Oracle WebLogic unter Docker/DCOS/Mesos

Dr. Páll Eggerz
virtual7 GmbH
Karlsruhe

Schlüsselworte

WebLogic Docker DC/OS Mesos Mesosphere Container Kafka Microservices WLSc Multitenant

Einleitung

Viele Firmen haben die Vision alles in Container zu packen und unter DC/OS Mesos laufen zu lassen. DC/OS Mesos sorgt dann für das Clustering und die Ausfallsicherheit über alle Container. Jedoch hat sich der WebLogic Server einen Namen gemacht als Clustering Software und steht mit den neuen Features der Oracle 12 Datenbank gewissermaßen in Konkurrenz zu DC/OS. Wann macht es Sinn WLS zu containerisieren, was sind die Grenzen? In wie weit hilft WLS 12.2 Multitenant eine Micro-Services Strategie umzusetzen? Macht unter DC/OS-Mesos WLS/JMS noch Sinn oder sollte es dann besser Kafka sein? Was genau wird von Oracle zertifiziert bzw. supported und was nicht?

Die Vision von Oracle für Micro-Services und Cloud

Viele Firmen setzen auf eine Micro-Services Strategie, d.h. man ist bestrebt monolithische Applikationen in leichtgewichtige Micro-Services aufzuspalten, die voneinander unabhängig sind, sich keine Daten teilen und ihren eigenen Lebenszyklus isoliert pflegen, um von dem Zwang einer umfassenden Stichtags-Release unabhängig zu werden. Durch diese Entkopplung erreicht man eine höhere Verfügbarkeit, bessere Ausnutzung von Hardware, und die Implementierung von DevOps Prozessen um Continuous Integration zu erreichen. Viele Firmen setzen auf Open Source Produkte aber Oracle hat ab Version 12 ein Angebot geschnürt, das vor allem für Firmen mit bestehender Oracle Basis äußerst reizvoll ist.

Die Lösung lautet nicht unbedingt WLS und Oracle DB in Docker Container zu packen und damit unter DCOS/Mesos oder Kubernetes in Produktion zu gehen – dies würde allenfalls als Übergangslösung in einem Migrationsszenario sinnvoll sein. Aber der WLS im Container ist sinnvoll für Test-, Entwicklungs-, Schulungs- und Support-Zwecke – einfach um eine bestimmte Konfiguration festzuhalten, die dann leicht reproduzierbar und portierbar ist – z.B. auf Docker Hub gibt es ein fertiges OSB 11 Cluster mit Datenbank als Docker Container – mit 6 GB ist dieses Docker-Image nicht gerade leichtgewichtig, aber immer noch kleiner als eine komplette VM. Wenn man nicht vor hat in Produktion zu gehen, spielt die Oracle Zertifizierung und der Support eine untergeordnete Rolle – denn der Docker Container ist schließlich Linux und wenn es unter Linux funktioniert wird‘s auch im Container funktionieren, egal ob Oracle diese spezielle Konfiguration mit einer offiziellen Zertifizierung absegnet oder nicht.

Der WLS im Container hat gegenüber einer VM jedoch den Nachteil, dass man eine zusätzliche „Level of Indirection“ einführt. Der WLS und erst recht die SOA Suite Produkte sind selbst schon immer Cluster- und Orchestrierungs-Plattformen gewesen mit diversen Tools (EM, Admin-Console,

WLST, Config-Wizard, etc.) und diese Container-Plattformen in einen anderen Container zu packen und von einer fremd-Plattform clustern und administrieren zu lassen macht in der Produktion wenig Sinn.

Der WLS ist als JEE Application Server selbst eine monolithische Container-Plattform: der WLS, als JEE Implementierung, versteht sich selbst als Konglomerat verschiedener Container + JMS-Servern + Web-Server + Load-Balancer, die beliebig vielen Artefakten eine Cluster-Laufzeitumgebung bietet – aber genau das konkurriert mit dem Angebot der Open-Source Container-Plattformen. Konsequenterweise ist die Unterstützung von Oracle für Docker lauwarm im Vergleich zu den 12c Produkten.

