Über RAC und Data Guard gibt es viele Publikationen, die aufzeigen, was dahinter steckt und welche Vor- beziehungsweise Nachteile diese Konfigurationen haben. Dieser Artikel beschäftigt sich hingegen damit, wie man eine Umgebung für unterschiedliche Anforderungen hinsichtlich Performance, Verfügbarkeit und Wartbarkeit bereitstellen kann.

Best Practices für den Aufbau und Betrieb einer Hochverfügbarkeitsumgebung mit RAC und Data Guard

Johannes Ahrends, CarajanDB GmbH, und Engelbert Wystrach, selbstständiger Datenbank-Berater

In einem gemeinsamen Projekt haben die beiden Autoren zunächst untersucht, wo die Grenzen bei der Verwendung einer Single-Instance-Datenbank mit Data Guard liegen, welche Unterschiede zwischen RAC und RAC One Node bestehen, ob RAC One Node überhaupt sinnvoll ist, und wie eine automatische Provisionierung einer kompletten Umgebung aussehen kann.

Zunächst muss für das Verständnis der Herangehensweise der Unterschied zwischen Hochverfügbarkeit (HA) und Disaster Recovery (DR) dargestellt werden. Wenn wir hier über Hochverfügbarkeit sprechen, meinen wir eine Konfiguration, bei der die Datenbank im Fehlerfall auf einen anderen Rechner im gleichen Rechenzentrum geschwenkt wird, also eine klassische Cluster-Lösung. Ob dies mit einer Single-Instance-, einer RAC- oder RAC-One-Node-Datenbank realisiert wird, ist erst einmal unerheblich. Von Disaster Recovery reden wir, wenn es sich um den Schwenk auf ein anderes Rechenzentrum, also den Totalausfall eines Rechenzentrums oder um den Ausfall des produktiven Clusters, handelt. Außerdem verwenden wir hier in der Regel den Begriff „Grid Infrastructure“ (GI), wenn wir über die 11*g*-R2-Clusterware und die Verwendung von Automatic Storage Management (ASM) beziehungsweise Automatic Storage Management Cluster File System (ACFS) reden.

Data Guard oder nicht?

Zu Beginn des Projekts stand die Frage im Raum, ob Disaster Recovery mit Data Guard oder mit EMC² Symmetrix Remote Data Facility (SRDF) realisiert werden sollte. Da sich SRDF im Unternehmen in vielen Bereichen als die Lösung für DR etabliert hatte, musste erst einmal hart darum gekämpft werden, trotzdem Data Guard einzusetzen. Letztendlich konnte aber Data Guard dadurch punkten, dass es auf die Anforderungen einer Oracle-Datenbank optimiert ist und seit vielen Jahren sehr erfolgreich, wie sich später herausstellte, auch in dem Unternehmen im Einsatz ist. Ein weiterer wesentlicher Pluspunkt für eine Data-Guard-Lösung ist außerdem die Hardware-Herstellerunabhängigkeit und die damit verbundene Möglichkeit, preiswerteren Storage, wie Network Attach Storage (NAS), als zukünftige Lösung einsetzen zu können.

Über ein Self-Service-Portal soll es möglich sein, eine Datenbank für verschiedene Anwendungen automatisch zu erstellen. Dabei gibt es die Möglichkeit der Konfiguration der Größe in drei Stufen (small, medium, large) sowie die Anforderung, ob es sich um eine RAC- oder RAC-One-Node-Datenbank handelt. Ein DR mit Data Guard wird dabei in jedem Fall mit provisioniert.

RAC, RAC One Node oder Single Instance

Aufgrund von Lizenzkosten wurde ursprünglich davon ausgegangen, dass nur Single-Instance- (SI) oder RAC-Datenbanken zum Einsatz kommen sollten. Die Möglichkeiten der Verwendung von RAC One Node wurden zunächst einmal nicht weiter betrachtet, da zu dem Zeitpunkt die eingesetzten Versionen des Enterprise Manager (11*g* und 10*g*) diese Konfiguration nicht unterstützten. Auch die Einbindung in die automatisierten Betriebsprozesse war nicht möglich. Im Laufe des Projekts stellte sich allerdings heraus, dass eine HA- und DR-Lösung mit Single Instance nur mit massivem Programmieraufwand aufgebaut werden kann und RAC One Node gerade für die bessere Ressourcenausnutzung besser geeignet ist als RAC.

