

Mit den stetig zunehmenden Ansprüchen an die Verfügbarkeit von Systemen wächst der Bedarf an ausfallsicheren Lösungen. Die Einbindung mehrerer Rechner in einen Shared All Cluster kann den Zugriff auf die Datenbank sichern, indem bei Ausfall eines Knotens einer der verbleibenden die Verbindung zur Anwendung übernimmt (reconnect). Hier setzt Oracles Real Application Cluster (RAC) an.

Prinzipiell setzt RAC voraus, dass die Instanzen auf den einzelnen Nodes sich die Control- und Tablespace-Dateien teilen. Voraussetzung ist daher der Zugriff auf gemeinsamen Massenspeicher.

Professionelle Lösungen arbeiten mit Disk Arrays auf Fibre-Channel-Basis, die mehreren Rechnerknoten das Arbeiten mit gemeinsamen Platten über Punkt-zu-Punkt-Verbindungen oder Switches gestatten. Ein teurer Spaß, denn für eine Lösung mit nur zwei

Knoten zuzüglich der Fibre-Channel-Platten und Switches kommen schnell Summen im fünfstelligen Bereich zusammen. Sofern eine solche Konstellation später in den produktiven Einsatz gehen soll, lohnt sich die Investition: 49 % aller Downtimes werden einer Studie des *Disaster Recovery Journal* zufolge durch Hardware- und Systemfehler verursacht, genau die Fehlerklasse, um die es bei RAC geht.

Oracle Real Application Cluster auf Firewire Fremde Wege

Andrea Held

Hochverfügbare Systeme und SAN-Speichercluster sind im professionellen Bereich in der Regel über Fibre Channel gekoppelt. Ein teures Unterfangen, wenn es nur um einen Testaufbau gehen soll. Oracles Clustersoftware bietet unter Linux eine Alternative: kostenlose Treiber für den Betrieb über Firewire.

Ohne Fibre, aber mit Fire

Möchte man aber nur einige Tests mit dem RAC durchführen, dürfte in Zeiten von Sparsamkeit wohl nur sel-

X-TRACT

- Für sein Real Application Cluster bietet Oracle einen Patch an, der den Betrieb auf Basis von Firewire erlaubt.
- Die Lösung ist allein für Tests gedacht und nur unter Linux verfügbar.
- Eine solche Konfiguration ist erheblich kostengünstiger als die mit Fibre Channel im produktiven Betrieb.
- Um den gemeinsamen Zugriff mehrerer Knoten auf einen Massenspeicher zu regeln, verwendet Oracle die von DEC entwickelte Technik „Cache Fusion“.



Komponenten für RAC über Firewire

Mindestens zwei Rechner mit Firewire -Schnittstelle

Je zwei Karten pro Rechner – je eine für den Cluster Interconnect und eine weitere für das Public Network

Externes Firewire-Gehäuse mit IDE-Harddisk

Oracle 9i Software (etwa per Download von Oracle Technet)

Oracle 9i RAC Firewire Patch

ten eine so kostspielige Konfiguration die Zustimmung eines Budget-Verantwortlichen finden.

Doch es gibt eine preisgünstige Alternative: Mittels eines frei erhältlichen Patches lässt sich RAC auf Basis einer kostengünstigen Firewire-Disk installieren. Die Kosten der „Testlösungen“ liegen bei etwa 10 % der professionellen. Oracle bietet für Linux Plattformen noch weitere Features: Ein eigenes Cluster-File-System (OCFS), Erweiterungen für den NIC Failover sowie einen Watchdog-Treiber. Letzterer prüft automatisch auf einen Out-of-Sync zwischen Cluster-Knoten und einer Shared Disk und leitet – um Beschädigungen der Daten vorzubeugen – notfalls einen Shutdown ein. Oracles Linux Team hat die frei verfügbaren Libraries samt Patches in C entwickelt und die Erweiterungen unter GPL gestellt.

