten abgesichert sein. Konsequenterweise im SPOF-Schema gedacht, stellt der Raum, das Gebäude oder sogar die Gegend, in der der oder die Server stehen, wieder einen Single Point of Failure dar. Um Ausfälle des gesamten Dienstes durch Gebäudebrand etc. auszuschließen, sollte der Backupserver räumlich getrennt vom Hauptserver betrieben werden.

Hat die Dienstverfügbarkeit des Clusters höchste Priorität, ist der Einsatz eines **Load Balancers** zu überlegen. Ein Load Balancer nimmt eine Anfrage von einem Client (z.B. eine http-Verbindung, um eine Website herunterzuladen) an und verteilt diese dann an einen verfügbaren Server. Dies bietet sich

insbesondere bei WWW- und Mail-Diensten, Proxy-Servern, Firewalls und ähnlichem an. Besonders gut geeignet sind Dienste, wo die einzelnen Verbindungen voneinander unabhängig sind, da die Server im Hintergrund ja auch mehr oder weniger eigenständig arbeiten. Wenn ein Server abstürzt, werden die aktiven Verbindungen zu diesem Server unterbrochen. Da aber auch ab sofort keine neuen Anfragen mehr an diesen Server geleitet werden, merkt der Client z.B. beim WWW nichts außer einer kleinen Verzögerung. Wenn er auf Reload drückt, erhält er die Seite korrekt angezeigt (die Anfrage wird von einem neuen Server beantwortet). Auch beim Failover tritt dieser Effekt auf: Die Verbindung vom Client zum Server muß in den meisten Fällen neu hergestellt werden. Damit kann aber die meiste Software umgehen - sonst würde sie auch einen normalen Neustart des Servers nicht abfangen können. Wenn es wirklich "unterbrechungsfrei" (man spricht dann nicht mehr von "highly available", sondern von "fault tolerant") weitergehen soll, benötigt dies auch Anpassungen auf Client-Seite. Das ganze System muß also darauf ausgerichtet sein.

## High Availability Clustering

Die Vorgänge innerhalb eines 2-Node-Clusters beim Ausfall eines Knotens und die verschiedenen Arten des Standby-Systems, das bei Bedarf übernimmt, werden in Folgenden skizziert.



Die beiden Server (Primary und Backup) stehen beide mit dem SAN (Storage Area Network) in Verbindung. Je nach Betriebsart greift nur der jeweils aktive Knoten hierauf auch zu. Untereinander kommunizieren beide in der Form, daß sie sich regelmäßig "Lebenszeichen"(Heartbeat) senden. Sterben die Lebenszeichen des Hauptsystems ab, wird das Standby-System aktiviert, übernimmt die Dienste des ausgefallenen Partners und unterbricht dessen Verbindung zum SAN. Richtig interessant wird die Situation in einem 2-Node-Cluster, wenn die interne Kommunikation ausfällt. Beide Knoten glauben nun, der jeweils andere wäre nicht mehr aktiv. Sie versuchen dann beide gleichzeitig, für den jeweils anderen einzuspringen und sich vom SAN abzuschneiden. Eine solche Pattsituation wird vermieden, indem ein bestimmtes Failover(Übernahme)-Verhalten beim Einrichten des Systems vordefiniert wird. Wie die genannten Aspekte realisiert werden, hängt stark von der Art der gewählten Standby-Strategie

und der gewählten Softwarelösung ab.

Zum Abschluß ein Überblick über die drei grundlegendsten Arten von Standby-Strategien:

### **Cold-Standby**

Die Ersatzhardware steht "kalt" bereit. Die Übernahme muß manuell erfolgen, so daß der Ausfall dementsprechend auch deutlich spürbar ist.

### **Warm-Standby**

Das Backupsystem läuft im Hintergrund mit, so daß die Übernahme automatisch erfolgen kann. In regelmäßigen Zeitabständen werden die Daten auf beiden Systemen synchronisiert. Der Ausfall ist nur kurz für den Benutzer spürbar, allerdings kann die aktuelle Transaktion möglicherweise verloren gehen, da die Daten vor dem Ausfall nicht mehr synchronisiert werden konnten.

