Online Wartung und Updates mit Real Application Clusters

Björn Rost
portrix Systems GmbH
Hamburg

Schlüsselworte:

Real Application Clusters, Services, rolling upgrades, high availability, Hochverfügbarkeit

Einleitung

Real Application Clusters sind Oracle's Lösung für horizontale Skalierung und Hochverfügbarkeit. Neben der Vermeidung ungeplanter Ausfallzeiten bieten Real Application Clusters auch sehr gute Möglichkeiten, die Auswirkungen geplanter Wartungen zu verringern. Mit der richtigen Strategie und Einrichtung sind für Patching der Datenbank und des Betriebssystems sowie Arbeiten an der Hardware keine Beeinträchtigungen des Produktionsbetriebs nötig. DBAs und Unternehmen profitieren davon, dass weniger Wartungsfenster in der Nacht notwendig sind.
Dieser Vortrag erklärt die notwendigen Schritte, die bei der Konfiguration des Clusters und der Clients beachtet werden müssen und demonstriert das Vorgehen von Wartungsarbeiten. Dabei geht es um Services, Rolling Updates, Load Balancing, Transparent Application Failover, Connection Pooling sowie besondere Anforderungen an Applikationen.

Verfügbarkeit

Die Verfügbarkeit eines IT-Dienstes bestimmt sich durch das Verhältnis von Uptime zu Downtime und wird als Parameter in Service Level Agreements festgeschrieben. Auftraggeber fordern in der Regel eine Mindestverfügbarkeit, die man dann in eine gewisse Ausfallzeit pro Jahr herunterrechnen kann. Bei einer geforderten Verfügbarkeit von 99,99% etwa darf ein Service pro Jahr maximal 52 Minuten down sein. Wichtig ist, dass hier bei Auszeit zunächst nicht zwischen ungeplanten Ausfällen (Komponenten- , Stromausfall usw) und geplanter Arbeit wie Hardwareupgrades, Patches und ähnlichem unterschieden wird.

$$Verfügbarkeit = \left(1 - \frac{(Defekte + Wartung)}{Gesamtzeit}\right) * 100$$

Da es allerdings je nach Art des Dienstes gewisse Zeiträume geben kann, zu denen ein Ausfall keine oder nur geringe Auswirkungen hat, werden häufig geplante Arbeiten von dieser Berechnung ausgenommen wenn sie zu vereinbarten Zeitfenstern durchgeführt werden. Da interne Systeme meist nur während Arbeitszeiten benutzt werden, sind geplante Ausfälle nachts oder an Wochenenden akzeptabel. Internetsysteme können zwar rund um die Uhr genutzt werden, doch auch hier lassen sich Zeiträume mit weniger Aktivität – meist ebenfalls Nachtzeiten – nutzen. Durch die Globalisierung von Geschäften und die Verteilung auf verschiedene Zeitzonen sowie den Anspruch von Internetplattformen jederzeit erreichbar sein zu wollen, ist der Druck, die Downtime insgesamt zu reduzieren mit der Zeit immer weiter gewachsen. Gleichzeitig werden Hardwareausfälle immer

seltener weil Komponenten immer zuverlässiger werden und Redundanzen auf fast allen Ebenen erschwinglich oder im Rahmen der Anforderungen notwendig sind. In unserer Arbeit sind daher Downtimes aufgrund von Defekten auch bei zig Systemen eine absolute Seltenheit, während Wartungsarbeiten zu Software- und Betriebssystemupdates, Hardwareupgrades und Konfigurationsänderungen zum Tagesgeschäft gehören.
Weil diese Arbeiten aber im Konflikt zur Verfügbarkeit stehen, werden solche Routinearbeiten auf Nächte und Wochenenden geschoben, was einen personellen und damit ebenfalls finanziellen Mehraufwand bedeutet. Oft leidet auch die Qualität der Arbeit an diesen Bedingungen weil Administratoren in der Nacht nicht voll ausgeruht oder mit der besten Konzentration arbeiten können. Ebenso muss meistens alles alleine erledigt werden, auch wenn vier Augen viele Flüchtigkeitsfehler entdecken könnten.
Wegen des Aufwands und dem Versuch, Downtime zu vermeiden werden Wartungsarbeiten auch immer wieder aufgeschoben oder schlichtweg nicht gemacht. Gleichzeitig mit den Anforderungen an eine höhere Verfügbarkeit steigt aber auch der Anspruch an sicherere IT-Systeme. Das regelmäßige Einspielen von Sicherheitsupdates gehört dabei zu den grundlegenden Maßnahmen, die Administratoren machen können und müssen um sich vor Angriffen, Datenmissbrauch und -Diebstahl zu schützen.
Zum Glück werden aber auch die Werkzeuge, die uns zur Verfügung stehen immer besser, und so kann diesen Anforderungen mit etwas Planung gelassen begegnet werden. Bei RAC Clustern und einem entsprechend passend eingerichteten Applikationsstack können Wartungen so durchgeführt werden, dass immer genügend Datenbankinstanzen erreichbar sind um die Nutzung der Applikation zu gewährleisten. Oracle nennt dieses Verfahren *rolling upgrades*.
Die neue Option RAC One Node setzt genau hier an und bietet auch Kunden, die nicht mehrere Instanzen gleichzeitig einsetzen wollen die Möglichkeit, von rolling upgrades zu profitieren.

