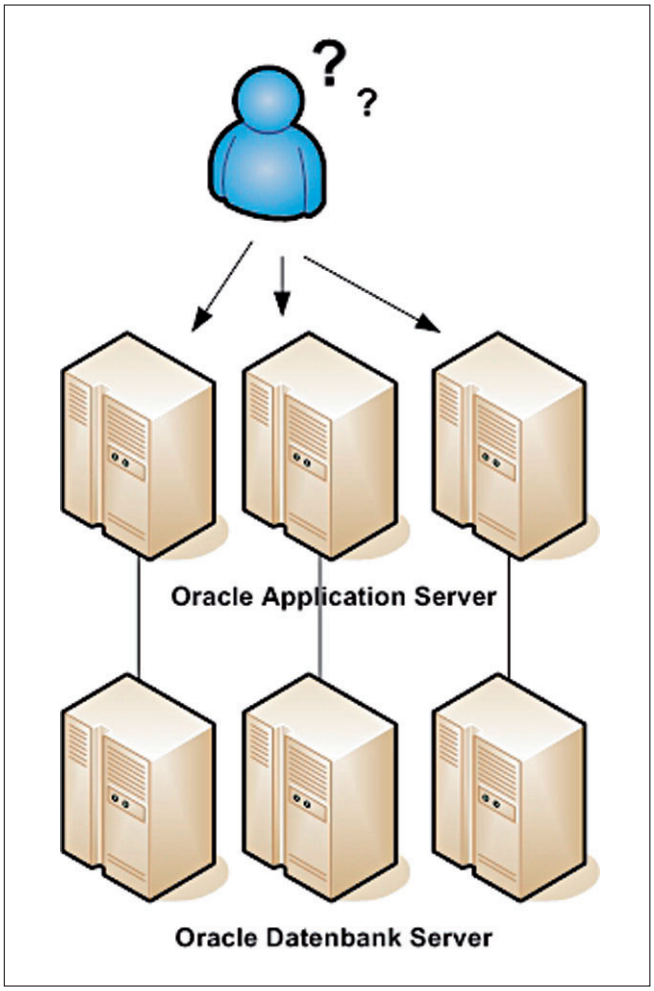
Open-Source-Technologie als Cluster für den Oracle Application Server

Autor: Björn Bröhl, Opitz Consulting Gummersbach GmbH

Es gibt verschiedene Konfigurationen, mit denen Oracle Application Server (OAS) hochverfügbar betrieben werden können. Die in der letzten Ausgabe aufgezeigten Architekturen basierten auf einem Failover-Cluster und Verwendung von Heartbeat. Diesmal werden Anwendungsfälle beschrieben, bei denen die auftretende Last auf mehrere Systeme verteilt werden soll.

Zur Lösung des Problems erfolgt der Einsatz mehrerer aktiver Server.



Zwischen den Anwendern und den Application Servern muss eine Instanz existieren, die entscheidet, auf welchen Application Server ein Anwender-Request weitergeleitet wird.

An dieser Stelle können verschiedene Systeme/Komponenten als Load-Balancer eingesetzt werden:

a) Handelt es sich bei der Kommunikation zwischen OAS und Client um reines http/https, lässt sich als Load-Balancer Oracle Webcache verwenden (siehe Metalink Note 207668.1)
b) Hat die Lastverteilung auch für andere Protokolle zu erfolgen, so muss ein Layer-4-Load-Balancer angewendet werden.

Dieser Artikel bezieht sich auf die Szenarien, bei denen ein Layer-4-Load-Balancer zum Einsatz kommt. Von verschiedenen Herstellern werden Load-Balancer angeboten, die in der Regel aus einem Layer-3-Switch bestehen, die durch weitere Funktionen wie etwa ein integriertes Server-System auf Linux-Basis zu Layer-4-Switchen/Load-Balancern erweitert werden. Diese Systeme zeichnen sich durch eine gute, meist Web-basierte Administration aus. Die Preisspanne liegt zwischen 4.000 € und 15.000 €, Anbieter sind beispielsweise F5, Cisco oder Coyote Point.

