Das OVM-3.3-Kommandozeilen-Interface

Martin Bracher, Trivadis AG

Dieser Artikel stellt das Kommandozeilen-Interface von Oracle VM (OVM) Version 3.3.2 vor („ovmcli“), das einerseits das Konfigurieren der OVM-Umgebung und andererseits das Erstellen von virtuellen Maschinen (VM) bietet. Wer „ovmcli“ von OVM 3.2 bereits kennt, sollte trotzdem weiterlesen, denn in der Logik und Syntax hat sich zwischen den beiden Versionen einiges geändert.

Architektonisch ändert sich am Konzept von OVM Manager und Server nichts. Im Zentrum steht weiterhin der OVM Manager, über den man die OVM Server verwaltet. Statt über das Web-Interface erfolgt die Konfiguration nun über Kommandozeilen-Befehle in einer speziellen Shell *(siehe Abbildung 1)*. Diese läuft auf dem OVM Manager und man erreicht sie über „ssh“ (PuTTY unter Windows), Port 10000.

Der kleine Befehl „ssh -P 10000 admin@ovmmanager“ genügt bereits, um sich mit „ovmcli“ zu verbinden. Mit den Standard-Einstellungen wird man aber nach etwa zehn Minuten Inaktivität ausgeloggt. Es empfiehlt sich deshalb, einen kleinen Alias zu erstellen: „alias ovmcli='ssh -p 10000 –o ServerAliveInterval=40 admin@ovmmanager'“.

Statt sich jedes Mal mit dem entsprechenden Passwort anzumelden, kann man auch die SSH-Authentifizierung mit Public/Private Key verwenden. Diese ist insbesondere notwendig, wenn man später gewisse Sachen per Script automatisieren möchte. Dazu fügt man seinen SSH-Public-Key in die Datei „/home/oracle/ssh/ovmcli_authorized_keys“ ein. Man beachte: Wenn man sich längere Zeit nicht eingeloggt hat, verlangt „ovmcli“ trotz Key nach dem Passwort.

Die Kommandos sind jeweils einzeilig einzugeben. Falls man das wie in nachfolgendem Beispiel als Script macht,

```
ssh -p 10000 –admin@ovmmanager <<EOD
list network ;
list vm ;
EOD
```

Listing 1

```
OVM> create network ?
                     *name
                     description
                     roles
                     on
OVM> create vm name=myVM domainType=?
      XEN_HVM,XEN_HVM_PV_DRIVERS,XEN_PVM,LDOMS_PVM,UNKNOWN
```