Für ein Cluster in der Minimalkonfiguration sind zwei Rechner und zusätzlich je eine Karte für das Public

sowie für das Private Network notwendig. Das „öffentliche“ Netz liefert später die Verbindung der Client-Sessions mit den Instanzen, während das „private“ den Kontakt zwischen den Clusterknoten herstellt. Über Letzteres melden sich die Knoten mit ihrem Heartbeat, um zu signalisieren, dass sie noch „leben“ und arbeitsfähig sind. Wenn das Signal bei einem ausbleibt, gehen die übrigen Knoten davon aus, dass der betreffende Rechner ausgefallen ist. RAC verteilt über das private Netz darüber hinaus Information aus Shared-Memory-Segmenten der im Cluster beteiligten Instanzen. Diese Technik nennt sich Cache Fusion, die Oracles Shared Disk Cluster zu einer gemeinsamen System Global Area verhilft (siehe Kasten „Shared Disk Storage“).

Für den Anschluss der externen Harddisk über Firewire kann man eine handelsübliche IDE-Festplatte in ein Firewire-Gehäuse einbauen. Sofern nur zwei Knoten im Cluster involviert

sind, ist kein zusätzlicher Firewire-Hub nötig. Das Gehäuse besitzt zwei Anschlüsse, die normalerweise als Eingang und Ausgang bei Verkettungen über den Bus dienen. Neben der externen Platte muss jeder Rechner mindestens einen Firewire-Anschluss haben. Ausbauen lässt sich ein solcher Cluster mit Hubs: ein 4-Port für drei und ein 5-Port für vier Knotenrechner.

Günstige Testplattform nur für Linux

Mindestanforderungen an die Rechner sind eine 500-MHz-CPU, 256 MByte RAM, 3 GByte freier Plattenplatz, für Firewire ein Chipset von Texas Instruments (TI) sowie zwei Netzwerkkarten. Die Rechnerknoten müssen jedoch nicht identisch sein. Die im Test verwendete Hardware unterscheidet sich nicht nur in der CPU-Leistung und RAM-Ausstattung, sondern auch in der internen Festplattenkapazität. Schwierigkeiten sind dadurch nicht entstanden.

Beim Betriebssystem für das Billig-Cluster hat man nur die freie Wahl zwischen Red Hats Enterprise Linux AS/ES 2.1, Suses SLES7 und United Linux 1.0 SP1. Support gibt es keinen, alles geschieht sozusagen „auf eigene Gefahr“.

Oracles Software inklusive Clusteroption gibt es entweder direkt bei

Shared Disk Storage

Professionelle Lösungen nutzen Fibre Channel, das wahlweise Zugriff über Point-to-Point-Verbindungen oder über Switches bietet. Unterstützte Protokolle sind SCSI sowie IP. Fibre Channel schafft bis zu 127 Cluster-Nodes mit bis zu 2,12 GByte/s.

Firewire (beziehungsweise i.LINK) stammt ursprünglich aus dem Gemeinschaftsprojekt „High Speed Serial Bus“ von Apple Computer und Texas Instruments und schlug sich später im IEEE-1394-1995-Standard nieder. Es unterstützt verschiedene Geschwindigkeiten auf demselben Bus und fand bislang vorrangig Verwendung zur Anbindung von digitalen Kameras und Videogeräten. Mit Transferraten von derzeit bis zu 400 MBit/s bietet Firewire das etwa 30-fache an Bandbreite gegenüber USB. Firewire ist hot-plugable, unterstützt bis zu 63 Geräte an einem Bus und gestattet Kabellängen bis zu etwa 4 Metern [1].

Treiber sind unter Windows, Mac OS und Linux gängig, fehlen bei den „großen“

Unix-Derivaten in aller Regel. Shared Access ist über Firewire zunächst nicht möglich. Der von Oracle gelieferte Patch setzt hier an und bohrt den Treiber so auf, dass mehrere PCs einen nicht exklusiven Zugriff auf die Disk erhalten. Hierzu haben die Entwickler die Bitmaske entfernt, die zum Identifizieren der Maschine beim Logon dient. Benutzen beide Nodes denselben modifizierten Treiber, können sie parallel auf die Geräte zugegreifen.