Wie auch im letzten Artikel soll hier der Aufbau eines Systems beschrieben werden, das sich durch möglichst geringe Kosten bei gleichem Funktionsumfang auszeichnen soll. Daher fällt auch hier die Wahl auf eine Open-Source-Technologie als Load-Balancer, dem Linux Virtual Server (LVS). Ähnlich wie Heartbeat ist auch LVS heute bei den gängigen Linux- und Enterprise-Linux-Distributionen im Kernel enthalten. Die im Folgenden beschriebenen Beispiele beziehen sich auf SuSE Enterprise Linux 8/9.

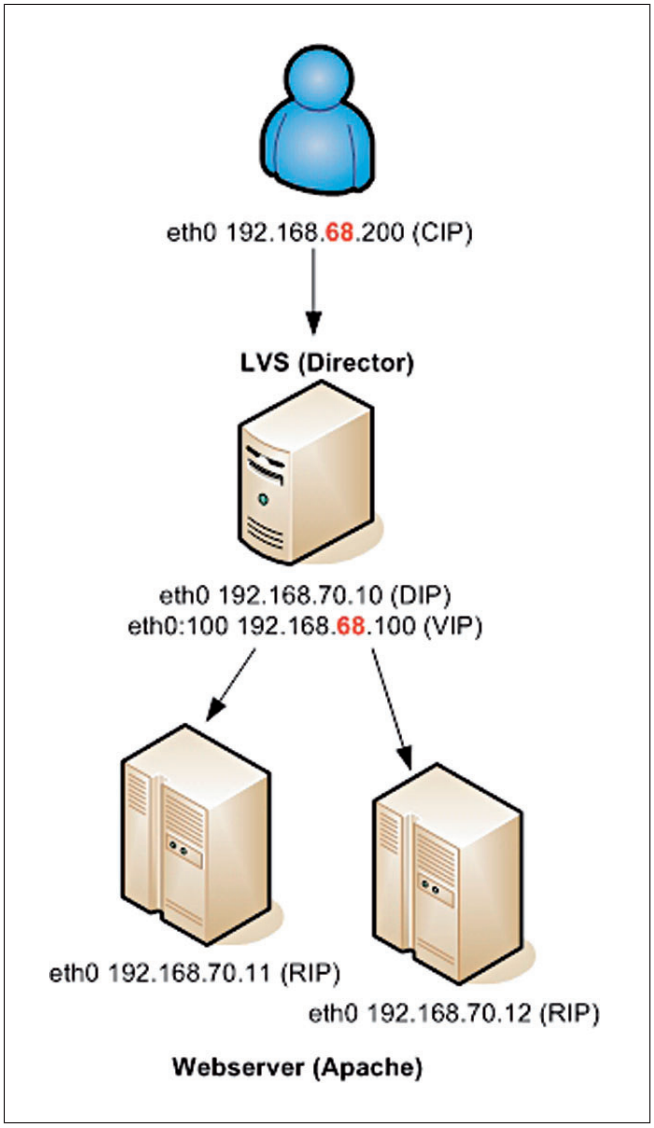
Im ersten Schritt ist die Grundkonfiguration des LVS-Servers (Director) beschrieben. Für den gesamten Aufbau werden vier Systeme benötigt:

- 1 x LVS (Director)
- 2 x OAS (Real Server)
- 1 x Client (zum Test)

Zum Ausprobieren der Funktionsweise können alle Systeme auch mit VMWare realisiert werden, wobei für den oben beschriebenen Aufbau ein ausreichend dimensioniertes Gast-System vorhanden sein muss. Wird anstelle des OAS ein einfacher Apache-Webserver verwendet, reicht ein Windows-System mit 2 GB RAM aus.