Der WLS im Container ist unterhalb von 2,2 GB nicht zu haben, außerdem möchte der WLS einiges persistieren, pro Domäne eine eigene technische Datenbank einrichten und am liebsten sich mit den Kollegen clustern. Ein WLS-Lite ist am Horizont nicht auszumachen und Oracle hat wohl nicht vor, von der bewährten Melkkuh Lizenz-Politik abzurücken – d.h. es bleibt dabei, dass pro (potenzieller) Laufzeit eines Clusters kassiert wird. Das Alles geht gegen den Container-Plattform-Strich – im Sinne von Pets vs. Cattle Bild ist der WLS klar in der Rolle von Pets im Gegensatz zu den Containern als „Cattle“. Container sollten als „Cattle“ keine Individualität haben, d.h. sie sind klein und werden nach Bedarf weggeworfen, neu gestartet, neu geschrieben (maximal 2 Wochen Arbeit), usw. WLS im Container bedarf aber aus genannten Gründen immer einer Sonderbehandlung als „Pet“.

Für Docker zertifiziert Oracle Produkte sind im wesentlichen die Folgenden:

- Oracle Database 12*c* Standard Edition 2 and Enterprise Edition
- MySQL und NoSQL
- OracleBI
- Oracle Java und Oracle Linux
- Glassfish
- FMW 12, d.h. WLS 12, SOASuite 12, Coherence 12, Tuxedo 12
- DataIntegrator
- HTTP Server 12, "Fusion Middleware Web Tier"

Siehe Oracle Dox für genaue Auflistung:

https://www.oracle.com/technetwork/middleware/ias/oracleas-supported-virtualization-089265.html#wls1221supportfordockercontainers

https://github.com/oracle/docker-images/tree/master/OracleWebLogic

https://github.com/oracle/docker-images/

Certified Docker Containers - Knowledge Base Doc ID 2017945.1

Oracle has certified the use of Oracle WebLogic Server and Oracle Fusion Middleware products running in Docker containers. For details on all the specific certified configurations, please consult Oracle certification information provided at:

https://www.oracle.com/technetwork/middleware/ias/oracleas-supported-virtualization-089265.html .

The certification information provided indicates host operating system versions and kernel versions that have been certified. Oracle WebLogic Server and Oracle Fusion Middleware products are certified running in Docker containers including these Linux versions (which this note refers to as "certified Docker containers"), and running these certified Docker containers on the host operating systems indicated, including specific Oracle Linux and Red Hat Linux versions. See the link above for details.

Im Wesentlichen sind die Oracle WLS FMW Produkte unter folgenden Bedingugen zertifiziert:

1) Container-Format ist Docker
2) Im Docker Container läuft eine zertifzierte Version von Oracle Linux oder RedHat
3) Der Docker-Container läuft auf einer zertifizierten Version von Oracle Linux
4) Ein WLS Cluster über mehrere Hosts kommuniziert über Docker Overlay Network
5) Passende Version von WLS (ab Version 12c)

WLS Version	JDK Version	Host OS	Kernel	Docker Version
12.2.1	**8**	**Oracle Linux 6 UL 6**	**UEK Release 3 (3.8.13)**	**1.7+**
12.2.1	**8**	**Oracle Linux 7**	**UEK Release 3 (3.8.13) or RHCK 3 (3.10)**	**1.7+**
12.2.1	**8**	**Red Hat Linux 7**	**RHCK 3 (3.10)**	**1.7+**
12.1.3	**7/8**	**Oracle Linux 6 UL 5**	**UEK Release 3 (3.8.13)**	**1.3.3+**
12.1.3	**7/8**	**Oracle Linux 7**	**UEK Release 3 (3.8.13) or RHCK 3 (3.10)**	**1.3.3+**
12.1.3	**7/8**	**Red Hat Linux 7**	**RHCK 3 (3.10)**	**1.3.3+**

Das bedeutet aber auch umgekehrt, dass alle anderen Oracle Produkte bzw. Kombinationen nicht zertifiziert sind, insbesondere ist keines der Version 11 Produkte zertifiziert (obwohl es sie auf Docker Hub Containern gibt, mit WLS 11, OSB 11 Cluster, etc.). Ehe man mit so etwas in Produktion geht, sollte man bedenken, dass der Oracle Support bei vermeintlichen Bugs darauf bestehen wird, dass die Issue auf einer von Oracle zertifizierten Kombination reproduziert wird. Wer mit solchen Services Requests zu tun gehabt hat, weiß, dass die Wahrscheinlichkeit in angemessener Zeit in Patch zu bekommen sich reduziert.

