Der Weg zur Hochverfügbarkeit

Verfügbarkeit ist ebenso nötig wie teuer. Entsprechend wichtig ist eine fundierte Strategie. Wie ein paar systematische Schritte effektiv zu höherer Verfügbarkeit führen, zeigt dieser Beitrag. Jens-Christoph Brendel

Wir stecken in einem Dilemma: Sowohl die Komplexität der IT wie auch unsere Abhängigkeit von dieser Rechentechnik nehmen beständig zu. Doch beide Tendenzen stehen unversöhnlich auf Kriegsfuß: Komplexität ist der Feind der Verfügbarkeit. Eine reibungslos funktionierende IT aber ist lange schon kein bloßer Wettbewerbsvorteil mehr, sondern eine Existenzfrage. Das ist keine leere Floskel, wie zahlreiche Untersuchungen von Marktforschern und Versicherungen belegen, denen zufolge 40 bis 70 Prozent der Firmen, die von schweren und länger anhaltenden Ausfällen ihrer IT betroffen waren, das darauf folgende Jahr nicht überlebten.

High Five

Also Verfügbarkeit um jeden Preis? Das wäre ein unbezahlbarer Ausweg. Denn besonders hohe Ausfallsicherheit ist nicht billig: Jede redundante Hardwarekomponente ist doppelt so teuer wie die einfache Ausführung, Failover-Software oft nicht nur kostspielig, sondern meist auch komplex. Das erfordert weitere Investitionen in Know-how und Training.

Zum Glück ist es aber auch gar nicht nötig, dass jede Allerwelts-Applikation die marketingträchtigen Five Nines (99,999 Prozent Verfügbarkeit) anstrebt – denn es geht nicht um unbedingte, sondern um angemessene Verfügbarkeit.

Was ist angemessen? Um die Anforderungen dafür zu sortieren, wurden verschiedene Klassifikationen entwickelt. **Tabelle 1** listet gängige Verfügbarkeitsklassen, die sich an der Anzahl der Neunen orientieren. Einen anderen und zugleich anschaulicheren Ansatz entwarf die Harvard Research Group (HRG), die ein Schema entwickelt hat, das sich nicht an der Downtime-Quantität, sondern an der Qualität der Auswirkung eines Ausfalls orientiert (**Tabelle 2**).

Erste Orientierung

Anstelle eines blinden Verfügbarkeits-Aktionismus ist also eine durchdachte Strategie gefragt. Dazu kann man sich für einen ersten Überblick daran orientieren, dass sich alle Rechner grob in folgende Klassen mit verschiedenen Ansprüchen an die Ausfallsicherheit einordnen lassen:

- **Allzwecksysteme** (General Purpose Computing): In dieser Gruppe sind gelegentliche Ausfälle noch tolerabel, wenn sie sich schnell und automatisch beheben lassen. Die Spannweite der Anwendungen ist bei diesen Rechnern sehr groß; sie reicht von Fileservern oder Computern für wissenschaftliche oder technische Berechnungen bis hin zu Datenbanken oder zu ERP-Systemen.
- **Hochverfügbare Systeme** (Highly Available Systems): In dieser Klasse ist die Verfügbarkeit das absolut wichtigste Kriterium. Lokale Störungen, die bereits nach kurzer Zeit wieder repariert sind, lassen sich zwar auch hier noch hinnehmen, aber ein Totalausfall des Systems ist keinesfalls mehr zu akzeptieren. Derartige Rechner stehen beispielsweise in Banken oder auch in Telekommunikationsunternehmen. ▶

Abbildung 1: Redundante Standardkomponenten – im Falle eines Blade Center sind das ganze Rechner – kompensieren einen Ausfall. Dasselbe Prinzip wirkt bereits beim zweiten Netzteil oder Ethernet-Interface.