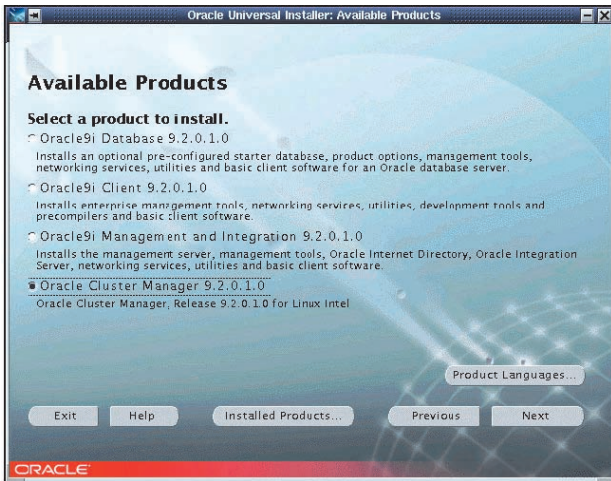
Cache Fusion versus Block Pinging

Die Clusteroption von Oracle blickt auf eine lange Geschichte zurück: 1989 für Vax/VMS erstmalig entwickelt, trug die Clusteroption von Oracle bis Version 8.1x den Namen Oracle Parallel Server (OPS). Der 9iRAC, Nachfolger des OPS ist heute auf den meisten C/S-Betriebssystemen verfügbar und nutzt nun einen Integrated Dis-

tributed Lock Manager (IDLM). Dieser koordiniert den gemeinsamen Datenzugriff im Cluster über alle Clusterinstanzen.

Kritisch war bis 8i vor allem der Austausch von Informationen bei konkurrierendem Zugriff auf dieselben Daten von mehreren Cluster Nodes. Benötigte ein Node einen Datenbankblock, den ein anderer Node aktuell ebenfalls im Zugriff hatte, so hat der anfordernde Knoten eine Nachricht an den besitzenden Knoten versandt. Der Eigentümer schrieb daraufhin den Datenbankblock auf Platte zurück, sodass der andere den Block lesen konnte. „Block Pinging“ nennt sich diese Technik, die einen unschönen Performanceverlust bei konkurrierenden Zugriffen zur Folge hat.

Um Plattenlasten zu vermeiden, hat Oracle die von DEC entwickelte Cache-Fusion-Technik in den 9iRAC implementiert. Alle Cluster Nodes nutzen einen Global Shared Cache, der bei konkurrierenden Zugriffen per Block Shipping über den Cluster-Interconnect Daten austauscht.



Gemeinsame Sache: Mit redundantem internem Clusternetz und gemeinsamen Plattenzugriff schützt sich Oracles Real Application Cluster vor Ausfällen (Abb. 1).

www.oracle.com oder aus Oracles Technet (OTN) per Download. Um das OTN nutzen zu können, muss man sich einmalig registrieren. Im OTN ist unter anderem der gepatchte Treiber inklusive der Quellen zu finden.

Die Installation eines RAC weicht ein wenig von der eines Oracle 9i ab und gliedert sich in folgende Schritte:

1. Einbau und Anschluss aller Hardwarekomponenten (NICs, Firewire - Karten, externe Harddisk),
2. Installation des Betriebssystems, Konfiguration aller Komponenten,
3. Hardwarefunktionstest (insbesondere Private und Public Network sowie Zugriff auf externe Festplatte via Firewire),
4. Installation des Clustermanagement Software/Operating System Dependent Layer (OSD) mit *runInstaller*,
5. Installation der Oracle Software mit *runInstaller*,
6. Implementierung des Firewire Patches,
7. optional: Installation des Cluster File System (ohne CFS müsste man

