Daten spiegeln ohne Grenzen

Franz Diegruber
Libelle AG
Stuttgart

Schlüsselworte
WAN, Oracle, Applikationsspiegelung, Desaster Recovery, Logische Fehler, RTO, RPO, RCO.

Einleitung
Bedrohungen wie Brand, Wasser, Terror; organisatorische Erfordernisse eines „Follow-the-sun"-Betriebes; Anforderungen aus der Revision und gesetzliche Bestimmungen: viele Gründe ein Standby-System räumlich getrennt vom produktiven Echt-System zu betreiben.

Anhand von Best Practices und konkreter Betriebserfahrungen mittelständischer und großer Unternehmen wird dargelegt, welche Ereignisse den Einsatz entfernter Desaster-Recovery-Standorte notwendig machen, welche Auswirkungen dies auf den operativen Betrieb haben kann, und wie diese Anforderungen speziell im Umfeld des Datenbanken- und Applikationsbetriebes umgesetzt werden können.

Der Vortrag zeigt auf, welche Möglichkeiten es gibt, Standby-Systeme im Zusammenhang mit optimierter Auslastung verfügbarer WAN-Bandbreiten zu nutzen.

Für die Spiegelung eines Applikationsservers gilt es auch Filesysteme der Applikation zu überwachen und auf dem Standby-System synchron zu halten.

Überlegungen zur wirtschaftlichen Nutzung von Hardware-Ressourcen spielen dabei eine zentrale Rolle: Können ausgemusterte Altsysteme als Standby System dienen? Können mittels Cross-over-Spiegelung Ressourcen besser genutzt werden? Kann Virtualisierung durch situationsabhängige Ressourcenverteilung zu Kosteneinsparungen führen?

Es wird auch gezeigt, welchen zusätzlichen Nutzen das Standby-System zur Verfügung stellen kann, z.B. für Reporting oder Backup.

Zudem wird beleuchtet, welche Auswirkungen Netzwerkunterbrechungen auf den Betrieb eines Datenbankspiegels haben? Wie der Umgang mit nologging-Transaktionen aussieht und wie unterschiedliche IP-Segmente gehandhabt werden?

Gründe für den Aufbau eines räumlich getrennten Standby-Systems

Denkt man an die Beweggründe ein Standby-System an einem entfernten Standort aufzubauen, kommen häufig zuerst die Desastervorsorge gegen Brand, Wasserschäden oder auch Terror in den Sinn. Daneben schreiben oft auch betriebliche Bestimmungen oder gesetzliche Vorgaben eine Spiegelung an einen räumlich getrennten Standort vor. Zu nennen sind hier beispielsweise Basel II und SOX (Sarbanes-Oxley Act).

Zusätzlicher Nutzen kann aus dem Standby-System für die Zentralisierung operativer RZ-Aufgaben wie z.B. Reporting und Backup gezogen werden. So gibt es den Ansatz von den Zweigstellen in ein zentrales Rechenzentrum zu spiegeln und dort das Reporting oder das Backup durchzuführen. Mit dem Backup vom Standby-System bietet sich die Möglichkeit ein offline-backup zu fahren und dennoch

für die Produktion einen 7x24-Stunden-Betrieb zu gewährleisten. Ein Standby-System kann beispielsweise auch für die Erzeugung von Systemkopien für Entwicklungs- oder Testzwecke ohne Beeinflussung des Produktivbetriebes herangezogen werden.

Auch im temporären Einsatz, z.B. bei Umzugs- und Migrationsprojekten, macht es häufig Sinn mit WAN-Spiegelungen auf Basis von Standby-Systemen zu arbeiten, da hierbei durch eine definierte Umschaltung lediglich minimale Betriebsunterbrechungen auftreten.