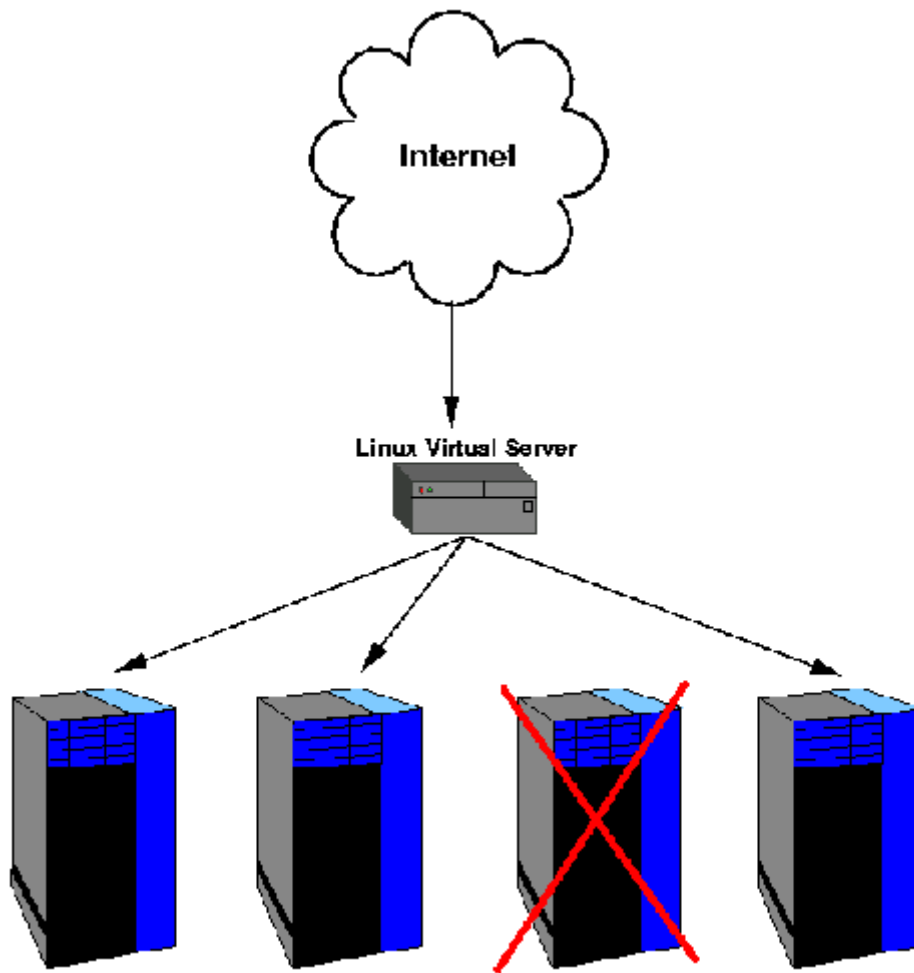
### **Hot-Standby**

Beide Systeme laufen ständig parallel - Daten auf beiden Systemen sind hundertprozentig synchron. Der Benutzer spürt nichts von eventuellen Ausfällen. Meist ist diese Stufe nicht ohne eine entsprechende Modifikation des Clients zu erreichen. Um beide Systeme 100% synchron zu betreiben, müssen auch die Verbindungen zum Client 100% gespiegelt werden. Dafür braucht man "normalerweise" Clients, die Verbindungen zu zwei oder mehr Servern gleichzeitig halten und mit allen reden. Das kann beispielsweise ein normaler Webbrowser nicht.