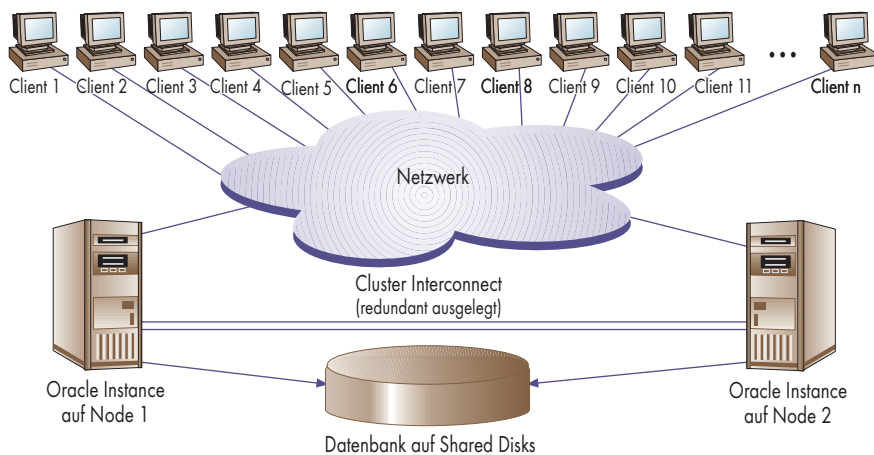
Raw Partitions auf den Shared Disks verwenden),

8. Konfiguration des Clustermanagers sowie
9. Erstellen der Datenbank.

Einarbeitung unbedingt erforderlich

Vor Installation eines Clusters sollte man sich die RAC-Dokumentation sowie die Installationshinweise in Oracles Metalink (184821.1) zu Gemüte führen. Ein erfahrener Datenbanker mit Linux-Hintergrund kann mit sorgfältiger Vorbereitung die Erstinstallation eines Mini-Clusters mit zwei Nodes auf Firewire-Basis in ein bis zwei Tagen schaffen. Sonst geht es nicht ohne einige Tüftelei – eine Clusterinstallation ist aber letztlich keine Zauberei.

Nach dem Einrichten des Betriebssystems sollte man sich tunlichst der Überprüfung der Grundfunktionen widmen, insbesondere der Netzverbindun-



Vorfahrt: Als Erstes gilt es, den Clustermanager zu installieren, der aktive Knotenrechner voraussetzt (Abb. 2).

Listing 1: Konfiguration des CM/OSD

```
# cmcfig.ora
HeartBeat=10000
ClusterName=Oracle Cluster Manager, version 9i
PollInterval=300
MissCount=20
PublicNodeNames=node1 node2
ServicePort=9998
WatchdogSafetyMargin=3000
WatchdogTimerMargin=6000
HostName=node1
```

und des Zugriffs auf die externe Harddisk via Firewire, da ein späteres Beheben von Störungen wesentlich mühseliger ist. Denkbar einfach sind die Konfigurationen des Private und des Public Network. Zwar ist der Clientzugang für die reine Installation von Oracle-Software und Datenbank nicht relevant, ohne ihn ist aber kein Zugriff von außen auf die Clusternodes möglich und damit wäre der Zweck des Ganzen etwas verfehlt. Um hier ebenso wenig nach dem Installieren in eine Falle zu laufen, sollte man den Zugang zum Public Network konfigurieren und testen, bevor es mit dem Einrichten der Software weitergeht. Einzutragen sind für das Public Interface jedes Clusterknotens die Netzmaske und je eine freie IP-Adresse des Subnetzes, in dem Clusterknoten hängen.

Feste Namen im Privaten

Da die Rechner im Private Network unter sich bleiben, sind keine „offiziellen“ IP-Adressen notwendig. Relevant ist nur, dass die Verbindung steht. Damit sich die Knoten auf dieser Ebene beim Namen nennen können, darf der

Eintrag in */etc/hosts* nicht fehlen. Findet Oracles Clustermanager einen Knoten über Namensauflösung nicht, verweigert er beim Installieren die Identifikation.

Den Zugriff auf die Firewire-Platte sollte man unbedingt für jede Maschine zunächst einzeln konfigurieren und testen. Auf einer lokalen Disk jedes Knotens sollte der Admin einen Mount Point für Oracle (etwa */oracle/home*) zur Verfügung stellen. Weiterhin benötigt die Installationsroutine eine Gruppe „dba“ und einen User namens „oracle“ als Mitglied mit Schreibrechten auf die zugehörigen Verzeichnisse.