Listing 2

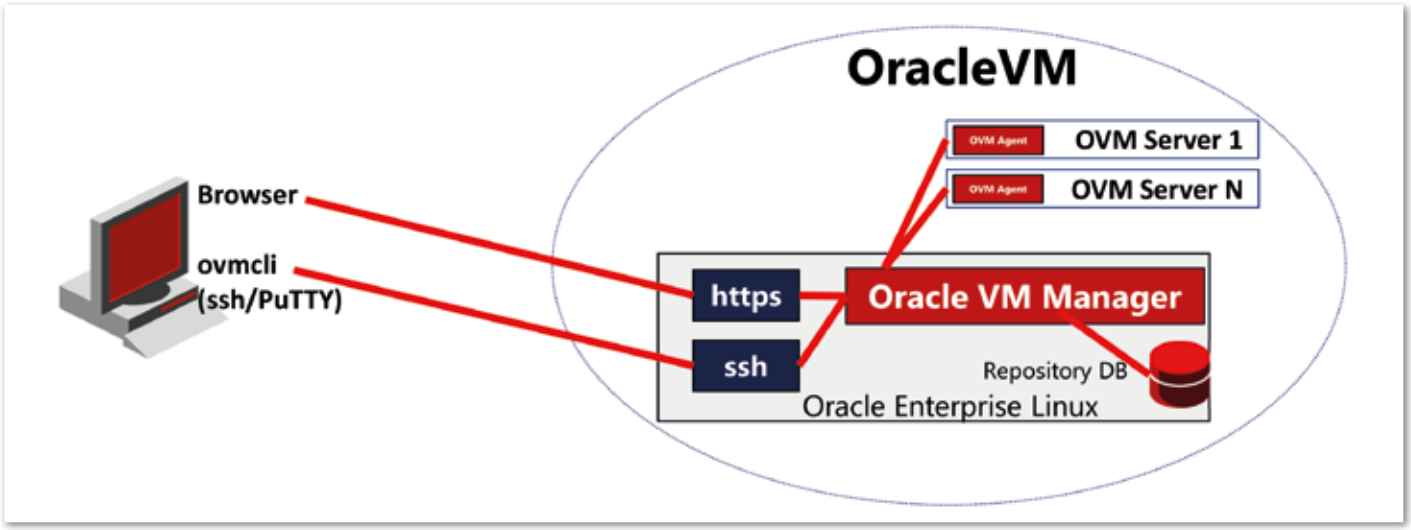

Abbildung 1: Die OVM-Architektur

müssen die Befehle mit Semikolon abgeschlossen sein, ansonsten ohne *(siehe Listing 1)*.

Die Syntax ist einfach verständlich und im „Oracle VM Manager Command Line Interface User's Guide for Release 3.3" gut dokumentiert. „ovmcli" selbst hat eine Hilfe eingebaut: „help" gibt eine Übersicht über die Befehle. Bei jedem Befehl kann man sich die weiteren Optionen mit „?" anzeigen lassen *(siehe Listing 2)*.

Setup der OVM-Infrastruktur

Nachdem wir den OVM Manager und auch die OVM Server per CD oder ISO-Image installiert haben, sind wir bereit, die Konfiguration der Umgebung über „ovmcli" zu erledigen. Die Konfiguration per „ovmcli" hat einige wesentliche Vorteile gegenüber dem Web-Interface: Einerseits ist die Konfiguration leicht zu dokumentieren und einfach nachvollziehbar, andererseits ist sie leicht wiederverwendbar zum Aufsetzen weiterer Umgebungen.

Nach der Installation kennt der OVM Manager noch keine OVM Server (die nach der Installation nur eine minimale Konfiguration haben). Diese müssen zuerst per „discoverServer ipAddress=172.16.98.35 password=<pwd> takeOwnership=yes" inventarisiert werden. Hier fällt schon der erste Syntax-Unterschied zu 3.2 auf: Die Option „username=oracle" entfällt nun endlich, der einzig gültige Wert war ohnehin nur „oracle".

Die Inventarisierung dauert einen gewissen Moment. In dieser Zeit werden die Hardware analysiert und die Komponenten im OVM Manager registriert. In den weiteren Schritten können wir dann diese Komponenten konfigurieren. Zuerst brauchen wir SAN-Storage, um einen Server-Pool definieren zu können. Wir suchen die 12-GB-LUN für das Pool-Filesystem über „list physicaldisk" und „show physicaldisk name="<name>"". Der erste Befehl zeigt alle vorhandenen LUNs an, der zweite unter anderem deren Größe. Diesen Schritt erledigt man jedoch meist im Web-GUI, da hier die Namen und Größen auf einen Blick ersichtlich sind. Wenn nun die 12-GB-LUN für das Pool-Filesystem gefunden ist, lässt sich der Server-Pool definieren *(siehe Listing 3)*.

Zu einem Server-Pool gehört mindestens ein Server, der mit „add server name=trovms01 to serverpool name=trainingpool01" hinzugefügt wird. Als Nächstes wird das Storage Repository erstellt, auf dem die virtuellen Maschinen gespeichert sind. Wiederum sucht man mit „list physicaldisk" und „show physicaldisk" (oder via Web-GUI) eine passende freie Disk. Mit dieser wird dann das Repository erstellt. *Listing 4* zeigt den Befehl für OVM 3.2. Bei OVM 3.3 hat sich die Syntax wieder signifikant geändert. Hier wird zuerst ein Filesystem erzeugt und auf dieses wird dann das Repository aufgesetzt *(siehe Listing 5)*.

