RESTful Automation von Oracle VM

Nico Henglmüller
Sphinx IT Consulting
Aspernbrückengasse 2, Wien

Schlüsselworte
Oracle VM, REST, Automatisierung

Einleitung
Dieser Vortrag geht zu Beginn auf die Motivation und den Nutzen der Webservice API ein. Anschließend werden die Grundzüge der Architektur des Oracle VM Managers erläutert und die zentrale Speicherung von Meta- und Steuerungsdaten im Oracle VM Manager eingegangen. Den Kern bildet eine praktische Beispielimplementierung eines automatisierten Backupplans, auf Basis der vorgestellten Konzepte. Abgerundet wird die Präsentation durch ein Fazit und einer Diskussion der Ergebnisse.

Motivation
Um Arbeitsschritte unter Oracle VM zu automatisieren stand bis zum Release 3.2 nur das Command-line Interface in Kombination mit Expect-Skripten zu Verfügung. Das bedeutet, dass innerhalb von Skripten auf Eingabeaufforderungen seitens der CLI reagiert wurde. Weicht die Ausgabe des CLI ab, schlägt das Expect-Skript fehl und eine Überarbeitung desselben ist erforderlich. Beispielsweise verwendet das aktuelle Backup Skript von Simon Coter, Product Manager Oracle VM, den folgenden Expect-Skript Auszug:

```
[...]
##ssh Login auf dem Oracle VM Manager mit user/pw
spawn ssh $loginUser@$mgmtServerAddress
expect_after eof {exit 0}

set timeout 10

##den Key Fingerprint akzeptieren
expect "yes/no" {send "yes\r"}
##Passwort Eingabe und Bestätigung mit Enter
expect "password:" {send "$loginPassword\r"}
[...]
```

Bereits in diesem kurzen Codeauszug, welcher für den SSH Verbindungsaufbau zu dem Oracle VM Manager verantwortlich ist, existieren mehrere Fehlerquellen. Ist beispielsweise der Fingerprint des Oracle VM Managers auf dem lokalen System bereits in der known_hosts Datei vorhanden, läuft dieses Skript endlos. Dies liegt daran, dass auf die Frage, ob der öffentliche Schlüssel mit dem des Servers übereinstimmt (`"yes/no"`) gewartet wird. Diese Abfrage stellt SSH allerdings nicht, wenn sich der öffentliche SSH Schlüssel bereits auf dem lokalen System befindet.

Sieht man über das Expect-Skript und dessen Mängel hinweg, bleibt das steuernde Bash-Skript, der Kern des Backupalgorithmus. Dieses ist teilweise mit Kommentaren versehen und benutzt das Command-line Interface, um VMs zu sichern. Durch die Grenzen der Skriptsprache Bash und der eingeschränkt vorhandenen Dokumentation ist dieser Code und sein Aufbau schwer lesbar. Immer mehr Funktionen wurden hinzugefügt und mehrmals wurde das Skript angepasst. Paradigmen aus modernen höheren Programmiersprachen sind innerhalb der Skriptsprache nicht verfügbar und schnell fühlt man sich beim Lesen des Skripts an „Spaghetticode“ erinnert. Darüber hinaus kann die vorhanden Infrastruktur Anpassungen an dem Skript erforderlich machen oder interne Policies den Zugriff über SSH aus einem Expect-Skript unterbinden. Allerdings bietet dieses Skript erste Anhaltspunkte für die Funktionsweise des Oracle VM Webservice APIs, da hinter dem Command-Line Interface eine ähnliche Codebasis steckt.