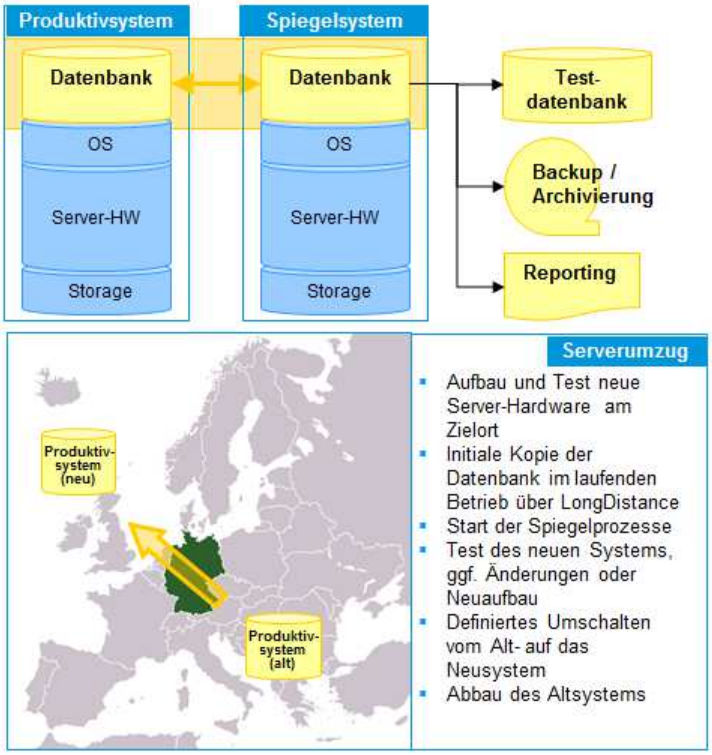

Abb. 1: Zusatznutzen Standby-System

Im Folgenden werden Architekturen dargestellt und besprochen, die genau die genannten Problemstellungen abdecken und IT-Organisationen die Möglichkeiten geben den täglichen Betrieb dieser Architekturen zu vereinfachen.

Das Konzept der Spiegelung: Hardware vs. logische Spiegelung

Das Konzept der Datenspiegelung basiert auf der Idee Datenveränderungen eines produktiven Systems auf ein Spiegelsystem zu übertragen, um diese für den Desasterfall dort zur Verfügung zu haben.

Prinzipiell gibt es zwei Herangehensweisen die Änderungen zu betrachten und abzugleichen. Zum einen die logische Ebene und zum anderen die Blockebene. Grundlegende Unterscheidungsmerkmale liegen hierbei in den Logical Units of Work mit Blick auf die transaktionale Integrität.

Spiegelung auf Blockebene

Die Spiegelung auf Blockebene arbeitet rein auf Blocklevel und somit auf der von der Applikation am weitesten entfernten Ebene. Es werden keine aus Systemsicht logisch zusammengehörende Transaktionen, sondern rein Block-Veränderungen auf der Hardwareebene übertragen. Das bedeutet im Falle einer Notumschaltung, dass z.B. langlaufende Transaktionen unvollständig recovert werden

können und somit die Datenbank inkonsistent ist. Dies wiederum muss dann im Rahmen eines Datenbank Crash Recovery behoben werden. Vor allem bei großen Datenbanken kostet dies viel Zeit und führt dabei nicht einmal in allen Fällen zum Erfolg.

Ein weiterer Schwachpunkt der Hardwarespiegelung besteht darin, dass physikalische Fehler auf dem Spiegelsystem (z.B. korrupte Blöcke) erst erkannt werden, wenn nach dem Umschalten vom Produktiv- auf das Notsystem darauf zugegriffen wird. Es laufen weder Betriebssystem-, noch Filesystem- oder Datenbank-Prozesse, die prüfen können ob die übertragenen Daten sauber weggeschrieben wurden. Dies bedeutet also im schlechtesten Fall: Produktivsystem nicht erreichbar/abgeschaltet/korrupt, Notsystem mit fehlerhaften Storageblöcken.