Das Einrichten der Software kann entweder von CD oder von einem lokalen Verzeichnis aus erfolgen, das die notwendige Software enthält. Bevorzugt man letzte Variante, sollte der Inhalt der Medien in einem eigenen Verzeichnis als *Disk1*, *Disk2* und *Disk3* hinterlegt sein.

Da während des Installierens direkt von der CD die Prozedur dreimal ein Wechseln des Mediums anfordert, sollte man Oracles Installer nicht starten, solange ein Verzeichnis auf einer eingehängten CD aktiv ist. Die Folge wäre, dass das *unmount* fehlschlägt und die Installationsroutine abbricht.

Nachdem man sich als „oracle“ angemeldet hat und aus *Disk1* oder von der ersten CD das Skript *runInstaller* aufgerufen hat, erscheint ein grafisches Interface, das einen bequem durch die notwendigen Schritte führt. Alternativ gibt es die „Silent Installation“ ohne GUI. Diese erwartet ein vorkonfiguriertes „Response File“ mit allen Angaben, die das GUI sonst erfragen würde, und trotzdem einen aktiven X-Server. Muster für solche Dateien (mit

der Dateieindung *.rsp*) sind auf der Installations-CD zu finden.

Clustermanager legt die Basis

Vor allem anderen geht es ans Einrichten des Clustermanagers, der für den RAC als Operating System Dependent Layer (OSD) fungiert. Ohne diesen können die einzelnen Instanzen im Cluster nicht miteinander kommunizieren. Danach erst kommt die Software zu Oracles Datenbank an die Reihe. Hierzu bedarf es eines zweimaligen Aufrufs von *runInstaller*: Zunächst muss man die Option *Oracle Cluster Manager* wählen. Nach dessen erfolgreicher Installation kann die *Database Installation* mit der Zusatzauswahl von *full software only* – also generieren einer Datenbank – erfolgen, da diese erst später auf der gemeinsamen externen Platte entstehen soll.

Denn vorher bedarf es der Firewire-Patches: Nach Download der Datei „fw-test-kernel-2.4.20-image.tar.gz“, etwa aus Oracles Technet, folgt das Entpacken und Implementieren auf beiden Knoten mit:

```
tar zxvf /fw-test-kernel-2.4.19-image.tar.gz
modify /etc/grub.conf
```

Wer *lilo* anstelle von *grub* als Bootmanager verwendet, muss hier */etc/grub.conf* durch */etc/lilo.conf* ersetzen. An das Ende der jeweiligen Dateien muss man folgende Codezeilen für den neuen Kernel anfügen:

```
title FireWire Kernel (2.4.19)
  root (hd0,0)
  kernel /vmlinuz-2.4.19 ro
  root=/dev/hda3
```

Listing 2: Beispiel eines Initfile für ORAC

```
db_block_size=4096
db_cache_size=74217728
cluster_database_instances=2      <- Anzahl der Clusternodes (2, 3, 4)
filesystemio_options="directIO"
open_cursors=300
user_dump_dest=/home/Oracle/9i/admin/node1/udump      <- Anpassen für den jeweiligen Node
background_dump_dest=/home/Oracle/9i/admin/node1/bdumpp <- Anpassen für den jeweiligen Node
core_dump_dest=/home/Oracle/9i/admin/node1/cdump      <- Anpassen für den jeweiligen Node
timed_statistics=FALSE
db_domain=held-informatik.de      <- Domänen Name anpassen
remote_login_passwordfile=EXCLUSIVE
control_files=(/ocfs/ctrl")
db_name=bob                       <- Datenbankname anpassen
java_pool_size=12428800
large_pool_size=10485760
shared_pool_size=47440512
processes=150
fast_start_mtr_target=300
resource_manager_plan=SYSTEM_PLAN
sort_area_size=524288
undo_management=AUTO
cluster_database=true;
undo_tablespace=UNDO_ts1         <- Anpassen auf Node 2 zu undo_ts2
instance_name=node1             <- Anpassen auf Node 2
instance_number=1               <- Auf Node 2 diesen Wert auf 2 setzen!
thread=1                        <- Auf Node 2 diesen Wert auf 2 setzen!s
```