Tabelle 1: Verfügbarkeitswerte

Jährliche Verfügbarkeit	Downtime pro Jahr	Verfügbarkeitsklasse	
90 %	36,50 Tage		
95 %	18,25 Tage		
98 %	7,30 Tage		
99 %	3,65 Tage	2	stabil
99,5 %	1,83 Tage		
99,8 %	17,52 Stunden		
99,9 %	8,76 Stunden	3	verfügbar
99,95 %	4,38 Stunden		
99,99 %	52,60 Minuten	4	hochverfügbar
99,999 %	5,26 Minuten	5	fehlerunempfindlich
99,9999 %	31,50 Sekunden	6	fehlertolerant
99.99999 %	3,00 Sekunden	7	fehlerresistent

- **Kritische Systeme** (Critical Computational Systems): Hier kann ein Ausfall Menschenleben gefährden oder hohe ökonomische Verluste nach sich ziehen, dementsprechend gilt es, diese Situation unter allen Umständen zu vermeiden. Beispiele finden sich etwa in der Medizin, der Flugsicherung, bei manchen Produktionssteuerungen oder im militärischen Bereich.
- **Langlebige Systeme** (Long-life Systems): In dieser Gruppe steht die Zuverlässigkeit im Vordergrund. Es handelt sich um Systeme, bei denen eine ungeplante Wartung nicht oder nur mit extremen Kosten möglich ist, weil sie autonom und weit entfernt operieren. Dazu zählen etwa Satelliten.

Den Computer für die Urlaubsplanung der Abteilung wie den Zentralrechner eines Kernkraftwerks zu sichern, wäre offensichtlicher Unsinn. Die meisten Rechner fallen in die Klasse der Allzwecksysteme, für die ein mittlerer Aufwand für die Vorbeugung vor lang andauernden oder häufigen Ausfällen und ein automatisches Recovery gerechtfertigt ist. Das allein sagt aber noch nicht allzu viel darüber, wie dieses Ziel zu erreichen ist. Dafür ist es nötig, noch weiter in die Details vorzudringen.

Risiken identifizieren

Der erste Schritt bei der Entwicklung einer effektiven Verfügbarkeitsstrategie besteht darin, zusammenzutragen, was ausfallen kann und also potenziell schützenswert ist. Hier mag man gleich das eine oder andere unkritische System ausklammern können, für das keine Vorsorge nötig ist.

Zu den Applikationen, die Unternehmen am häufigsten als unverzichtbar einstufen, zählten nach einer Umfrage der Aberdeen Group in diesem Jahr (1) in erster Linie E-Mail-Server mit 83 Prozent der Nennungen (siehe auch zahlreiche Artikel zu verschiedenen Mail-Clustern in dieser Ausgabe) und Datenbanken mit 63 Prozent (vergleiche die Beiträge zu etlichen Datenbank-Clustern weiter hinten). ERP-Daten nannte ein Drittel der Befragten besonders wichtig, gefolgt von den Daten gerade in Arbeit befindlicher Projekte (30 Prozent).

Sich allein auf solche Applikationen zu konzentrieren, reicht aber nicht aus. Es gilt, den Gesichtskreis zu erweitern, denn jeder Server hängt von der Infrastruktur ab, in die er eingebettet ist, und der beste Cluster ist vollkommen nutzlos, wenn Strom oder Kühlung ersatzlos ausfallen, im Brandfall keine Löschanlage existiert, Kon-