Weitere Nachteile blocklevel-/storagebasierter Lösungen ergeben sich vor allem im WAN-Umfeld, wie beispielsweise

- Massiver Bandbreitenbedarf im WAN-Umfeld, da jede Veränderung in einer Datenbank redundant übertragen wird: die Änderung im Datafile, die Änderung im Logfile, Rollback-Segmente etc.
- Bei Netzwerkstörungen oder unterbrochener WAN-Verbindung sind auch bei asynchroner Spiegelung zeitaufwändige Resynchs notwendig, bei Synchronspiegelungen wird sogar der Produktivbetrieb des Systems beeinträchtigt.
- Häufig identische Hardwareumgebungen für Produktiv- und Spiegelseite notwendig
- Häufig zusätzliche manuelle Arbeiten im Zuge einer Umschaltung notwendig, z.B. Mounten der Laufwerke etc.
- Umschaltung ganz oder gar nicht: Sind mehrere Systemeinheiten in einer consistency group definiert, werden diese alle gemeinsam umgeschaltet.
- Reine Hardwareumschaltung: Entscheidet sich der Systemverantwortliche für eine Storageumschaltung, wird genau diese gemacht, weitere Aktivitäten sind individuell zu handhaben.

Zusammenfassend zeigt sich somit, dass speziell bei der Desastervorsorge auf Basis der Spiegelung an entfernte Standorte die blocklevelbasierte Spiegelung immense Herausforderungen mit sich bringt.

Spiegelung auf logischer Ebene

Wesentlich einfacher und flexibler zu handhaben sind dagegen Spiegelungen auf logischer Ebene. Bei der logischen Datenspiegelung bestehen die Logical Units of Work in den tatsächlichen logischen Änderungen der Datenbanken und Dateisysteme. Konkret fokussiert sie sich somit auf die Datenbank-Archive-Logs und die geänderten Directories und Dateien.

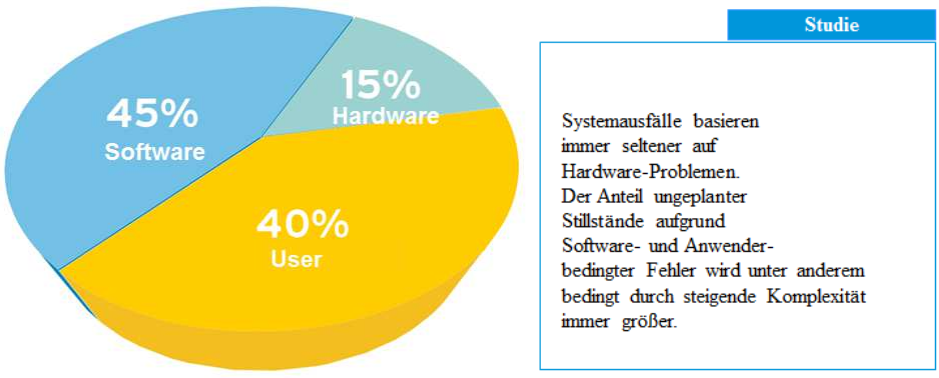

Abb. 2: Ursachen für Systemausfälle

Laut Studien lassen sich etwa 70% aller Ausfälle auf logische Fehler zurückführen. Unter logischen Fehlern sind Datenkorruptionen oder -löschungen aufgrund von Benutzer- oder Softwarefehlern zu verstehen. Nur durch ein zeitversetztes Recovery kann auf diese reagiert werden.

Hier bietet Oracle auf Datenbankebene beispielsweise mit dem Konzept der Standby-Datenbanken unterschiedliche Methoden der Realisierung an.