Auf dem Director-Server wird SuSE SLES 8 oder 9 installiert und auf den aktuellsten Stand gepatcht. Dabei sollte das Paket "IPVSADM" mit installiert werden, über das die spätere Administration erfolgt. Sollen die Sourcen von LVS verwendet werden, um diese von Hand zu kompilieren, so ist zu beachten, dass die IPVSADM-Version mit den verwendeten LVS-Sourcen identisch ist. Die Abkürzungen in Abbildung 2 bedeuten:

CIP = Client ip adress
DIP = Director ip adress
VIP = Virtual ip adress
RIP = Real server ip adress



Wie in Abbildung 2 dargestellt, befinden sich die Clients im Netzwerk auf 192.168.68.0/24 (zur Vereinfachung wird in diesem Beispiel NAT verwendet). Der Director muss über mindestens ein Netzwerk-Interface verfügen, das in diesem Beispiel auf 192.168.70.10/24 konfiguriert wird. Die Application Server verfügen ebenfalls jeweils über ein Netzwerk-Inferface, welche auf 192.168.70.11/24 und 192.168.70.12/24 eingestellt werden. Auf beiden Servern wird zu Testzwecken ein Apache-Webserver gestartet. Um später erkennen zu können, von welchem Server ein Request beantwortet wurde, sind die Index-Seiten entsprechend zu modifizieren.

Da LVS standardmäßig versucht, alle IP-Adressen in Namen aufzulösen, sollten alle Servernamen in der HOSTS-Datei des LVS-Servers eingetragen sein. In unserem Beispiel ist das:

```
127.0.0.1       localhost
192.168.70.11   host1.vmware.int host1
192.168.70.12   host2.vmware.int  host2
192.168.70.10   lvs.vmware.int lvs
192.168.68.100  vip1.vmware.int vip1
```

Nun kann die Konfiguration des LVS beginnen. Dazu muss zunächst einmal bei der Verwendung von NAT das IP-Forwarding aktiviert werden (echo "1" >/proc/sys/net/ipv4/ip_forward). Da ebenfalls zu Vereinfachung als Interface "eth0:100" verwendet werden soll, ist dafür zu sorgen, dass keine ICMP-Requests weitergeleitet werden. Dies erfolgt durch Setzen der folgenden Parameter:

```
echo "0"
>/proc/sys/net/ipv4/conf/all/send_redirects
cat
/proc/sys/net/ipv4/conf/all/send_redirects
echo "0"
>/proc/sys/net/ipv4/conf/default/send_
redirects
cat
/proc/sys/net/ipv4/conf/default/send_
redirects
echo "0"
>/proc/sys/net/ipv4/conf/eth0/send_
redirects
cat
/proc/sys/net/ipv4/conf/eth0/send_redirects
```

Anschließend kann das Interface "eth0:100" erstellt werden:

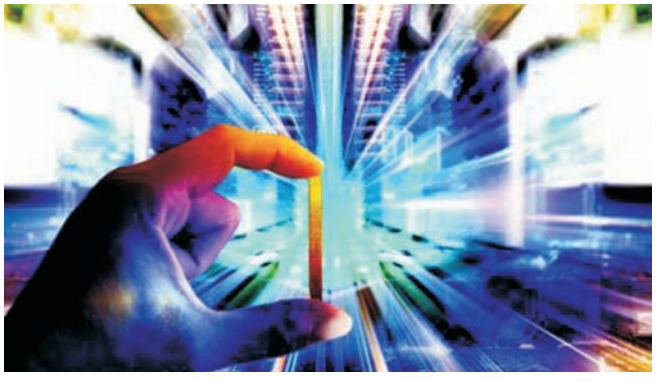
```
ifconfig eth0:100 192.168.68.100 broad-
cast 192.168.68.255 netmask
255.255.255.0
```

Nachdem dieses Interface erstellt wurde, sollte das VIP-Interface vom Client-System per Ping erreichbar sein. Ist dieser erste Schritt erfolgreich absolviert, kann man sich der Konfiguration des Loadbalancing widmen. Hierzu wird LVS mithilfe von "IPVSADM" parametrisiert.