Tabelle 2: HRG-Verfügbarkeitsklassen

Klasse	Auswirkungen	Definition
ABC-0	Conventional	Für Geschäftsfunktionen, die unterbrochen werden dürfen und bei denen die Datenintegrität nicht so wichtig ist. Fallen Rechner dieser Klasse unvermittelt aus, unterbrechen sie die Arbeit der Anwender abrupt, und es kann zu Datenverlust kommen.
ABC-1	Highly Reliable	Für Geschäftsfunktionen, die unterbrochen werden dürfen, solange die Datenintegrität gewahrt bleibt. Auch hier ereignen sich plötzliche Ausfälle, es kommt aber nicht zu Datenverlusten.
ABC-2	High Availability	Für Geschäftsfunktionen, die nur eine kurzzeitige oder geplante Unterbrechung erlauben. Die Arbeit der Anwender stoppt, sie können sie aber unmittelbar danach wieder fortsetzen. Daten gehen nicht verloren, unter Umständen sind jedoch Transaktionen zu wiederholen, und die Performance kann vermindert sein.
ABC-3	Fault Resilient	Für Geschäftsfunktionen, die einen unterbrechungsfreien Betrieb erfordern. Die Anwender bleiben durchgängig online, müssen unter Umständen allerdings Transaktionen wiederholen und verminderte Performance in Kauf nehmen.
AEC-4	Fault Tolerant	Für Geschäftsfunktionen, die unterbrechungsfrei ablaufen müssen und bei denen Fehler für die Anwender transparent bleiben. In einem 24x7-Betrieb gibt es hier keine Unterbrechung, außerdem gehen keine Transaktionen verloren, und es ist keine Performanceeinbuße spürbar.
AEC-5	Disaster Tolerant	Für Geschäftsfunktionen, die unterbrechungsfrei ablaufen müssen und bei denen Fehler transparent bleiben. Es gehen weder Transaktionen verloren, noch kommt es zu einer spürbaren Verschlechterung der Performance. Das gilt für diese Klasse selbst im Fall eines Desasters, wie Feuer, Erdbeben, Sturm, Stromausfall oder Vandalismus.

denswasser in die Serverschränke tropft, das Netzwerk zusammenbricht oder auch nur das DNS versagt.

Risiken priorisieren

Sind die Risiken zusammengestellt, gilt es, sich zu überlegen, wie wahrscheinlich jedes einzelne ist und welcher Schaden entsteht, wenn es eintritt. Das Produkt aus Eintrittswahrscheinlichkeit und Folgekosten ergibt einen Vergleichswert, der es ermöglicht, seltene, aber teure mit häufigen, aber billigen Risiken zu vergleichen.

Diese Risikorangliste gibt die Marschrichtung vor. Sie bewahrt davor, viel Zeit und Geld auf unbedeutende oder extrem seltene Gefahren zu verschwenden und dabei das Wichtigste zu übersehen. Zudem erzwingen meist begrenzte Mittel eine abgestufte Vorgehensweise. ▶

Grundlegende Begriffe

Die **Zuverlässigkeit** (Reliability) ist eine Eigenschaft, die angibt, wie wahrscheinlich ein System während einer bestimmten Zeitspanne den ihm zugedachten Zweck erfüllt. Zuverlässigkeit ist nicht direkt messbar (im Unterschied etwa zu Größe oder Gewicht), man ermittelt sie entweder empirisch durch Beobachtung der Ausfallhäufigkeit oder analytisch aus der Zuverlässigkeit der einzelnen Komponenten.

Ein in der Technik häufig gebrauchtes Maß für die Zuverlässigkeit ist die Mean Time Between Failures (MTBF), die besagt, wie viel Zeit im Durchschnitt zwischen zwei aufeinanderfolgenden Fehlern vergeht.

Ein häufiger Irrtum ist, zu erwarten, dass eine MTBF von 100 000 Stunden bedeutet, dass das Gerät 100 000 Stunden oder 4 166 Tage oder rund 11,5 Jahre durchläuft, ehe es zum ersten Fehler kommt. Das ist leider nicht der Fall. Die MTBF (ein statistischer Wert) kann wesentlich größer als die Lebensdauer sein, mit der sie nichts zu tun hat. So könnte beispielsweise eine Batterie einen Einsatzzeitraum (Useful Life) von vielleicht fünf Stunden, aber eine MTBF von 100 000 Stunden haben. Das bedeutete, dass von 100 000 Batterien eine pro Stunde Betriebsdauer einen Fehler aufweist. Umgekehrt kann die MTBF kürzer als die Lebenszeit sein, in der sich dann im Mittel mehrere Fehler ereignen.