Neben der Datenbankebene gilt es jedoch auch die Applikationsebene zu betrachten. Zahlreiche Applikationen schreiben im laufenden Betrieb umfangreiche Daten außerhalb der Datenbank. Diese Applikationen sind nur dann mit einem aktuellen Stand auf dem Standby-System lauffähig, wenn diese Daten in das Ausfallrechenzentrum übertragen werden. Hier ist z.B. SAP mit dem Java-Stack, Profilen, Transportverzeichnissen etc., sowie den eigentlichen Programm-Binaries zu nennen. Aus diesem Grund ist es in vielen Fällen nicht ausreichend auf eine reine Datenbankspiegelung zurückzugreifen. Es bedarf eines integrierten Ansatzes zur ganzheitlichen Applikationsspiegelung. Hierfür bietet sich neben der individuellen Entwicklung von Skripten auch die Möglichkeit auf den Einsatz entsprechender Produkte von Drittanbietern, wie beispielsweise der Firma Libelle.

Best-Practice/Betriebserfahrung: Anforderungen an die Hardware

Unabhängig der Umsetzung einer Applikationsspiegelung mit individuellen Mitteln oder mit Standardprodukten lassen sich für die meisten Szenarien Hardwareanforderungen wie folgt definieren:

Produktiv-System

Für das bestehende oder geplante Echt-System gilt es zu betrachten, ob für die Spiegelung zusätzliche Anforderungen an die Hardware entstehen. Dies könnte ein zusätzliches Netzwerkinterface sein oder auch weitere CPUs. Grundsätzlich sollten die Auswirkungen der Spiegelung auf den operativen Betrieb gering gehalten werden. Somit kann über ein dediziertes privates Netzwerk der Netzwerkverkehr für den normalen Applikationsbetrieb und für die Spiegelung sauber getrennt werden. Damit die Kommunikation möglichst reibungslos läuft, sollte ein eigener Netzwerk-Port definiert und in beide Richtungen freigeschaltet sein.

Auch kann über ergänzende CPU-Ressourcen die Gefahr vermindert werden, dass die zusätzlich erforderliche Rechenleistung, welche beispielsweise zur Komprimierung der Daten vor dem Versand über das Netzwerk notwendig ist, die Produktion ausbremst.

Viele Peaks bezüglich Transaktionslast treten nachts oder am Wochenende auf. Hierauf kann über die Virtualisierung reagiert werden, um situationsabhängig Rechner-Ressourcen wie Memory und CPU zuzuteilen. Gerade im Hinblick auf WAN-Spiegelungen über Zeitzonen hinweg lassen sich somit sehr gut die Ressourcenanforderungen anpassen.

Standby-System

Das Standby-System muss nicht identisch wie das Echt-System aufgebaut sein. Hier werden oft ausgemusterte Produktivrechner herangezogen, oder bestehende DEV-/QS-Systeme übernehmen während des Normalbetriebes die zusätzliche Funktion als Standby-System. Im Umschaltfall treten dann DEV/QS in den Hintergrund, das aktivierte Standby-System erhält die vollen Ressourcen für den produktiven Notbetrieb.

Desweiteren lässt sich auch eine Cross-Over-Spiegelung realisieren. So kann eine produktive Maschine zusätzlich als Standby-System für eine andere Applikation genutzt werden. Allerdings gilt es zu berücksichtigen, dass das Standby-System genügend Ressourcen zur Verfügung haben sollte, um

im Fehlerfall dort auch wirklich den produktiven Betrieb mehrerer Produktionen über einen längeren Zeitraum fahren zu können.

Netzwerk

Grundlage der Kommunikation bildet das Standard-TCP/IP-Protokoll. Der physikalische Typ der Verbindung ist nahezu beliebig. Hier kann von Funk- über Satellitenverbindung, über Verbindungen im Internet bis zu Standleitungen Unterschiedlichstes umgesetzt werden.