Wir möchten unter der VIP-IP-Adresse "192.168.68.100" zwei Server via http erreichen. Dazu muss als Erstes diese VIP-IP-Adresse mit folgendem Kommando angelegt werden:

```
ipvsadm -A -t 192.168.68.100:80 -s rr
```

Dabei bedeutet der Parameter "-a", dass ein neuer virtueller Server auf der IP-Adresse "-t" 192.168.68.100 und Port 80 (an dieser Stelle könnte alternativ auch "http" verwendet werden) eingerichtet wird. Mit dem Parameter "-s" und der Option "rr" wird der Verteilungsalgorythmus ausgewählt, wie in diesem Falle rr = RoundRobin. Dabei werden die ankommenden Requests gleichmäßig zwischen den Servern verteilt. Im nächsten Schritt müssen die Application Server zu dem neuen Service hinzugefügt werden. Dies geschieht mit den Kommandos:



```
ipvsadm -a -t 192.168.68.100:80 -r
192.168.70.11:80 -m -w 1

und

ipvsadm -a -t 192.168.68.100:80 -r
192.168.70.11:80 -m -w 1
```

Der Parameter "-a" besagt, dass ein neuer Server dem Service "-t" mit der Adresse 192.168.68.100 auf Port 80 hinzugefügt wird. Der Application Server "-r" hat die IP-Adresse 192.168.70.11 bzw. 192.168.70.12 und hört ebenfalls jeweils auf den Port 80. Mit dem Parameter "-m" aktiviert man NAT (masquerading), und der Parameter "-w" für weight gibt die Gewichtung des Servers an. Mittels der Gewichtung kann erreicht werden, dass ein Server durch einen niedrigeren Wert gegenüber einem anderen bevorzugt wird. Sind diese Schritte ausgeführt, kann mit dem Kommando "ipvsadm" die Konfiguration angezeigt werden.

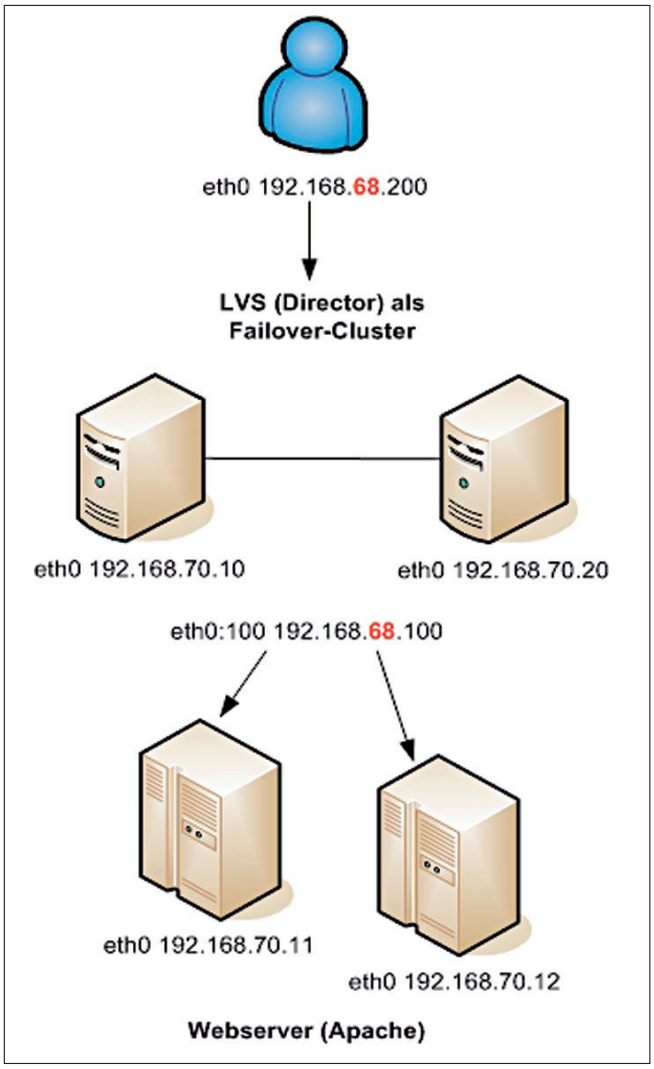
```
IP Virtual Server version 1.2.0 (size=4096)
Prot LocalAddress:Port Scheduler Flags
-> RemoteAddress:Port          Forward Weight
ActiveConn InActConn
TCP vip1.vmware.int:http rr
-> host2.vmware.int:http    Masq  1  0   0
-> host1.vmware.int:http    Masq  1  0   0
```