Frei als Open Source Software für Linux-HA-Lösungen sind *heartbeat* und *Linux FailSafe* erhältlich. *heartbeat* wurde ursprünglich von Alan Robertson als eine Art Designstudie für das KeepAlive Protokoll geschaffen. Mittlerweile hat es sich verselbständigt und diente als "Initialzündung" für das [Linux-HA-Projekt](#). Es stellt "Warm Failover" für 2-Node-Cluster zur Verfügung. "Lebenszeichen" können über Ethernet und serielle Verbindungen ausgetauscht werden. *heartbeat* eignet sich, da es auf kleine Cluster zugeschnitten ist, besonders gut für den Low Cost Markt. *Linux FailSafe*, ursprünglich SGI's hauseigene Software für HA-Cluster, wurde von SuSE und SGI auf Linux portiert. Seit August 2000 ist es als Open Source freigegeben. Es ist der zukünftige offene Standard für HA unter Linux. Seine Leistungen: Es bietet flexibles "Warm Failover" für Cluster mit bis zu acht Knoten und ein umfassendes Administrations-GUI unter Java. *Linux FailSafe* bietet zudem einen Rahmen für "Hot Standby" HA-Lösungen und kann aufgrund seiner modularen Architektur schrittweise nach Bedarf erweitert werden. Der Leistungsumfang von *Linux FailSafe* ist mittlerweile dem der kommerziellen HA-Lösungen mindestens ebenbürtig, wenn nicht sogar überlegen.

## **Load Balancing - mit dem Linux Virtual Server**

Webfarmen oder Mailcluster lassen sich unter Linux mit [Linux Virtual Server](#) realisieren. Die einzelnen "real existierenden" Server können untereinander entweder über LAN oder geographisch getrennt über WAN miteinander verbunden sein. Ihnen vorgeschaltet ist ein Load Balancer, der die Last möglichst gleichmäßig über die ihm untergeordneten realen Server verteilt. Der Parallelbetrieb der Server erscheint nach außerhalb als die Leistung eines einzelnen virtuellen Servers unter einer einzigen IP-Adresse. Bei dem Ausfall einzelner Knoten wird das System entsprechend rekonfiguriert.




## DRBD

DRBD ist die Abkürzung für "distributed replicated block device". Es steht RAID1 (also Plattenspiegelung) auf Software-Basis zur Verfügung, jedoch ist im Gegensatz zu einem normalen RAID Verbund die zweite Platte nicht lokal angeschlossen, sondern die Spiegelung erfolgt über ein Netzwerk (Fast Ethernet, Gigabit Ethernet) auf ein zweites System. Im Falle eines Falles ist die Kopie des Blockdevices aktuell und sofort lokal auf dem zweiten Server verfügbar. Der riesengroße Vorteil dieses Ansatzes: Das zweite System kann sehr weit entfernt sein - somit brennen zum Beispiel nicht beide Platten gleichzeitig ab. Im Gegensatz zu den üblichen Shared Storage Lösungen, die auf Fibre-Channel und SCSI basieren, ist dieses System sehr günstig, da z.B. auch preiswerte IDE Platten zum Einsatz kommen können. Auch eliminiert dies wirklich auch den letzten SPOF, nämlich den Shared Storage selbst: Die Daten sind räumlich voneinander getrennt, oder zumindest auf zwei Platten in zwei Rechnern verteilt. Ideal ist dieses Konzept für anspruchsvolle geographisch verteilte Cluster oder auch Low-Cost Cluster, da man mit preisgünstiger Hardware (zwei normale Rechner, je eine zusätzliche Platte und Netzwerk) ein HA-System realisieren kann. Details zu DRBD finden sich unter <http://www.complang.tuwien.ac.at/reisner/drbd/>.

## Zum Schluß

Pflichtlektüre für alle diejenigen, die mehr über HA unter Linux erfahren wollen, ist die [Linux HA Project Site](#). Hier finden sich viele nützliche Links, Konfigurationsbeispiele und Downloads. Trotzdem sollte die Konzeption, Einrichtung und Wartung einer (großangelegten) HA-Lösung von Fachleuten übernommen werden. Die [SuSE Linux Solutions AG](#), bietet als Teil ihres umfassenden Serviceangebots

auch die Einrichtung von HA-Lösungen an.

*Lars Marowsky-Brée ist bei der SUSE Linux Solutions AG als Consultant und Entwickler mit dem Schwerpunkt HA Systeme tätig.* 

**Linux auf dem Server 12.12.2000**