Rolling Upgrades

Um einen Knoten in einen Modus zu bringen, kann man eine einfache Liste mit wenigen Schritten abarbeiten. Ziel muss es dabei sein, laufende Transaktionen gar nicht oder nur so wenig wie möglich zu beeinträchtigen. Die allgemeinen Schritte lauten dabei so:

1. Umleiten neuer Connections auf einen bestehenden Knoten
2. Trennen der verbleibenden Verbindungen
3. Abschalten der Instanz, Durchführen von Arbeiten, Neustarts, usw
4. Neustart der Instanz und Verteilung von Connections auf alle Knoten

Die einzelnen Schritte und die Voraussetzungen werden in den nachfolgenden Absätzen erklärt.

Datenbankconnections und Services

Zunächst ist es wichtig zu verstehen, wie Connections bei einem RAC Cluster funktionieren und wie load-balancing eingerichtet, modifiziert und kontrolliert werden kann. Anwendungen bauen über das Netzwerk eine oder mehrere Verbindungen auf, über die dann jeweils eine oder hintereinander mehrere Transaktionen abgewickelt werden. Häufig überlässt man diese Logik dem Connection Pool eines Anwendungsservers. Aus Sicht des Programmierers bedient man sich Verbindungen aus diesem Pool, der wiederum vom Application Server verwaltet wird. Eine neue Connection kommt zustande, indem der Client bei einem Listener nach einer Verfügbaren Instanz für einen jeweils angegebenen Service fragt. Bei 10g war dies eine Listener-IP Adresse pro Instanz, die dann auch jeweils in den connection-Parametern, zum Beispiel der tnsnames.ora. Weil das mitunter viel Arbeit war, wurde mit 11g das Konzept der SCAN-Listener eingeführt. Hier muss in den Clients nur ein einzelner DNS-

Name hinterlegt werden, der dann auf meistens drei IP-Adressen auflöst, undzwar auch bei nur zwei oder mehr als drei Knoten. Dabei spielt es keine wirkliche Rolle, mit welchem Listener die erste Kommunikation stattfindet. Denn jeder Listener wird die endgültige Verbindung nach Load-Balancing Richtlinien an eine der vorhandenen Instanzen weiterleiten. So kommen auch Verbindungen zu Knoten zustande, die gar keinen SCAN-Listener haben. Steht eine Verbindung erstmal, bleibt sie auf diesem Knoten bis die Verbindung von einer der beiden Seiten getrennt wird. Zwei Dinge sind dabei besonders wichtig:
Die Anwendungen sollten regelmäßig ihre Verbindungen zur Datenbank trennen und neu aufbauen. Denn so haben die Listener eine Chance, die Verbindungen jeweils so zu verteilen, dass die Last zwischen den Knoten möglichst gleichmäßig geteilt wird. Man muss sich nur einen Cluster vorstellen, bei dem zum Start der Applikation nur ein Knoten verfügbar ist. In dem Fall würden alle Verbindungen zu diesem einen Knoten aufgebaut. Werden später weitere Knoten hinzugefügt, kann die Applikation diese Knoten nur nutzen, wenn alte Verbindungen getrennt und neue aufgebaut werden.
Die zweite wichtige Anforderung an Applikationen ist, dass Verbindungen nach einer Trennung durch den Datenbankserver automatisch neu aufgebaut werden. Denn nach dem Ausfall eines Knotens oder im Rahmen von geplanten Arbeiten wird es immer wieder vorkommen, dass Verbindungen getrennt werden. Dabei muss man noch unterscheiden, ob sich die Verbindungen gerade innerhalb oder außerhalb einer Transaktion befinden, denn abhängig davon kann es notwendig sein, eine weitere Fehlerbehandlung in der Anwendung vorzusehen.
Connection Pools erfüllen in der Regel beide Anforderungen, ohne dass ein Eingriff von Entwicklern oder Administratoren notwendig ist. Nur bei der Trennung von Verbindungen innerhalb von Transaktionen werden Exceptions an die Applikation weitergereicht, die dann ggfls den Vorgang erneut versuchen kann oder einen Fehler an den Anwender weitergibt.
An welche Instanz ein Listener eine Verbindung weiterreichen wird, hängt von den Einstellungen des jeweiligen Services ab. Der Servicename wird bei den Verbindungseinstellungen am Client angegeben und auf der Serverseite konfiguriert. Die Einstellung am Client sieht etwa so aus:

```
RAC_OLTP =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = scan.myrac.com)(PORT = 1521))
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = MyRACService)
))
```