Der Kehrwert der MTBF (1/MTBF) ist die Fehlerrate. Die Fehlerrate ist in der Regel nur während des eigentlichen Einsatzzeitraums (Useful Life) annähernd konstant. Insgesamt folgt sie dagegen einer Badewannen-Kurve (*Abbildung 2*). Den hohen Werten in der Anfangsphase versucht man oft durch einen Burn-In-Prozess zu begegnen, der Frühausfälle vor Beginn des produktiven Einsatzes aussondert. Den vermehrten Problemen am Lebensende kommt ein rechtzeitiger Austausch zuvor.

Die **Verfügbarkeit** (Availability) ist eine weitere Systemeigenschaft, die die Wahrscheinlichkeit dafür angibt, dass ein System während einer bestimmten Zeitspanne einen bestimmten Service unterbrechungsfrei anbieten kann. Das klingt so ähnlich wie die Definition der Zuverlässigkeit, ist aber nicht dasselbe. In die Verfügbarkeit gehen nämlich zusätzlich die Auswirkungen geplanter und ungeplanter Auszeiten (Downtime) ein. Eine Messgröße für diese Downtime ist die Mean Time To Repair (MTTR), die angibt, wie lange die Wiederherstellung nach einem Ausfall durchschnittlich dauert. Daraus ergibt sich: Availability = (Gesamtzeit - Downtime) / Gesamtzeit oder V = MTBF / (MTBF + MTTR). Wie man sieht, ist die Verfügbarkeit somit eine Funktion von Zuverlässigkeit und Wartbarkeit (Serviceability).

Ein System mit geringer Zuverlässigkeit kann dennoch hoch verfügbar sein, wenn die Zeit zur Wiederinbetriebnahme nach einem Fehler sehr kurz ist. Fällt ein Dienst etwa jeden Monat aus, ist aber nach einer Minute wieder funktionstüchtig, erreicht er damit eine jährliche Verfügbarkeit von 364 Tagen, 23 Stunden und 48 Minuten oder mehr als 99,99 Prozent der Gesamtzeit.

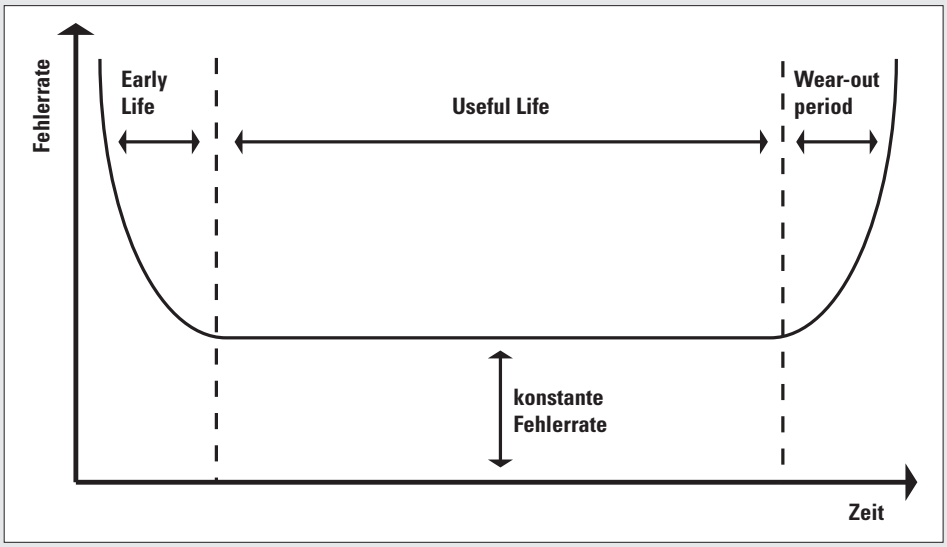

Abbildung 2: Die Fehlerrate steigt in frühen und späten Lebensphasen eines Systems stark an, ist über den eigentlichen Gebrauchszeitraum hinweg aber meist konstant.