Es ist durchaus möglich, eine HA-Lösung mit der Single-Instance-Datenbank zu realisieren. Oracle bietet über die Grid-Infrastruktur die Möglichkeit, beliebige Anwendungen als Cluster-Ressource zu definieren und damit im Fehlerfall auf einen anderen Knoten zu schwenken. Allerdings wird bei einem Schwenk auf die Standby-Seite der Rollentausch nicht erkannt und muss über ein zusätzliches Skript manuell nachgezogen werden. Schwerwiegend ist außerdem, dass es – auch mit Enterprise Manager 12*c* – nicht möglich ist, diesen Schwenk zu erkennen. Das heißt, im Enterprise Manager wird eine neue Datenbank-Ressource sichtbar und die Historie geht komplett verloren.

Im Vergleich zu einer Single Instance ist RAC One Node sicherlich die wesentlich bessere Wahl, wenn es um die HA-Möglichkeiten geht. Aber in großen Umgebungen, in der viele Datenbanken auf dem gleichen Rechnerverbund (in diesem Fall ein Bladecenter mit acht Servern/Knoten) be-

trieben werden, hat RAC One Node aufgrund der besseren Ressourcen-Zuteilung Vorteile gegenüber RAC. Als HA-Lösung können mehrere Kandidaten angegeben werden, auf denen die Instanz wieder gestartet werden kann (Relocate). In Tests war die Anwendung sowohl im Fehlerfall als auch nach einem bewussten Schwenk (Maintenance) nach wenigen Sekunden wieder mit der Instanz verbunden. Da der Enterprise Manager 12*c* RAC One Node als Konfiguration erkennt, gibt es auch keinerlei Probleme beim DR oder Monitoring.

Natürlich war Oracle RAC zunächst das Maß aller Dinge, ist es doch die HA-Lösung schlechthin. Allerdings sind im Lauf des Projekts Einschränkungen aufgetaucht. Wenn man auf einem Blade Center mit acht Servern eine RAC-Datenbank betreiben möchte, dann lautet die Frage, wie viele Instanzen diese Datenbank haben und was geschehen soll, wenn ein Server ausfällt. In diesem Projekt wird davon ausgegangen, dass zunächst nur RAC-Datenbanken mit zwei Instanzen betrieben werden. Daraus ergeben sich folgende Möglichkeiten:

- *Betrieb als „Administrator-Managed"-Datenbank*
  In diesem Fall werden die zwei Instanzen genau zwei Servern zugeordnet. Fällt ein Server aus, wird die Datenbank nur noch mit der verbliebenen Instanz betrieben. Ein automatisches Starten der zweiten Instanz auf einem anderen Rechner ist nicht vorgesehen. Dies kann nur manuell über „srvctl modify instance" beziehungsweise ein entsprechendes Skript erfolgen.
- *Betrieb als „Policy-Managed"-Datenbank*
  Dafür werden Server-Pools mit einer minimalen und maximalen Anzahl von Servern und einer Priorität erstellt und die Datenbanken diesen Server-Pools zugeordnet. Die Anzahl der gestarteten Instanzen richtet sich dann danach, wie viele Server zur Verfügung stehen. Über die Priorität wird festgelegt, was passiert, wenn die minimale Anzahl von Servern in dem Pool unterschritten wird: Entweder werden Server aus einem anderen Pool übernommen (und dadurch der Pool verkleinert) oder nicht (wenn die Prioritäten aller anderen Pools größer ist).

Im Projekt stellte sich heraus, dass beide Methoden die Anforderungen bezüglich einer hochgradigen Automatisierung für ein Self-Service-Portal nicht abdecken. Bei Administrator-Managed-Datenbanken gibt es keine Möglichkeit, Instanzen woanders wieder zu starten, und bei Policy-Managed-Datenbanken beeinflusst man immer einen kompletten Server und kann die Instanzen beim Ausfall eines Servers nicht auf die anderen verteilen. Dies ist einer der Gründe, warum sich in diesem Projekt interessanterweise RAC One Node als optimale Lösung herauskristallisiert hat.

Oracle Data Guard

Wie bereits erwähnt, hat sich Data Guard recht schnell als die optimale DR-Lösung erwiesen. Active Data Guard scheidet zunächst aus, weil die benötigten Lizenzen nicht zur Verfügung stehen, es bleibt aber die Frage nach dem Aufbau eines Observers. In unserer Konfiguration haben wir es mit zwei Rechenzentren zu tun und daher ist die Frage, auf welcher Seite ein Observer installiert werden sollte. Optimalerweise müssten wir ein drittes Rechenzentrum verwenden, da der Observer in jedem Fall verfügbar sein sollte, egal ob die primäre oder die Standby-Seite ausfällt. In dem Projekt wurde auch deshalb auf den Observer verzichtet, weil man vor einem Schwenk auf die DR-Seite eine Überprüfung und ein Okay des Datenbankadministrators möchte. Die Entscheidung fiel dahingehend, den Observer nicht einzusetzen und damit auf den automatischen Switch im DR-Fall zu verzichten.