Serverseitig lässt sich pro Service einstellen, welche Instanzen diesen Service bedienen sollen und nach welchen Parametern das load-balancing stattfindet. Bei der RAC-Installation wird auch immer ein default-Service eingerichtet, den man aber nur für die Administration und auf keinen Fall für Anwendungen benutzen soll. Denn Instanzen lassen sich aus diesem Service nicht entfernen und Instanzen melden diesen Service als verfügbar, auch wenn sie sich nicht im Status „open“ befinden. Man legt also mindestens einen neuen Service an und nutzt diesen für alle Connections von Clients. So ergibt sich dann die Möglichkeit, im laufenden Betrieb den Service so zu modifizieren, dass einzelne Instanzen diesen Service nicht mehr anbieten. Bestehende Verbindungen bleiben davon unberührt, aber neue Verbindungen werden nur noch an die verbleibenden Knoten geleitet. So kann mit etwas Vorlaufzeit die Wartung eines Knotens vorbereitet werden. Denn wenn die bestehenden Verbindungen ihre Arbeit beendet haben und neu aufgebaut werden, sind nach einiger Zeit keine Sessions mehr auf diesem Knoten aktiv. Mit diesem Befehl wird der Service RAC_OLTP, der vorher auf zwei nodes aktiv war, nur noch auf RAC2 laufen.

```
srvctl modify service -d RACDB -s RAC_OLTP -n -i RAC2
```

Wenn Applikationen und Service erst einmal korrekt eingerichtet sind, ist dies ein wichtiges Werkzeug für Administratoren, das so oft wie möglich geübt und angewandt werden sollte.

Instanzen herunterfahren

Die Parameter beim Stoppen einer Instanz sind dieselben wie bei nicht-RAC Systemen. Während man bei Single-Instanz-Systemen allerdings nur selten TRANSACTIONAL LOCAL oder gar NORMAL herunterfährt, kann dies bei Clustersystemen durchaus ein schlaues Mittel sein. Nachdem alle Services einer Instanz auf andere umgeleitet wurden, sorgt ein shutdown transactional local dafür, dass zunächst alle Transaktionen auf dieser Instanz beendet werden können, trennt aber auch sofort jede Connection, die die sich gerade nicht in einer Transaktion befindet. Das Resultat ist, dass die Instanz herunterfährt, ohne dass Endanwender davon beeinträchtigt werden weil neue Sessions und Connections zu den verbleibenden Knoten aufgebaut werden und bestehende Sessions noch ihre Arbeit beenden können.
Mit diesen beiden einfachen Tools können also Wartungen einzelner Knoten eingeleitet werden. Ist die Instanz erst heruntergefahren, können gefahrlos alle anderen Clusterdienste gestoppt werden.

Transparent Application Failover

Ganz ohne Eingriffe funktioniert also das Umverteilen von Sessions im Cluster nicht. Dabei gibt es aber eine Funktion von Oracle, deren Name Hoffnung darauf macht, dass eben doch alles ganz simpel ist. Transparent Application Failover, kurz TAF, soll dafür sorgen, dass laufende Transaktionen beim Ausfall eines Knotens für den Anwender völlig transparent auf einen verbleibenden Knoten wechseln. Leider klingt das allerdings wirklich zu gut um wahr zu sein. Denn die Einschränkungen schränken den sinnvollen Einsatz sehr stark ein. So funktioniert das Failover nur, solange in der Transaktion noch keine Daten durch DML modifiziert wurden. Also geht es nur bei Sessions, die entweder inaktiv sind oder bisher nur SELECTs ausgeführt wurden. In den allermeisten Systemen wird es aber so sein, dass doch recht viel DML gemacht wird, was dann verloren wäre und auch Queries bei einem Instanzausfall komplett erneut gestartet werden. TAF ist daher keine Funktion, auf die man sein komplettes Failover-Szenario aufbauen sollte sondern höchstens ein Werkzeug für einzelne Jobs, die lediglich Daten lesen und dabei so lange brauchen, dass ein kompletter Neustart des Jobs sehr aufwendig wäre.