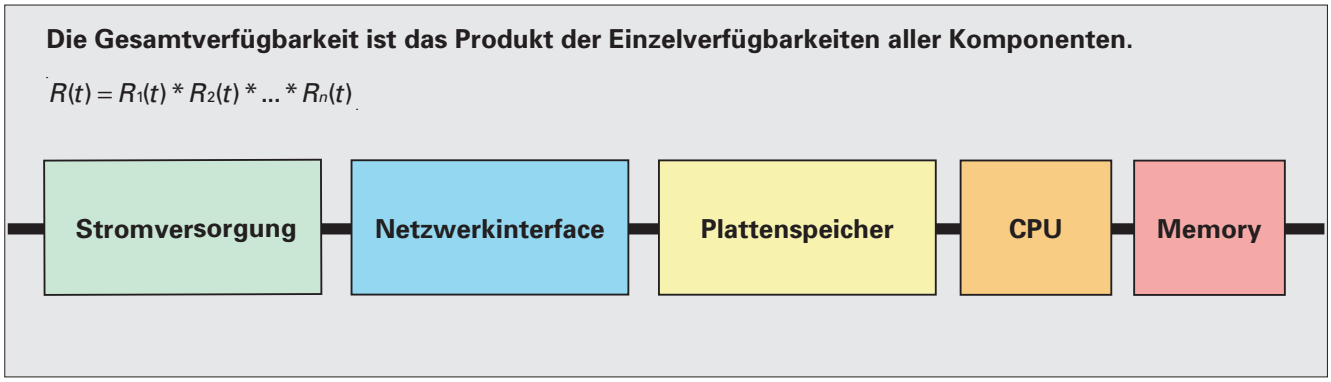

Abbildung 3: In einer Reihenschaltung von Komponenten multiplizieren sich die Einzelverfügbarkeiten; mit steigender Komponentenzahl erhöht sich das Fehlerrisiko.

Bei der Ermittlung der Eintrittswahrscheinlichkeit kann man sich unter anderem an der MTBF orientieren (siehe **Kasten „Grundlegende Begriffe“**), die Kosten der Downtime sind anwendungs- und branchensspezifisch, können aber beträchtliche Höhen erreichen.

Der amerikanische Marktforscher Gartner etwa schätzt die durchschnittlichen Kosten für eine Stunde Netzwerkausfall auf 42 000 Dollar (2) allein an Personalkosten für die zur Untätigkeit verdammten Mitarbeiter; noch gar nicht eingerechnet sind die Kosten für die Schadensbehebung selbst, verlorene Transaktionen, unzufriedene Kunden, mögliche Regressforderungen und so weiter. In bestimmten Branchen, etwa bei Banken oder Börsen, fallen sie jedoch noch um ein Vielfaches höher aus.

Aktionen festlegen

Für jedes Risiko der nun sortierten Liste gibt es im Wesentlichen vier grundsätzliche Handlungsmöglichkeiten:

- **Risikovermeidung** (Risk Avoidance) Das scheint auf den ersten Blick die Pauschalantwort auf alle Risiken zu sein: Man tut etwas, damit sie gar nicht erst entstehen. Die pauschale Konsequenz im vorliegenden Fall wäre allerdings: Man arbeitet nicht mit Computern, denn auf andere Weise sind IT-Risiken wohl kaum vollständig auszuschließen. Im kleinen Rahmen jedoch ist das Vermeiden sehr wohl ein probates Mittel. Eine RAID-Gruppe etwa hilft, das Risiko eines Datenverlusts nach einem Festplattenausfall zu vermeiden.