Architektur

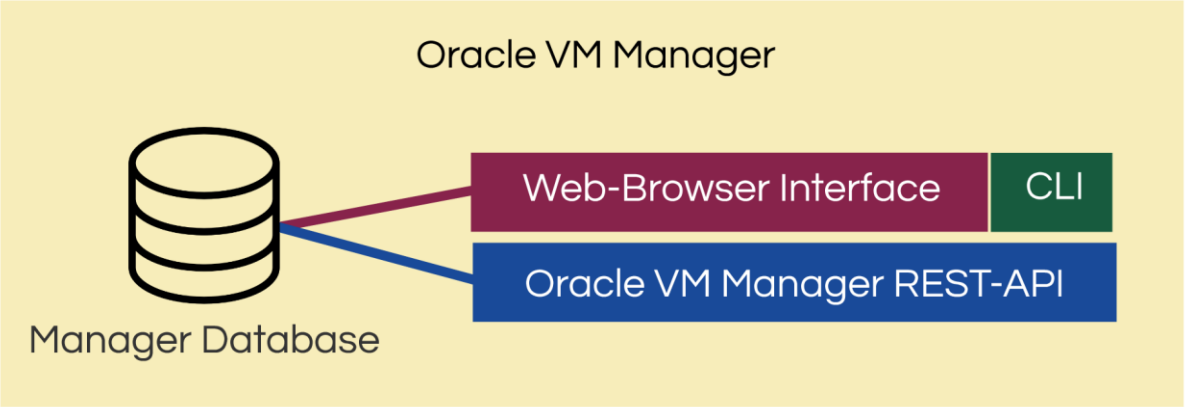

Abb. 1: Architektur des Oracle VM Managers

Der Oracle VM Manager besteht aus vier elementaren Teilen:

1. Die Manager Datenbank:
   Zur dauerhaften Speicherung der anfallenden Daten verwendet der Manager eine MySQL Enterprise Datenbank. Die Weblogic Applikation des Managers verwendet einen Objekt-relationalen Mapper, welcher die Java Objekte und die Verwaltung der Daten in der Datenbank verantwortet. Entsprechend handelt es sich nicht um ein klassisches relationales Datenmodell, sondern um einen Objektorientierten Ansatz. Aus diesem Grund ist die Vererbungshierarchie der Objekte von elementarer Bedeutung, da sich in vielen Superklassen nützliche Attribute zur Speicherung von Metainformationen verbergen.
2. Das Webbrowser Interface:
   Dieses Interface wird benutzt um virtuelle Maschinen, Storage, Netzwerkressourcen, usw. zu verwalten. Es arbeitet asynchron, das bedeutet, dass sich die Informationen innerhalb der Datenbank und der Manager Oberfläche zum gleichen Zeitpunkt unterscheiden können. Das Browser Interface gibt gute Anhaltspunkte, wie spezielle Workflows funktionieren und hilft bei der Erstellung von automatisierten Arbeitsschritten.
3. Das Command-line Interface:
   Das Command-line Interface ist über SSH auf Port 10000 auf dem Oracle VM Manager Host erreichbar und benutzt die Anmeldedaten des Oracle VM Managers. Das CLI benutzt eine eigenwillige Syntax und ist durch mehrere Mäkel (fehlende Autocompletion, keine Historisierung der abgesetzten Kommandos, usw.) sehr gewöhnungsbedürftig. Allerdings handelte es sich bei der CLI um das erste Werkzeug zur Automatisierung von Oracle VM, welches durch die Webservice API abgelöst wurde.

4. Oracle VM Manager REST-API:
   Das REST API bietet mit XML oder JSON zwei Dialekte bei der Kommunikation mit dem Webservice. Eine detaillierte und absolut notwendige Dokumentation der Schnittstelle kommt mit dem Installationsimage des Oracle VM Managers. Prinzipiell ist die Kommunikation mit dem API durch HTTPS verschlüsselt und verwendet für den Webservice bei einer frischen Installation ein selbst erstelltes Serverzertifikat. In produktiven Umgebung ist die Verwendung von der jeweiligen verwaltenden CA verwalteten Zertifikaten empfohlen.

Wegen der *deprecation* des SOAP API mit dem Oracle VM Manager Release 3.4 und weil dieses in zukünftigen Versionen entfernt wird, wurde darauf verzichtet dieses hier zu beschreiben.