Listing 3: Beispiel für Create Database

```
cd /home/Oracle/9i
. ENV
rm -f /ocfs/orapw
rm -f /ocfs/ctrl
orapwd file=/ocfs/orapw password=manager entries=5
sqlplus << EOF
/ as sysdba
startup nomount
create database bob
  maxinstances 5
  maxlogfiles 10
  character set "we8iso8859p1"
  datafile '/ocfs/system.dbf' size 400m reuse
  default temporary tablespace tempts1 tempfile '/ocfs/temp.dbf' size 15m reuse
  undo tablespace undo_ts datafile '/ocfs/undo.dbf' size 40m reuse
  logfile
    '/ocfs/log1a.dbf' size 15m reuse,
    '/ocfs/log1b.dbf' size 15m reuse;
@?/rdbms/admin/catalog.sql
@?/rdbms/admin/catproc.sql
create undo tablespace undo_ts2 datafile '/ocfs/undo2.dbf' size 40m reuse;
alter database add logfile thread 2 '/ocfs/log2a.dbf' size 15m reuse;
alter database add logfile thread 2 '/ocfs/log2b.dbf' size 15m reuse;
alter database enable thread 2;
shutdown immediate
exit
EOF
```

Listina 4: Beispiel einer TAF-Konfiguration

```
Rac1.world =
(DESCRIPTION_LIST =
(FAILOVER = true)
(LOAD_BALANCE = true)
(DESCRIPTION =
(ADDRESS =
(PROTOCOL = TCP)
(HOST = rac_server_1)(PORT = 1521))
(CONNECT_DATA =
(SERVICE_NAME = Rac1)
(SERVER = dedicated)
(FAILOVER_MODE =
(BACKUP=rac_server_2)
(TYPE=select)
(METHOD=preconnect)
(RETRIES=20)
(DELAY=3)
)
)
)
)
```

Nach dem Einbau der Kernel-Patches ist ein Reboot unumgänglich. Um das Hochfahren so zu vereinfachen, dass der Prozess nicht jedes Mal die Modifikation durchlaufen muss, sollten in */etc/modules.conf* außerdem folgende Einträge stehen:

```
options sbp2 sbp2_exclusive_login=0
post-install sbp2 insmod sd_mod
post-remove sbp2 rmmod sd_mod
```

Bei jedem Wiederhochfahren sollte jeder Rechner seine Firewire-Treiber laden:

```
modprobe ohci1394
modprobe sbp2
```

Linux identifiziert die externen über Firewire angeschlossene Platten wie bei USB als SCSI-Geräte. Die Ausgabe von *dmesg* erlaubt eine Überprüfung und sollte etwa Folgendes enthalten:

```
Attached scsi disk sda at scsi0, channel 0, id 0, lun 0
SCSI device sda: 35239680 512-byte hdwr sectors (18043 MB)
sda: sda1 sda2 sda3
```

Die Meldung besagt, dass der Linux-Kernel eine 18-GB-Platte mit drei Partitionen erkannt hat.

Oracles Clusterfilesystem ist zu empfehlen

Zum Partitionieren der Platte mit *fdisk* sollte nur ein Rechner aktiv sein. Der zweite Knoten erkennt die vorgenommenen Änderungen beim Hochfahren. Zum Vorbereiten der gemeinsamen Platte für das Erstellen der Datenbank benötigt Oracles Software entweder den Zugriff auf Raw Devices der Shared Disk oder auf Oracles Cluster File System; der Hersteller empfiehlt Letzteres.