An welche Server dieses Repository präsentiert wird, bestimmt der Befehl „add Server name=trovms01 to repository name=tr_repo01". Dieser ist für jeden neuen Server zu wiederholen. Dann ist es an der Zeit, sich der Netzwerk-Konfiguration zu widmen. Was im Bereich „Netzwerk" alles erkannt wurde, lässt sich mit wenigen Befehlen anzeigen *(siehe Listing 6)*.

Der schon automatisch eingerichtete, aber nur mit einer Netzwerk-Karte konfigurierte „Bondport 0" lässt sich nun um eine zusätzliche Netzwerk-Karte erweitern *(siehe Listing 7)*. Dieser automatisch erstellte Bondport gehört bereits zum Netzwerk „172.16.98.0". Man kann nun auch definieren, welche Rollen dieses Netzwerk haben soll *(siehe Listing 8)*. Auf diesem Interface lassen sich nun zusätzlich auch VLANs einrichten *(siehe Listing 9)*.

Über „ovmcli" lässt sich natürlich auch Network Time Protocol (NTP) konfigu-

```
create serverpool virtualIP=172.16.98.36
clusterEnable=yes physicaldisk="3PARdata (3)"
migrateUsingSsl=yes name=trainingpool01
```

Listing 3

```
create Repository name=tr_repo01
 PhysicalDisk=0004fb0000180000c9a1d731dd955569
serverPool= trainingpool01
```

Listing 4

```
list fileserver
create filesystem physicaldisk="3PARdata (1)" name=tr_repofs01 on
fileserver name="Local FS trovms01.trivadistraining.com"
create repository name=tr_repo01 on filesystem name=tr_repofs01
```

Listing 5

```
list network
list port
list bondport
```

Listing 6

```
add port id=0004fb0000200000489d9d4c5243d394
 to bondport id=0004fb0000200000ffb29078190405e5
```

Listing 7

```
edit network name=172.16.98.0 roles='MANAGEMENT,LIVE_
MIGRATE,CLUSTER_HEARTBEAT,VIRTUAL_MACHINE,STORAGE'
```

Listing 8

```
create vlaninterface vlanid=801 name=HACluster_trovms01 on bondport name="bond0 on trovms01.trivadistraining.
com"
create network roles="VIRTUAL_MACHINE" name=tr_public
add vlaninterface name=HACluster_trovms01 to network name=tr_public
```

Listing 9

rieren. Auch hier hat sich die Syntax zwischen den Versionen völlig geändert. Bei OVM 3.2 heißt es „setNTP list=193.5.60.8“ und „syncNtp“. Bei OVM 3.3 muss man NTP für jeden Server einzeln definieren, dafür könnte man auch für jeden Server andere Einstellungen wählen, auch wenn es dafür keinen vernünftigen Grund gibt: „edit server name=trovms01 ntpservers=“swisstime.ethz.ch““. Zusammengefasst kann man sagen, dass sich das Setup eigentlich auf die folgenden wenigen Befehle reduziert *(siehe Listing 10)* plus ein paar Befehle zur Netzwerk-Konfiguration, abhängig von der vorhandenen Netzwerk-Hardware und den Anforderungen (Bonding, VLAN).

Diese Syntax eignet sich also hervorragend, um das Setup zu dokumentieren, und als Vorlage für weitere Umgebungen. Ein vollständig automatisiertes Setup ist jedoch nicht so einfach möglich. Insbesondere die Konfiguration des Storage (Pool-Filesystem, Storage Repository) erfordert die Kenntnis der ID oder des Namens der zu verwendenden LUN, wie sie OVM beim „discoverServer“ vergeben hat. Um dies auch zu automatisieren, ist Script-Programmierung erforderlich, beispielsweise mit „expect“-Scripts.