Im Gegensatz zum lokalen Spiegelbetrieb steht bei WAN-Umgebungen in der Regel jedoch nur eine sehr limitierte Bandbreite zur Verfügung. Generell ergeben sich unter anderem dadurch folgende Probleme:

- Hohe Bandbreitenanforderung resultiert in hohen laufenden Kosten
- Latenzzeiten im Netzwerk führen zu ineffizientem Datentransfer
- Unstabile Leitungen erfordern ggfs. regelmäßig manuelle Eingriffe
- Intransparente Replikationslösungen bieten kaum Kontrollmöglichkeiten und Aufsetzungspunkte bei Netzwerkstörungen
- Datenkonsistenz aus Applikationssicht wird oft nicht gewährleistet
- Typischerweise sehr hohen Anfangsinvestitionen

Bei der Kalkulation der Bandbreitenanforderung für den Normalbetrieb muss das Volumen der Änderungsdaten der Datenbank und Applikation berücksichtigt werden. Die erforderliche Bandbreite ist von Datenbankstrukturen, Datenbankgrößen und Nutzungsverhalten (z.B. nologging-transaktionen) abhängig und von System zu System unterschiedlich.

Für die Bandbreitenkalkulation sind auch Prüf- und Verwaltungsverkehr im Netzwerk zu berücksichtigen. Reduzierend wirkt sich eine etwaige Kompression aus. Oracle-Archive-Logs lassen sich hier erfahrungsgemäß mit rund 30% recht gut komprimieren.

Spitzen im Datenaufkommen auf dem Primärsystem tauchen oft während Zeiten intensiver manueller oder automatisierter Dateneingabe auf. Zu berücksichtigen ist dabei, dass sich bei limitierter Bandbreite definierte Recovery Point Objectives (maximaler Datenverlust nach einem Totalausfall) in diesen Zeiten eventuell nicht halten lassen. Eine Bandbreitenberechnung mit historischen Daten bietet hier Planungssicherheit.

Bandbreitenengpässe und aufgestaute Veränderungsdaten müssen somit zeitweise auf dem Primärsystem zwischengespeichert werden. Bereits bei der Planung sollte dafür in einer Bandbreitenberechnung entsprechend Plattenplatz vorgesehen werden. Die Übertragung großer Datenmengen erfordert hohe Bandbreiten, was sich speziell im WAN-Umfeld häufig in steigenden Leitungskosten widerspiegelt. Aus diesem Grund sollte das Optimierungspotenzial für die Netzwerkkommunikation geprüft werden. Hier gilt es die folgenden Punkte zu betrachten:

- Paketgrößen
- Latenzzeiten
- Offene Sockets (Parallelisierung)
- Netzwerkunterbrechungen / Quality of Services (QoS)
- Nologging-Transaktionen
- Kompressionsfaktoren

Die Paketgrößen sollten möglichst groß gewählt werden, um das Verhältnis von produktiven Daten und Verwaltungsoverhead für die Netzwerkkommunikation zu optimieren. Um die gegebene Bandbreite möglichst gut auszuschöpfen, sind die Datenpakete parallel übers Netz zu versenden.

Somit muss nicht jeweils die Latenzzeit abgewartet werden, bevor das nächste Paket verschickt werden kann.

Initial Copy

Input for Calculation		
Database Size	1000	Gbyte
Compression Factor*	3,5	
Parity Bit Overhead	15,00%	
Application Overhead	8,00%	
Bandwidth**		Mbit/s
Projected Copy Duration**	24,00	hours

Conversion Table	...bit	...byte
Kilo	8.388.608.000,0000	1.048.576.000,0000
Mega	8.192.000,0000	1.024.000,0000
Giga	8.000,0000	1.000,0000
Tera	7,8125	0,9766

Results		
Database Size	1000,00	Gbyte
Bandwidth	33,32	Mbit/s
Projected Copy Duration	24,00	hours

Archive Log Shipping

Input for Calculation		
Volume Archive Logs/day	20	Gbyte
Compression Factor*	2,5	
Parity Bit Overhead	15,00%	
Application Overhead	8,00%	
Bandwidth**		Mbit/s
Projected Copy Duration	24,00	hours

Conversion Table	...bit	...byte
Kilo	167.772.160,0000	20.971.520,0000
Mega	163.840,0000	20.480,0000
Giga	160,0000	20,0000
Tera	0,1563	0,0195

Results		
Volume Archive Logs/day	20,00	Gbyte
Bandwidth	0,93	Mbit/s
Projected Copy Duration	24,00	hours