Rolling Upgrades

Updates am Betriebssystem oder der Serverhardware lassen sich wie oben beschrieben einfach Knoten für Knoten durchführen. Unterschiedliche Versionen von Betriebssystempatches für kurze Zeit stellen in der Regel kein Problem dar. Ein Sonderfall sind allerdings Updates der Oracle Software selber. Denn hier ist es notwendig, dass mehrere Instanzen mit unterschiedlichen Versionen der RDBMS Software gleichzeitig laufen. Dies ist nicht immer möglich, wird aber vom Oracle Support immer öfter in Patches und Patchsets implementiert und getestet und ist mittlerweile bei der Mehrzahl der Patches machbar. Ob sich ein Patch für rolling upgrades eignet, steht meist schon im readme, kann aber auch einfach selber mit OPatch abgefragt werden.

```
[oracle@rac1 tmp]$ opatch query -is_rolling_patch 10352368
Invoking OPatch 11.1.0.6.6

Oracle Interim Patch Installer version 11.1.0.6.6
Copyright (c) 2009, Oracle Corporation.  All rights reserved.


Oracle Home       : /u01/app/oracle/product/11.2.0/db_1
Central Inventory : /u01/app/oraInventory
   from           : /etc/oraInst.loc
OPatch version    : 11.1.0.6.6
OUI version       : 11.2.0.1.0
OUI location      : /u01/app/oracle/product/11.2.0/db_1/oui
Log file location :
/u01/app/oracle/product/11.2.0/db_1/cfgtoollogs/opatch/opatch2011-
09-15_11-28-05AM.log


Patch history file:
/u01/app/oracle/11.2.0/db_1/cfgtoollogs/opatch/opatch_history.txt

--------------------------------------------------------
 Patch is a rolling patch: true
```

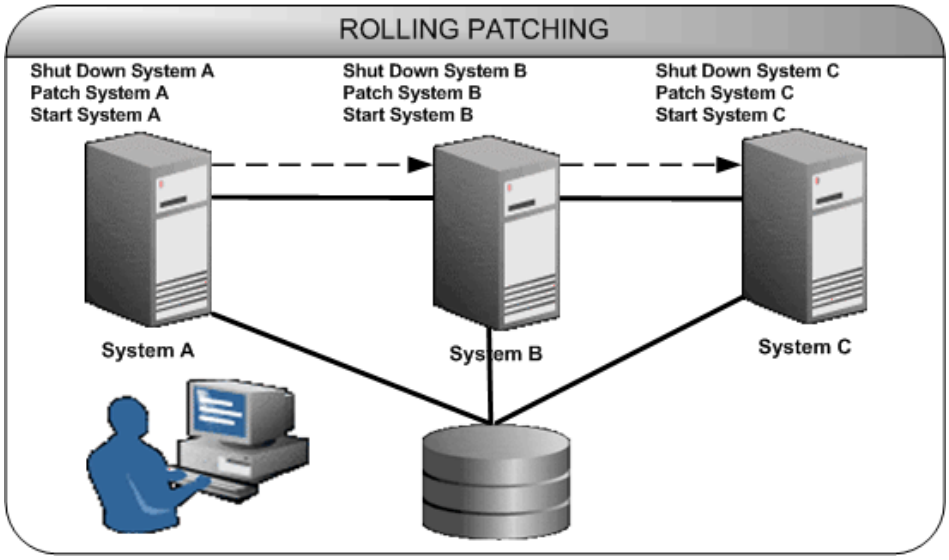

Abb. 1: rolling patching, Quelle: Oracle

Ist für ein Patch oder Patchset kein rolling upgrade möglich, gibt es noch die „minimum-downtime Methode“. Wenn der Patch nicht zulässt, dass gleichzeitig zwei unterschiedliche Softwareversionen aktiv sind, können zunächst einer oder mehrere Knoten heruntergefahren und gepatcht werden während die alte Version noch auf den verbleibenden Knoten läuft. Danach müssen allerdings kurz alle Knoten beendet werden bevor die zuerst aktualisierten Knoten wieder gestartet werden dürfen.

Fazit

Real Application Cluster und RAC One node bieten Administratoren einige Möglichkeiten, Updates und Wartungen ohne Betriebsunterbrechungen auszuführen. Dafür ist es nur notwendig, die Applikationen und Services entsprechend einzurichten und sich mit der Konfiguration von Services im Cluster und srvctl auseinanderzusetzen und deren Benutzung konsequent zu üben und in die tägliche Arbeit aufzunehmen.

Referenzen

Jeremy Schneider - Unleashing Oracle Services: A Comprehensive Review of “Services” in Oracle Databases

Kontaktadresse:

Björn Rost
portrix Systems GmbH
Friesenweg 2b
D-22763 Hamburg

Telefon: +49 (0) 40-398053-19
Fax: +49 (0) 40-398053-20
E-Mail b.rost@portrix-systems.de
Internet: www.portrix-systems.de