Erstellen virtueller Maschinen

Nun steht der Server zur Verfügung und man kann damit beginnen, virtuelle Maschinen (VMs) zu erstellen. Eine VM ist eine Definition der virtuellen Hardware (CPU, Memory etc.) sowie der Disks, die ein File in einem Storage Repository oder eine physische LUN sein können. Zum Schluss werden diese Disks in die VM eingebaut. Unschwer zu erraten, lässt sich eine VM durch den Befehl „create vm“ erzeugen *(siehe Listing 11)*. Dieser erstellt eine leere, paravirtualisierte VM, ohne Netzwerk-Karten und ohne Storage.

Im nächsten Schritt lassen sich nun Netzwerk-Karten hinzufügen und mit dem passenden Netzwerk verbinden. Auch hier haben sich Logik und Syntax zur Version 3.2 geändert. Bei OVM 3.2 generierte man sich vorgängig eine oder einen ganzen Pool („vnicCreate“) von Netzwerkkarten und konnte diese dann der VM zuweisen *(siehe Listing 12)*. Neu bei OVM 3.3 erstellt und verbindet man die Netzwerk-Karte in einem Schritt *(siehe Listing 13)*.

Im Prinzip kann man irgendeine eindeutige MAC-Adresse wählen. Nach Erfahrung des Autors funktioniert es allerdings nur mit dem Prefix „00:21:f6“, ansonsten kommt beim Start die Meldung „Error: Device 0 (vif) could not be connected“. Falls einen die MAC-Adresse nicht interessiert, kann man sie auch weglassen; dann wird irgendeine nicht verwendete Adresse zugewiesen.

Als Nächstes sind Disks an die VM zu präsentieren. In diesem Beispiel kommt eine zuvor geklonte Disk („cloneVdToRepo“) eines Templates vor, also eine Disk mit einem bereits installierten Betriebssystem, das sich beim ersten Start konfigurieren lässt. Die Disks steckt man in einen Slot. Zu beachten: Bei Hardware-Virtualisierung („XEN_HVM“) stehen nur fünf Slots zur Verfügung. Erst paravirtualisiert oder zumindest mit paravirtualisierten Treibern steht das Zehnfache an Slots zur Verfügung. Die Disk in Slot 0 wird paravirtualisiert als „/dev/xvda“, die in Slot 1 als „/dev/xvdb“ präsentiert. Der Befehl lautet „create VmDiskMapping slot=0 VirtualDisk=vm201_xvda-tr1 name=vm201_slot0 on Vm

```
discoverServer
create serverpool
add server to serverpool
create filesystem
create repository
add Server to repository
```

Listing 10

```
create Vm name=vm201  repository=tr_repo01 domainType=XEN_PVM
osType='Oracle Linux 6' bootOrder=DISK cpuCount=4 highAvailability=no
memory=4096 memoryLimit=8192 hugePagesEnabled=yes on ServerPool
name=trainingpool01
```

Listing 11

```
create vnic name=00:21:f6:01:00:f8 network= tr_public
add vnic name=00:21:f6:01:00:f8 to vm name= vm201
```

Listing 12

```
create vnic macAddress=00:21:f6:01:00:f8 name=00:21:f6:01:00:c7
network=tr_public on vm name=vm201 ;
```

Listing 13

```
list PhysicalDisk
create VmDiskMapping slot=1 PhysicalDisk=…
```

Listing 14

```
sendvmmessage vm name=vm201 key=com.oracle.linux.network.hostname message=vm201.trivadistraining.com log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.network.device.0 message=eth0 log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.network.onboot.0 message=yes log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.network.bootproto.0 message=static log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.network.ipaddr.0 message=172.16.97.211 log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.selinux.mode message=disabled log=no;
sendvmmessage vm name=vm201 key=com.oracle.linux.root-password message=manager log=no;
```