Abb. 3: Netzwerkkalkulation

Bei Netzwerkunterbrechungen sollte ein reibungsloses Wiederaufsetzen automatisiert werden können. Jeder notwendige manuelle Eingriff bedeutet eine gewisse Unsicherheit und auch eine zeitliche Unterbrechung in der Synchronisation und somit Datenverlust im Fehlerfall. Nologging-Transaktionen kommen laut DBA eigentlich nie vor. Die Praxis zeigt jedoch, dass diese z.B. bei Dataloads oder Reorganisationen diese häufig genutzt werden, was jedoch nicht in jedem Desaster-Recovery-Konzept vorgesehen ist.

Da die Netzwerkstrecke im WAN-Umfeld häufig der Engpass in der Architektur ist, sollten die Daten vor dem Versand komprimiert werden. Die Komprimierung bietet noch einen weiteren Vorteil: Daten werden hierbei auch verschlüsselt. Hierzu können auch ssh-Tunnel zur Komprimierung genutzt werden.

Aufbau der Spiegel-Umgebung

Für die initiale Übertragung der Datenbank können vorhandene Datenbank-Mittel genutzt werden. Hierfür bieten sich unterschiedliche Methoden an. Das initiale Aufbauen der Standby-Datenbank kann über eine Kopie über das Netzwerk oder per Backup/Restore durchgeführt werden. Die Datenbank kann bei einer Copy über das Netzwerk mittels eines Online-Backups übertragen werden. Dazu sind die einzelnen Tablespaces nacheinander in den „begin backup“-Mode zu versetzen und die

dazugehörigen Datafiles auf den Standby zu übertragen. Aus Performance-Gründen sollte man nicht die komplette Datenbank in den „begin backup“-Mode bringen.

Die entsprechenden, zur Applikation gehörenden Flat Files können mittels rsync (Unix) bzw. robocopy (Windows) kopiert werden. Möglichst identische Filesystemstrukturen auf Produktiv- und Standby-System erleichtern die Kopie und später auch den möglichen Backup und Restore.

Aufgrund der Leitungsbegrenzung kommt die Kopie über das Netzwerk speziell im WAN oft nicht in Frage.

Beim Erstellen der initialen Kopie über Backup/Restore mit Hilfe externer Medien, wie z.B. Tapes, gilt es folgende Punkte zu berücksichtigen:

- Band lesbar auf dem Standby-System
- Zollbestimmungen zum Transport der Tapes/Daten
- Änderungen der Datenbank und der Dateien vom Start des Erstellens des Backups bis zum endgültigen Restore der Daten auf dem Standby-System. Hierzu gehören sicherlich die Archive-Logs der Datenbank, aber auch Nologging-Transaktionen in der Datenbank und ggf. neue Tablespaces oder Datafiles. Auf Flat File-Ebene muss ermittelt werden, was sich während des Erstellens des Standby-Systems an Daten geändert hat.

Herausforderungen im laufenden Betrieb

Im laufenden Betrieb sind für die Datenbank die Archive-Logs zu replizieren. Hierbei sollte nicht nur das Vorhandensein der Archive-Logs geprüft werden, sondern auch ob die vorhandenen Archive-Logs von Oracle bereits abgeschlossen wurden. Ansonsten werden die Archive-Logs korrupt auf das Standby-System übertragen. Darüber hinaus sollte die Datenbank zyklisch auf Strukturänderungen wie neue Tablespaces, Datafiles, Nologging-Transaktionen oder Änderungen der Oracle-Parametrisierung geprüft werden. Auch diese Veränderungen müssen auf der Spiegelseite nachgezogen werden.

Auf Filesystemebene muss nach editierten und gelöschten Files und Directories gesucht werden, genauso wie nach Permission- und owner:group-Änderungen. Volumen und Peaks können in der Datenbank über die Information der Logswitches gewonnen werden. Die Ermittlung des Volumens der Änderungen pro Tag und Peaks auf Flat File-Ebene kann nur auf historischen Erfahrungswerten basieren. Auch gilt es zu ermitteln, inwieweit die Daten komprimiert werden können und welche CPU-Auslastung hierfür notwendig ist.