Best Practices

Es muss nicht immer der teuerste Cluster sein. Die Verfügbarkeit lässt sich schon durch eine Reihe einfacher Maßnahmen verbessern, die häufig wenig kosten, aber großen Einfluss auf eine hohe Ausfallsicherhiet haben. Dazu zählen:

- **KISS** („Keep it simple and stupid“): Jede unnötige Komplexität verschlechtert die Verfügbarkeit.
- **Single Points of Failure** vermeiden, wo es geht. Mehr Redundanz ist oft einfach zu erreichen. Zu den simplen Maßnahmen zählen etwa ein zweites Power Supply oder Netzwerkinterface.
- **Backups**: Man kann nie für unendliche Redundanz sorgen, es ist immer möglich, dass gleichzeitig mehr Komponenten versagen, als das System verkraftet. Dann rettet nur ein funktionierendes Backup vor dem Totalverlust.
- **Training**: Backup- und Disaster-Recovery-Prozesse wie auch Failover-Szenarien müssen trainiert und regelmäßig getestet werden. Ansonsten gehen sie im Stress des Ernstfalls vermutlich schief.
- **Monitoring**: Ein proaktives Monitoring kann häufig Probleme erkennen, bevor sie zu einem Ausfall führen. Dann lässt sich die Katastrophe noch vermeiden.
- **Dokumentation**: Ohne aktuelle Systemdokumentation, ein penibles Change-Management, redundant gespeicherte Konfigurationsdateien und die akurate Beschriftung der Hardware ist eine Krisensituation ungleich schwerer zu beherrschen. Wer nach einem Crash erst Grundlagenforschung betreiben muss, um die letzte Konfiguration zu rekonstruieren, hat schon verloren.
- **Patches**: Viele Patches vermeiden Fehler, die andere Anwender bereits getroffen haben. Ein aktuelles System ist in der Regel ein stabileres System.
- **Spare Parts**: Austauschteile sollten griffbereit sein. Wo sich das aus Kostengründen nicht rechnet, müssen mit Systemhäusern oder Hardwareherstellern Wartungsverträge geschlossen werden, die garantieren, dass ausgefallene Hardware in kurzer Zeit zu ersetzen ist.