Authentifizierung

Da der Zugriff auf das API im Gegensatz zur CLI nicht über SSH, sondern über HTTPS erfolgt, existieren zwei Möglichkeiten zur Authentifizierung. Die erste Option ist den Benutzernamen und das Passwort bei Requests im HTTP Header als Authentication Attribut zu übergeben. Eine Alternative hierzu ist die Authentifizierung durch ein Clientzertifikat, welches mit Hilfe des Webservice durch die CA auf dem Manager signiert wird. Hierbei ist darauf zu achten, dass die für die HTTP Anfragen verwendete Library verschlüsselte private Schlüssel unterstützt oder die privaten Schlüssel als Klartext auf dem Server abgelegt werden müssen. Als Beispiel sei hier die Python Bibliothek `requests` angeführt, welche diese Funktionalität nicht enthält.

Steuerungs- & Metainformationen

Wie bereits erwähnt organisiert Oracle VM Stammdaten in Form von Java Objekten und speichert diese mit Hilfe eines Objekt-relationalen Mappers in die MySQL Datenbank. Beispielsweise verfügen alle Objekte, welche von der Superklasse baseObject abgeleitet werden, über ein Attribut, welches benutzerdefinierte Metadaten (`userData`) zu dem Objekt speichert. Dieses Array ermöglicht es Metainformationen beispielsweise zu einer virtuellen Maschine oder einer virtuellen Disk zu speichern. Die dort abgelegten Informationen werden von der Oracle VM Manager Applikation ignoriert und dienen ausschließlich der Speicherung von benutzerdefinierten Informationen. Ein Anwendungsfall wäre die Speicherung von Backup Zeitplänen, -Zyklen und -Retentionpolicies für virtuelle Maschinen. All diese Informationen werden dem Oracle VM Manager Objekt (VM, vDisk, Network, etc.) als JSON Array hinzugefügt und die Notwendigkeit eines Metadaten Repositories außerhalb des Oracle VM Managers entfällt.

Backup Scheduling

Der Oracle VM Manager bringt mit der Installation einen Scheduler in Form des „Recurring“-Tabs innerhalb der Weboberfläche mit. Allerdings ist dieser nicht für benutzerdefinierte Jobs freigegeben und enthält ausschließlich default Jobs. Dies macht einen externen Service wie beispielsweise den cron daemon notwendig, um die Backup Jobs regelmäßig anzustoßen. Eine Alternative ist die Verwendung von Frameworks oder Bibliotheken, welche diese Funktionalität mitbringen. Unter Python bietet diese Funktionalität beispielsweise das Modul `python-crontab`.

Fazit

Mit dem Command-line Interfaces schaffte Oracle die Basis zur Automatisierung von Oracle VM. Der logische nächste Schritt war es diese Funktionalität in Form eines Webservice bereit zu stellen. Dadurch eröffnen sich ungeahnte Möglichkeiten unabhängig von dem Command-line Interface, dessen eigenwilliger Syntax und dem oftmals damit verbundenem Urwald an Skripten. Mit Version 3.3 vollzog Oracle diesen Schritt in Form einer standardisierten und gut dokumentierten XML- und JSON-basierten Schnittstelle. Waren bisher mühsame und unwartbare Expect-Skripte notwendig, um Arbeitsschritte des Multipurpose Hypervisors zu automatisieren, schafft das API hier Abhilfe. In Bezug auf das praktische Beispiel wird das Scheduling Feature, welches das Scheduling von benutzerdefinierten Jobs ermöglicht, schmerzlich vermisst.

Referenzen

Oracle VM Web Services API Developer's Guide for Release 3.4
Oracle VM Concepts Guide for Release 3.4
OvmSDK_3.4.4.1709: befindet sich auf dem Oracle VM Manager 3.4.4 Installationsimage
Simon Coters Backup Skript: https://blogs.oracle.com/scoter/oracle_vm_3_3_31
Python requests Library: http://docs.python-requests.org/en/master/user/advanced/

Kontaktadresse:

Nico Henglmüller
Sphinx IT Consulting GmbH
Aspernbrückengasse 2/6, Wien
AT-1020 Wien

Telefon:	+49 (0) 1-599 31-0
Fax:	+49 (0) 1-599 31-99
E-Mail	nhenglmueller@sphinx.at
Internet:	www.sphinx.at