Der Einsatz des Data Guard Broker war jedoch obligatorisch und bietet sowohl für RAC als auch RAC One Node wesentliche Vorteile gegenüber einer manuellen Konfiguration mittels SQL. Generell muss man aber bei der Verwendung des Data Guard Broker Folgendes beachten:

- Die Konfiguration der Datenbank beziehungsweise der Datenbank-Parameter sollte nur noch über den Data Guard Broker erfolgen. Gerade die Parameter, die sich mit dem Log-Shipping (ARCHIVE_LOG_DEST_n etc.) beschäftigen, dürfen nur über den Broker geändert werden.
- Der Login an den Data Guard Broker darf nur mit expliziter Angabe von Benutzername und Passwort (beziehungsweise über den Einsatz von Identity Management wie eines Wallet oder Enterprise User/OVD) erfolgen, da ansonsten bei Switchover oder Failover der Connect an die Datenbank fehlschlägt. Besser ist es, auch den TNS-Alias mit anzugeben, da der Connect in der Regel als Benutzer mit der Rolle „sysdba" erfolgt und damit die lokale Eingabe eines Passworts ignoriert wird.
- Beim Switchover der Datenbank (Tausch der Rollen, Standby wird Primär und Primär wird Standby) ist darauf zu achten, dass die Standby-Datenbank nicht als Active Standby gestartet wird. Dabei ist vor allem wichtig, dass bei manuellen Fehlerkorrekturen die Datenbanken mittels GI gestartet beziehungsweise gestoppt werden. Die einfache Eingabe von „srvctl start database …" auf der neuen Standby-Datenbank führt dazu, dass diese „read only" geöffnet und somit die Active-Data-Guard-Option verwendet wird. Daher ist nach einem Schwenk auf die DR-Seite die Datenbank-Rolle entsprechend anzupassen.

Der erste Punkt lässt sich sicherlich einhalten, wenn den Administratoren ein entsprechendes Runbook zur Verfügung steht. Punkt zwei stellt momentan vor große Probleme, die im Hinblick auf die Automatisierung noch nicht gelöst sind. Grund dafür ist, dass natürlich das SYS-Passwort nicht abgelegt werden darf und dieses außerdem aufgrund der Security Policy regelmäßig geändert werden muss. Hier wird es wohl notwendig sein, eine Identity-Management-Lösung aufzubauen und die Passwort-Abfrage dorthin zu delegieren. Beim Ändern

Neu:
Oracle NoSQL Database 2.0

Die neue Version 2.0 von Oracle NoSQL Database ist eine hochskalierbare, Key-Value-Datenbank mit niedrigen Latenzen für die Bearbeitung von Big Data in Echtzeit. Mit der Vorstellung der neuen Version setzt Oracle die Innovation im Data-Management-Technologie-Portfolio fort. Das Produkt wurde ergänzt um effiziente Unterstützung für Storage und die Abfrage von großen Objekten wie Dokumenten und Bildern sowie dynamische Elastizität und automatisches Rebalancing für die Zuweisung von Storage und Rechenressourcen. Diese Erweiterungen entsprechen den sich verändernden Anforderungen an die Verarbeitung von Produktivdaten.

des SYS-Passworts muss außerdem darauf geachtet werden, dass nach der Änderung auf der primären Seite die „orapw<SID>“ auf alle Knoten verteilt werden muss.

Fazit

Oracle Data Guard ist sicherlich als Lösung für Disaster Recovery nicht zu schlagen, wenn man ein paar Details beachtet, wie zum Beispiel, dass eine Datenbank gern mal als Active Data Guard hochgefahren wird und man dadurch in Lizenzprobleme kommt. Interessant waren in diesem Projekt die Erkenntnisse bezüglich des Einsatzes von RAC One Node. Ursprünglich eher belächelt, nach dem Motto: „Wir haben RAC, wofür brauchen wir das?“, sind die Autoren letztendlich zu dem Schluss gekommen, dass RAC One Node in dieser Umgebung die bevorzugte Konfiguration sein wird.

Johannes Ahrends
johannes.ahrends@carajandb.com

Engelbert Wystrach
engelbert.wystrach@enwycon-it.de