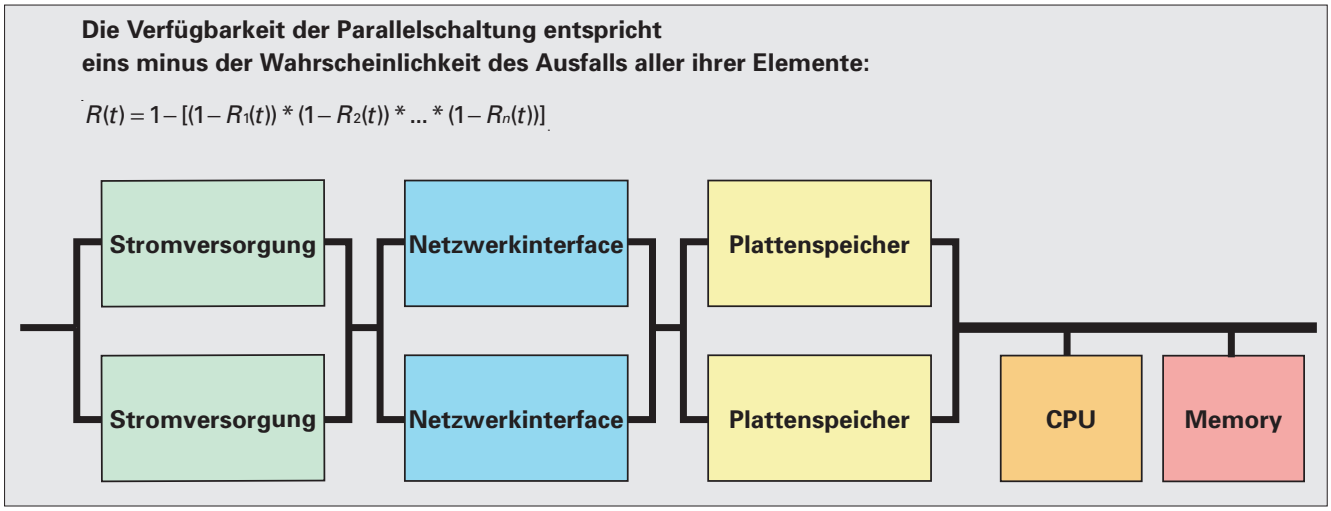

Abbildung 4: In einer Parallelschaltung von Komponenten multiplizieren sich die einzelnen Ausfallwahrscheinlichkeiten, mit steigender Anzahl paralleler Komponenten steigt auch die Verfügbarkeit.

- **Risikominderung** (Risk Reduction): Kann man das Risiko nicht völlig vermeiden, so lässt es sich doch oft mindern. Beispielsweise minimiert ein funktionierendes Backup das Risiko, Daten zu verlieren, weil in der RAID-5-Gruppe eine zweite Platte ausfällt, bevor die erste ersetzt wurde.
  Eine Sprinkleranlage verringert das Brandrisiko. Allerdings kann hier der Schaden durch Löschwasser größer als der Nutzen sein – einen Ausweg böte eine Halon-Löschanlage, aber die ist teuer. Hier, wie in vielen Fällen, wären dann wieder die oben ermittelten Kosten im Schadensfall zum Vergleich heranzuziehen. Je höher der potenzielle Schaden, je eher rechtfertigt er eine kostspielige Gegenwehr.
- **Risikoverlagerung** (Risk Transfer): Dieses Herangehen kennt jeder, der je eine Versicherung abgeschlossen hat. Auch für IT-Risiken ist das möglich. Allerdings reduziert diese Strategie weder Gefahr noch Folgen, sondern nur die Kosten für den Versicherten.
- **Risikoakzeptanz** (Risk Retention): Das bedeutet: Man nimmt das Risiko in Kauf, ohne etwas dagegen zu unternehmen. Das kann in manchen Fällen durchaus sinnvoll sein, beispielsweise dann, wenn die Kosten der anderen Handlungsalternativen über dem Schaden liegen, der schlimmstenfalls eintreten könnte.

Unter diesen Möglichkeiten sind sicher Vermeidung und Minderung das Mittel der Wahl für den Entwickler einer guten HA-Strategie.

Redundanz

Weiß man nun, mit welchem Aufwand man versuchen kann, bestimmte Gefahren abzuwenden, stellt sich die Frage, wie das zu erreichen wäre. In einer ersten Näherung kann man sich dazu einen Rechner als Reihenschaltung von Komponenten vorstellen (**Abbildung 3**). Fällt eine Komponente aus, dann ist – wie bei einem Stromkreis dieser Art – das gesamte System funktionsuntüchtig.

Mathematisch betrachtet ist hier die Gesamtverfügbarkeit das Produkt der Einzelverfügbarkeiten. Da keine Komponente per se vor einem Ausfall geschützt ist, ihre Verfügbarkeit also stets kleiner als eins ist, nimmt die Gesamtverfügbarkeit mit jeder weiteren Komponente ab. Das ist der Beleg für die Eingangshypothese, wonach Komplexität mit Verfügbarkeit auf Kriegsfuß steht.

Legt man nun einige Komponenten doppelt aus, dann entspricht jedes der dadurch entstehenden Komponentenpaare einer Parallelschaltung. Der Ausfall eines Glieds der Parallelschaltung beeinträchtigt die Funktion des Pärchens nicht, weil nur ein Element der Gruppe tatsächlich nötig ist. Jedes weitere parallelgeschaltete Element taugt als potenzieller Ersatz nach einem Fehler (**Abbildung 4**).