Auf den beiden Application Servern muss als Standard-Gateway die DIP-IP-Adresse (192.168.70.10) eingestellt werden. Um Probleme zu vermeiden, schaltet man hier das IP-Formwarding ab:

```
"echo "0" >/proc/sys/net/ipv4/ip_forward".
```

Sind alle Konfigurationsschritte erfolgreich durchgeführt, kann nun von einem Client (z.B. 192.168.68.200) aus über die Adresse http://192.168.68.100 auf die VIP-Adresse des Director und damit auf die beiden Application Server zugegriffen werden. Wurde zuvor auf beiden Servern eine unterschiedliche Index-Seite erstellt, ist durch mehrmaligen Refresh im Browser festzustellen, dass die Requests auf mehrere Server verteilt sind.

Natürlich ist das bisher beschriebene Szenario nur ein Beispiel für die Möglichkeiten, die LVS bietet. Für einen produktiven Einsatz sollte man LVS ebenfalls als Cluster betreiben, wie in Abbildung 3 dargestellt. Da LVS neben



http/https auch andere Protokolle unterstützt, lässt sich so eine hochverfügbare Konfiguration für die Oracle OAS-Infrastruktur erstellen. Beispiele für eine solche Konfigurationen (z.B. als "Rack mounted identity Management") finden sich im OTN unter: http:// www.oracle.com/technology/products/ias/hi_av/904_dif_ha_infra.html

Wie der vorherige Artikel soll dieses Beispiel zeigen, dass die Verbesserung der System-Verfügbarkeit auch mit geringem finanziellen und zeitlichen Aufwand erreicht werden kann. Weitere Informationen zu LVS finden Sie unter http://www.linuxvirtualserver.org

Kontakt:
Björn Bröhl
bjoern.broehl@opitz-consulting.de

Neu:

Weitere Informationen zu den Themen der DOAG *News* finden Sie unter
http://email.doag.org/

Oracle End to End Application Tracing und Analyse

Autor: Felix Castillo Sanchez, ORACLE Deutschland GmbH

Mit Einführung von 10*g* bietet Oracle die Möglichkeit, Datenbank-Verbindungen mehrschichtiger Anwendungen zu untersuchen. Dabei sind verschiedene Aspekte zu berücksichtigen, um akkurate und vollständige Daten zu erhalten.

Die Möglichkeiten des Tracings sind seit Einführung der "dbms_application_info"- und "dbms_monitor"-Packages in Version 10*g* stark erweitert worden. Statt eines Logon-Triggers oder eines allgemeinen Tracings können spezifische Teilbereiche untersucht werden. So lassen sich sowohl Sitzungen überwachen, die wegen eines Connection Poolings über mehrere Trace-Dateien verteilt sind, als auch sitzungsübergreifende Tracing-Informationen sammeln.

Client Identifier

Um Sitzungen über mehrere Trace-Dateien zurückzuverfolgen, kann mithilfe eines Client Identifiers ein Marker gesetzt werden, mit dem dann nach Beginn des Tracings selbst neu hinzukommende Sitzungen automatisch protokolliert werden. Der Client muss beim Starten lediglich den Client Identifier setzen:

```
begin;
   dbms_session.set_identifier('MyId');
end;
```