Listing 15

```
vm=vm201
device=xvda
repo=repo01
ssh -P 10000 admin@ovmmanager
<<EOD
edit VirtualDisk
name=${vm}_${device}_${repo}
shareable=yes;
EOD
```

Listing 16

name=vm201;". Schon ist eine virtuelle Maschine erstellt. Bei Bedarf lassen sich noch weitere leere Disks erstellen („create VirtualDisk") und hinzufügen. Für Oracle-Datenbanken empfiehlt sich jedoch aus Gründen der Performance, physische LUNs direkt in die VM zu mappen, statt virtuelle Disks (Files auf dem Storage-Repository) einzusetzen *(siehe Listing 14)*.

In vielen Fällen ist noch Hard Partitioning erforderlich, also die explizite Limitierung von VMs auf bestimmte CPU-Kerne. Dabei werden Oracle-Lizenzen nur noch für die effektiv zugewiesenen Cores benötigt (siehe Metalink-ID 466538.1). Leider kann man dies bis heute weder über das Web-GUI noch per „ovmcli" erledigen. Die einzige offizielle Möglichkeit ist, dies über das „ovm_vmcontrol"-Programm aus den OracleVM3 Utilities (Patch 13602094) zu erledigen. Diese installiert man am besten gleich auf dem OVM Manager. Für Version 3.3 muss zuerst noch beim OVM Manager „tcps" auf „Port 54322" aktiviert sein. Wie das geschieht, steht leider nicht in diesem Utility-Paket. In „/u01/app/oracle/ovm-manager-3/bin" muss das „secureOvmmTcpGenKeyStore.sh"-Script aufgerufen werden und danach ist der OVM Manager zu restarten. Nun lässt sich das Hard Partitioning über „ovm_vmcontrol -u admin -p pwd -h localhost -vm vm201 -c vcpuset -s 0-3" konfigurieren.

Jetzt kann man die VM über „start vm name=vm201" starten. Falls nur leere Disks zugewiesen wurden, lässt sich nun das Betriebssystem installieren. In unserem Beispiel haben wir die Root-Disk eines Templates zugewiesen und können es nun konfigurieren, da im Template die „Oracle VM Guest Additions" installiert und konfiguriert sind. Dies geschieht entweder interaktiv in der Konsole (diese lässt sich nicht über „ovmcli" starten, sondern nur über das Web-GUI) oder die benötigten Angaben lassen sich via OVM-Messaging-System übergeben *(siehe Listing 15)*. Wichtig ist, dass „com.oracle.linux.root-password" als letzte Message übermittelt wird. Das Template geht nun davon aus, dass keine weiteren Messages mehr übermittelt werden. Detailliertere Informationen zur Erstellung von VMs mit Templates stehen im Vortrag des Autors „Oracle VM3: Virtuelle Maschinen per Script erstellen" von der DOAG 2014 Konferenz *(siehe http://www.slideshare.net/trivadis/doag2014-vms-percli")*.

Im Gegensatz zur Konfiguration der OVM Server lässt sich die Erstellung von VMs sehr gut automatisieren. Das Skript zur Erstellung ist immer gleich, die individuellen Werte übergibt man per Variable. Ein passendes Namenskonzept, das eindeutige Objektnamen gewährleistet, erleichtert dabei die Arbeit. Beispielsweise kann man die Disks nach VM-Name, Device-Name und Repository-Name benennen „vm201_xvda_repo01" *(siehe Listing 16)*.

Fazit

Dieser Artikel stellte das Kommandozeilen-Interface von Oracle VM (OVM) Version 3.3.2 vor („ovmcli"), das einerseits das Konfigurieren der OVM-Umgebung und andererseits das Erstellen von virtuellen Maschinen (VM) bietet. Wer „ovmcli" von OVM 3.2 bereits kannte, hat interessante Neuigkeiten in der Logik und Syntax erfahren.

Martin Bracher
martin.bracher@trivadis.com