In einer Parallelschaltung entspricht die Gesamtausfallzeit N_{gesamt} dem Produkt der einzelnen Ausfallzeiten, das heißt, sie vermindert sich mit jeder weiteren parallelen Komponente. Im Gegenzug erhöht sich dabei die Verfügbarkeit entsprechend. ▶

Abbildung 5: Auch Thin Clients – hier die Sun Ray 1g – können einen Beitrag zu hoher Verfügbarkeit leisten. Sie selbst schützt geringe Komplexität, und der zugehörige Server ist in der Regel redundant ausgelegt.

Damit ist die schärfste Waffe gefunden, die die Verfügbarkeit gegen die Komplexität ins Feld führen kann: Sie heißt Redundanz. Je redundanter ein System ausgelegt ist, je verfügbarer wird es sein. Allerdings: Redundanz erzeugt immer auch Komplexität, die wiederum eine potentielle Fehlerquelle ist. Deshalb ist auch hier abzuwägen, ob im Einzelfall nicht eine etwas weniger redundante, aber robustere Lösung einer stärker redundanen, dafür aber hochkomplexen vorzuziehen ist.

SPOF vermeiden

Im hier gezeigten Beispiel sind noch Solokomponenten ohne paralleles Pendant verblieben. Jede bildet einen so genannten Single Point of Failure (SPOF), den es nach Möglichkeit zu vermeiden gilt, weil dessen Ausfall zu einem Totalausfall des Systems führt. Redundanz vertreibt diese Schwachstellen automatisch.
Allerdings ist das nicht immer möglich. Beim Hauptspeicher im Beispiel etwa kann man nur auf die interne Fehlerkorrektur (Error Correction Code, ECC) setzen und durch proaktives Monitoring der Bitfehlerraten versuchen, sich anbahnende Ausfälle frühzeitig zu erkennen, um das betroffene Modul vorher zu tauschen.
Die Auswirkung jeder weiteren Parallelisierung auf die Zuverlässigkeit ist ausrechenbar. Auch für komplizierte Strukturen oder Komponenten, wie RAID-Gruppen mit Spare Disks, gibt es entsprechende Formeln. Für einfache Fälle bekommt man das mit dem Taschenrechner hin; für komplexe Anwendungen existiert spezielle Software , die unter anderem auch umfangreiche Datenbanken mit MTBF-Werten aller möglicher Komponenten enthält. Ein Beispiel für so ein Programm ist das Relex Reliability Studio (3). So lässt sich bei Bedarf immer ermitteln, wie viel Redundanz sich rechnet und zu wie viel zustzlicher Verfügbarkeit sie führt.

Auf allen Ebenen

In der Praxis reicht das doppelte Vorhandensein einer Komponente alleine meist noch nicht aus, um die Parallelschaltung zu realisieren. Stattdessen muss man den Reservespielern erst auf die Sprünge helfen.
Ein zweites Netzteil schaltet der Rechner in Sekundenbruchteilen noch selbst ein, sollte das erste ausfallen, Netzwerkinterfaces lassen sich durch so genanntes Bonding zu hochverfügbaren Einheiten koppeln , beim Plattenspeicher gelingt das durch RAID-Konfigurationen oder Replikation, der komplette Server wird durch ein Standby-System redundant. Möchte man die Redundanz allerdings auf die Dienste ausdehnen, die der Server anbietetet, dann benötigt man einen Cluster.
Mit den Typen, Grundbegriffen und Problemen der Cluster-Technologie beschäftigt sich der folgende Beitrag. Konkrete Anwendungsfälle dagegen demonstrieren viele weitere Artikel dieser Ausgabe. Dabei kommen weder Lösungen für den universellen Einsatz, noch spezielle Techniken zu kurz. ■■■

Infos

(1) Aberdeen Group, The Importance of High Availability: (http://www.aberdeen.com/summary/report/benchmark/3947_RA_HACDP.asp)
(2) Kosten von Netz-Downtime: (http://findarticles.com/p/articles/mi_qa3649/is_199604/ai_n8749298)
(3) Relex Reliability Studio: (http://www.relex.com/products/index.asp)