Durch das Splitten eines Archive-Files und die parallele Übertragung kann, wie bereits oben erwähnt, die Bandbreite möglichst gleichförmig ausgelastet werden. Somit kann zusammengefasst werden, dass die Kalkulation der notwendigen Bandbreite für die laufende Spiegelung der Datenbank und Flat Files von den folgenden Faktoren abhängt:

- Anzahl und Größe Log-Dateien und Änderungen der Flat Files pro Intervall (z.B. pro Tag oder pro Stunde)
- Verfügbare Bandbreite
- Belastungsspitzen
- Erreichbarer Kompressionsfaktor
- Anzahl paralleler Prozesse

Für die Synchronisierung von Datenbanken, Applikationen und Dateien können Kombinationen aus unterschiedlichen Tools und eigenen Skripten genutzt werden. Der Einsatz einer solchen Kombination unterschiedlicher Methoden bedeutet jedoch wiederum eine deutlich höhere Komplexität in Architekturen und Abläufen, sowie drastisch höhere Wartungsanforderungen. Deswegen bietet der

Markt Standardprodukte wie z.B. Libelle an, die eine Automatisierung und Integration ermöglichen, welche mit obigen Tools nicht sichergestellt werden können.

Szenarien: Arbeiten im Notbetrieb und Wiederherstellung des Normalbetriebs

Im Fehlerfall sollten Datenbank und Applikation möglichst schnell und weitgehend automatisiert auf dem Standby-System aktiviert werden können. Dabei muss aber vorab schon definiert sein wie auf welche Szenarien reagiert werden soll. Solche Szenarien können sein:

- Hardwareausfall
- Logische Fehler (z.B. korrupte Daten, fehlerhafte Software-Updates)
- Maintenance

Bei Hardwareausfällen empfiehlt es sich auf einen möglichst aktuellen Zeitpunkt zu recovern. Auf welchen Zeitpunkt maximal recovert werden kann ist hier abhängig, wann die letzten Änderungsdaten vor dem Fehler übertragen wurden.

Bei logischen Fehlern soll lediglich bis kurz vor den Zeitpunkt des logischen Fehlers recovert werden. Für die Analyse des Zeitpunkts des logischen Fehlers kann hier der Oracle Logminer ein sehr hilfreiches Tool sein. Auch ist es nicht immer erforderlich auf dem Standby-System produktiv zu gehen. Ist man mit der Datenbankstruktur vertraut und sind somit die Abhängigkeiten der Tabellen bekannt, kann die Standby-Datenbank read only geöffnet werden. Konsistente Daten des Spiegelsystems können mittels export/import auf dem Produktiv-System wieder eingespielt werden.

Ist die Performance und Konnektivität auf dem Standby-System nicht allzu gravierend geringer, kann das Standby-System auch dafür genutzt werden bei Maintenance-Arbeiten am Echt-System die Downtime zu minimieren, indem während der Wartungsarbeiten am Produktivsystem die Benutzer auf dem aktivierten Standby-System produktiv weiterarbeiten. Mittels entsprechender Mechanismen muss dafür gesorgt werden, dass die während der Wartungsarbeiten anfallenden Veränderungen des Standby-Systems auf das eigentliche Produktivsystem zurückgespiegelt werden. Nach Abschluss der Wartungsarbeiten wird in den Normalbetrieb zurückgeschaltet.

Die gesamte Kommunikation zwischen Clients, Applikationsserver und Datenbankserver findet über eine spezifizierte IP-Adresse, Hostname oder NetBIOS-Name statt. Wird eine Datenbank oder ein Filesystem von einem Server auf einen anderen umgeschaltet, so ist die Anwendung normalerweise nicht in der Lage diesen anderen Server automatisch zu identifizieren. Generell setzt eine Umschaltung auf einen anderen Datenbankserver mehrere manuelle Eingriffe in der Anwendung (z.B. SAP R/3-Profiles) voraus. Dies führt zu längeren Downtimes für die Applikation und Anwender.