Eines der Gründe warum WebLogic zum führenden JEE Application Server wurde, sind seine bestechenden Clustering Fähigkeiten. Und ähnlich, ein Hauptgrund für den Erfolg der Oracle Datenbanken ist, dass sie in verlässlicher Monotonie eine Enterprise-ausgereifte administrative Basis für SQL bieten – Ausfallsicherheit, Backup, Security, Administration, Industrie-Standards, Support, kalkulierbare Release-Politik – es ist alles da, durchgängig von der kleinen bis zur Mega-Datenbank. Das Gleiche kann man beim besten Willen nicht über die aufsprießenden, neuen Produkte wie DCOS, Marathon, Mesos, Kafka, Cassandra, Docker, usw. sagen – mit ihrer nicht-standardisierten Beta-Qualität, sprunghaften Releases, lückenhaftem Support - generell mit Mut zur Lücke. So überrascht es nicht, dass die Antwort von Oracle auf die Open Source Container und Clustering Angebote (z.B. Docker + Mesos oder Kubernetes) lautet, dass man auf Bewährtes - angereichert durch neue 12c Features - setzt:

- **ORACLE DB12c** Multitenancy Container Database (CDB) bzw. Pluggable Databases (PDB) ab DB 12c - werden von Oracle's Micro-Service Strategie empfohlen, um das Problem der Daten-Trennung zu lösen: ein PDB pro Micro-Service. Ein klarer Trend bei Oracle ist: Pluggable Database. Es ist absehbar, dass der klassische Non-CDB in Zukunft obsolet wird - mit 12cR1 ist non-CDB jetzt schon deprecated. "*The non-CDB architecture was deprecated in Oracle Database 12c. It can be desupported and unavailable in a release after Oracle Database 12c Release 2. Oracle recommends use of the CDB architecture."* https://docs.oracle.com/database/121/UPGRD/deprecated.htm#BABDBCJI
- **WLS 12c mit Traffic Director als Load Balancer und Coherence als Cache "**WebLogic Server Multitenant" ist das Komplement zu Oracle Database 12c pluggable Databases, um eine agile Applikations-Plattform zu implementieren. WLS Multitenant liefert Container-ähnliche Fähigkeiten um 12-factor Applikations-Strategien zu unterstützen aber unter Verwendung von WLS Features wie Clustering, Transaktionsverwaltung und Security. Das dürfte der schnellste und sicherste Weg sein, um monolithische JEE Applikationen in kleinere Services aufzuspalten. Ohne die Oracle JEE-Welt verlassen zu müssen erreicht man ähnliche Vorteile wie mit Docker + Mesosphere oder Kubernetes auf der Basis von bewährter, bekannter Technologie. Die WLS-Container sind konzipiert für Isolation: zur Laufzeit, in Ressourcen-Management (Container eigener Workmanager und Container-speifische Kontrolle von CPU, Heap und File-Opens), Security (der Container kann sein eigenes Security-Realm haben), Administration und vor allem auch in Datenhaltung ob DB, JMS oder Coherence Cache. Die WLS-Container enthalten im Gegensatz zu Docker-Containern nichts vom OS, JVM oder Application Server, sondern erben das vom zu Grunde liegenden WLS. Weil die WLS-Container weder OS, noch JVM, noch Prozesse starten müssen, sondern nur die Applikation selbst, startet der WLS-Container sehr schnell. Die Schwerfälligkeit von WLS im Docker-Container wird von WLS-Multitenant Containern in eine enorme Verfügbarkeit umgekehrt: schnelles Kreieren, Starten und Stoppen von Applikationsprozessen. Hinzu kommt maximale Portierbarkeit zwischen Umgebungen (z.B. live Migration von einem Container zwischen Domänen ohne die Session zu verlieren!) um Parität zwischen Entwicklung und Produktion zu erreichen. So wie ein Docker Container überall laufen kann wo ein Docker Runtime vorhanden ist (weil die Abhängigkeiten von Ressourcen containerisiert, d.h. Teil des Containers sind) , so kann ein WLS Microcontainer überall laufen wo WLS12c läuft. Ähnlich wie bei Marathon von Mesosphere ist der Traffic-Director eng integriert mit den WLS-Containern – z.B., Kreieren oder Import eines