OCFS lässt sich nach dem Download und Entpacken mit folgendem Befehl implementieren:

```
ocfsformat -f -l <partition> -c <clustersize> -v ocfsvol
-m /ocfs -n <nodename> -u <oracle uid> -p <directory
permission> -g <oracle gid>
```

Beispiel:

```
ocfsformat -f -l /dev/sda1 -c 128 -v ocfsvol -m /ocfs
-n node1 -u 1053 -p 755 -g 1053
```

Die Datei */etc/ocfs.conf* gestattet die Konfiguration des Kernelmoduls, etwa für Node 1:

```
ipcdm:
ip_address = 192.168.1.101
ip_port = 9999
subnet_mask = 255.255.252.0
type = udp
hostname = node1
active = yes
```

Mit dem Befehl *insmod* lädt man den OCFS-Treiber dynamisch nach:

```
insmod ocfs.o name=<nodename>
insmod /root/ocfs.o name=node1
```

Zum Einhängen einer OCFS-Partition dient

```
mount -t ocfs /dev/sda1 /ocfs
```

So entsteht ein Shared File System, das dem Benutzer „oracle“ gehört und das auf jedem der beiden Knoten sichtbar ist. Jetzt kommen die letzten Schritte: Die Konfiguration des Clustermanagers und das Erstellen der Datenbank.

Als „root“ konfiguriert man auf jedem der beiden Knoten den Clustermanager durch Anpassen der Datei *ocmargs.ora* im Verzeichnis *\$ORACLE_HOME/oracm/admin* (siehe Konfiguration des Clustermanagers /OSD). Wichtig sind die korrekten Hostnamen. Abschließend erfolgt der Start des Clustermanager mit *\$ORACLE_HOME/oracm/bin/ocmstart.sh*. Dieses Skript ist ein Muster zum Start des Clustermanager Daemons, das man gegebenenfalls entsprechend anpassen kann. Damit bei einem Reboot der Clustermanager mit hochfährt, sollte dieser Befehl im passenden *rc*-Skript stehen.

Vor dem Erstellen der Datenbank geht es an das Einrichten der Verzeichnisse für die Logfiles auf den lokalen Platten der beiden Nodes, was für Node 1 so aussehen könnten:

```
cd $ORACLE_HOME
mkdir admin ; cd admin ; mkdir node1 ;
cd node1 ; mkdir udump ; mkdir bdump ;
mkdir cdump
```

Für Oracles Passwort-Datei, die die Installationsroutine später noch anlegt, sollte jeweils ein lokaler Link hinzukommen:

```
cd $ORACLE_HOME/dbs
ln -sf /ocfs/orapw orapw
```

Nun darf das Skript *CreateDatabase* laufen. Nach Starten der Datenbank und der Konfiguration von Oracles *Listener* können Benutzer darauf zugreifen.

Fazit

Nach all diesen Mühen steht einem ein echter Datenbankcluster auf Basis von Firewire mit eigenem Cluster File System zur Verfügung, auf dem man nach Herzenslust verschiedene Szenarien durchspielen kann. Oracle bietet damit einen Laboraufbau, der bis auf die Übertragungsleistung der Realität recht nahe kommt. Eine Konfiguration, die sich außerdem hervorragend als Trainings- und Schulungsplattform eignet. (rh)

ANDREA HELD

ist tätig bei der Deutschen Post Itsolution als leitende Datenbankspezialistin und lehrt an der FH Frankfurt am Main Datenbanksysteme und HA-Cluster.

Literatur

- [1] Tom Hensel, Ralph Hülsenbusch; Peripherie; Busanschluss; USB-2 und Firewire im Vergleich; *iX* 6/2003, S. 114

FUNDORTE IM WEB

Oracle und Open Source	technet.oracle.com/tech/linux/open_source.html
Zertifizierte Konfigurationen	www.oracle.com/ip/deploy/database/features/ops/certification/index.html?linux.html
IEEE 1394 for Linux	www.linux1394.org
Raw Partitions	www.fors.com/orasupp/unix/37914_1.HTM
Software und Dokumentation	otn.oracle.com/software/products/oracle9i/content.html
Installation	ftp.suse.com/pub/suse/i386/supplementary/commercial/Oracle/docs/9iR2-RAC_sles8.txt