Ist die Möglichkeit gegeben, dass im Notbetrieb über eine Virutelle IP-Adresse (VIP) die bisherige Konnektivität wieder verfügbar gemacht werden kann, sind keine Anpassungen für den connect notwendig. Dennoch kann es für die Umschaltung erforderlich sein, Prozesse auf Applikationsservern im Netzwerk durchzustarten. Dies kann über Agents oder Skripte auf den Applikationsservern umgesetzt werden, welche bei der Umschaltung angetriggert werden.

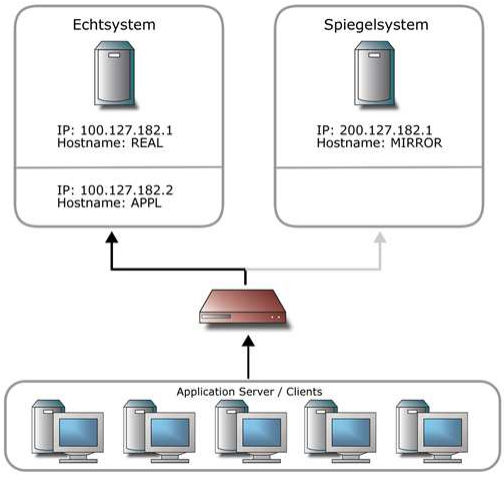

Abb. 4: Applikationszugriff über VIP

In einem WAN-Umfeld muss zusätzlich neben dem Umziehen der IP-Adresse/Hostname dem Netzwerk bzw. den Netzwerkkomponenten, wie Domain Name Server oder Routern, noch mitgeteilt werden, in welchem Netz bzw. Netzklasse der Notfallrechner zu finden ist. Dies wird realisiert durch die Anpassung der entsprechenden Routing-Einträge. Somit ist es möglich dass die virtuelle IP in einer anderen Netzwerkklasse gefunden wird. Für die Frage „Wie kommen die User an das Standby-System?“ können folgende Lösungen in Betracht gezogen werden:

- Identische IP über VLAN oder DNS-Anpassung oder
- geänderte Logon-Konfiguration (Hiermit lässt sich z.B. sehr einfach der Zugriff auf die Key-User einschränken)

Für die Applikation kann es auch notwendig sein einzelne Mountpoints bspw. von SAN-Storage umzuziehen, vorausgesetzt es sind noch Storage-Platten am Standort des Produktivsystems oder einem weiteren Standort vorhanden, auf die technisch zugegriffen werden kann.

Wurde erfolgreich auf das Standby-System umgeschaltet, die Applikation aktiviert und die Benutzer mit diesem verbunden, ist die Grundlage für die Weiterführung des Geschäftsbetriebes hergestellt. Jedoch sollte auch bedacht werden wie der Umzug zurück auf das Produktiv-System realisiert werden kann. Meist hat das Standby-System eine geringere Performance und/oder die Konnektivität ist eingeschränkt. Durch entsprechende Vorarbeiten im normalen Betrieb kann vermieden werden, dass die Spiegelung eine reine Einweglösung ist.

Schlussbemerkung

Die Möglichkeiten eine Spiegelung über WAN einzurichten und in Betrieb zu halten sind vielfältig. Hier gilt es vorab sauber zu definieren, welche Ziele mit der Spiegelung erreicht werden sollen, um die optimale Mischung aus Performance, Kosten und Verfügbarkeit umsetzen zu können.

Kontaktadresse:
Franz Diegruber
Libelle AG
Gewerbestr. 42
D-70565 Stuttgart
Telefon: +49 (0) 711-78335 312
Fax: +49 (0) 711-78335 340

E-Mail fdiegruber@libelle.com
Internet: www.libelle.com