Containers wird automatisch registriert und in die Lastverteilung aufgenommen. Man kann mit WLS12c sehr effizient Mandantenfähigkeit implementieren, in dem man die gleichen Apps in mehrere Microcontainer steckt, die sich nur in den Ressourcen unterscheiden (z.B. Datenbank, Security, etc.). Oracle verspricht, dass man durch Konsolidierung von Applikationen in WLS Mikrocontainern bis zu Faktor 3 mal so viele Applikationen auf der gleichen Hardware laufen lassen kann, weil man nicht mehr auf der Domänen Ebene isolieren muss, sondern auf der Partitions Ebene schon isolieren kann. Durch diese Virtualisierung auf höherer Ebene erreicht man eine höhere Deployment Dichte. Siehe: (http://www.oracle.com/us/products/middleware/cloud-app-foundation/weblogic/weblogic-server-multitenant-ds-2742664.pdf)

- **ORACLE DB 12c mit SODA**: http://www.oracle.com/technetwork/database/application-development/oracle-document-store/index.html https://blogs.oracle.com/jsondb/oracle-rdbms-as-a-nosql-json-document-store - ein NoSql Interface (SODA) - Simple Oracle Document Access (SODA) ist eine REST API die auf DB 12c (12.1.0.2) installiert wird. Damit kann man Kollektionen von Dokumenten in der Oracle DB speichern, lesen und abfragen ohne SQL zu kennen oder zu wissen wie die Dokumente in der Datenbank abgespeichert sind. Man kann SODA verwenden um die herkömmlichen CRUD Operationen auf Dokumente jeglicher Art auszuführen, insbesondere auch auf JSON Dokumente. Für ein Shop wo viel Oracle DB Know-how schon vorhanden ist, wäre dies die Dokumenten-Datenbank der ersten Wahl.
- **Oracle NoSQL Database**: Key-Value Datenbank, automatisches Replizieren um Ausfallsicherheit zu garantieren. http://www.oracle.com/technetwork/database/database-technologies/nosqldb/overview/index.html Von Open-Source Produkten wie Cassandra, Kafka, HDFS, Elasticsearch, Mesosphere, usw. ist das Konzept bekannt, dass das Produkt Out-Of-The-Box seinen eigenen Replizierungsmechanismus implementiert. Es wird bei diesem Konzept davon ausgegangen, dass Hardware & Netzwerke unweigerlich mal versagen werden und deswegen schon die Basis-Version des Produkts mit konfigurierbarer Ausfallsicherheit ausgestattet sein muss, anstatt dies erst nachträglich mit hohem Aufwand zu konfigurieren. Dem trägt Oracle mit einem neuen Produkt „Oracle NoSQL Database“ Rechnung.
- **OEP** - Als Streaming, bzw. Messaging BackBone eignet sich OEP (Oracle Event Processing) um Millionen von Messages per Sekunde zu bewältigen (vgl. Apache Kafka, das aus einem Framework für Social Media Anbieter LinkedIn erwachsen ist). OEP ist ergänzend zu JMS oder eventuell auch AQ zu sehen – d.h. JMS oder AQ wird immer präferiert sein, dort wo eine transaktional geschützte Persistenz verlangt wird (Kafka kann neuerdings auch Transaktionen aber ist nicht XA-fähig). OEP ist ein leichtgewichtiger Java-Container, der einige Komponenten von WLS übernommen hat: (http://www.oracle.com/us/products/middleware/soa/overview/complex-event-processing-ds-066411.pdf).
- **Jersey** - Das neue von Oracle gesponserte Framework zur Erstellung von REST Web Services, „Jersey“, wird nicht mit WLS, sondern mit Glassfish als Stand-Alone Server ausgeliefert.

Oracle hätte am liebsten, dass man sich nicht mehr die Mühe macht diese Produkte selbst on-site zu installieren, sondern die Features als Cloud-Services von Oracle abonniert. Es kommt wohl jetzt schon vor, dass neue Produkte erst mal nur in der Oracle Cloud angeboten werden, aber prinzipiell kann man alles on-site installieren und damit eine Private-Cloud implementieren. Monolithische on-Site Legacy Ansätze werden bei Oracle zukünftig immer weniger Unterstützung finden.

Kontaktadresse:

Dr. Páll Eggerz

virtual7 GmbH

Reichenbachstraße 1

D-80469 München

Telefon:	+49 (0) 15154000042
Fax:	+49 (0) 89 122 28 31 20
E-Mail	pall.eggerz@virtual7.de
Internet:	www.virtual7.de