

Oracle Unified Directory in Docker

Directory Server verpackt im Container

Stefan Oehrli



BASEL ▪ BERN ▪ BRUGG ▪ DÜSSELDORF ▪ FRANKFURT A.M. ▪ FREIBURG I.BR. ▪ GENÈVE
HAMBURG ▪ KOPENHAGEN ▪ LAUSANNE ▪ MÜNCHEN ▪ STUTTGART ▪ WIEN ▪ ZÜRICH

trivadis
makes IT easier. ■ ■ ■

■ Trivadis – Unsere Mission.

Trivadis makes IT easier:

- Wir unterstützen unsere Kunden massgeblich bei der **intelligenten Nutzung** von Daten im **digitalen Zeitalter**.
- Wir reduzieren **Komplexität** für unsere Kunden durch herausragende **Technologiekompetenz**.
- Wir übernehmen **Kernaufgaben** der bestehenden und zukünftigen IT unserer Kunden.



■ Trivadis – Was uns auszeichnet und unterscheidet.

- Wir verstehen die Business-Prozesse und wirtschaftlichen Herausforderungen unserer Kunden und unterstützen sie durch IT-Beratung und bei der Entwicklung ganzheitlicher IT-Lösungen.
- Unsere selbstentwickelten, bewährten Produkte und Methoden basieren auf dem tiefen Know-how in den Kerntechnologien von Microsoft, Oracle und Open Source. Dies unterscheidet uns von unserem Wettbewerb.



trivadis
makes IT easier. ■ ■ ■

■ Trivadis – Unsere wichtigsten Kennzahlen.



- Gründung: 1994.
- 15 Trivadis Niederlassungen mit über 650 Mitarbeitenden.
- Umsatz CHF 111 Mio. (EUR 96 Mio.).
- Über 250 Service Level Agreements.
- Mehr als 4'000 Trainingsteilnehmer.
- Forschungs- und Entwicklungsbudget: CHF 5.0 Mio.
- Mehr als 1'900 Projekte pro Jahr bei über 800 Kunden.
- Finanziell unabhängig und nachhaltig profitabel.

trivadis
makes IT easier. ■ ■ ■

■ Stefan Oehrli



Solution Manager BDS SEC / Trivadis Partner

- Seit 1997 IT-Bereich tätig
- Seit 2008 bei der Trivadis AG
- Seit 2010 Disziplin Manager SEC INFR
- Seit 2014 Solution Manager BDS Security

IT Erfahrung

- DB Administration und DB Security Lösungen
- Administration komplexer, heterogenen Umgebungen
- Datenbank Teamleiter

Spezialgebiet

- Datenbank Sicherheit Security und Betrieb
- Sicherheitskonzepte und deren Umsetzung
- Sicherheitsbewertungen
- Oracle Backup & Recovery
- Enterprise User Security und Oracle Unified Directory

Skills

- Backup & Recovery
- Oracle Advanced Security
- Oracle AVDF und DB Vault
- Oracle Directory Services
- Team / Projekt Management
- Referent O-SEC, O-BR,...

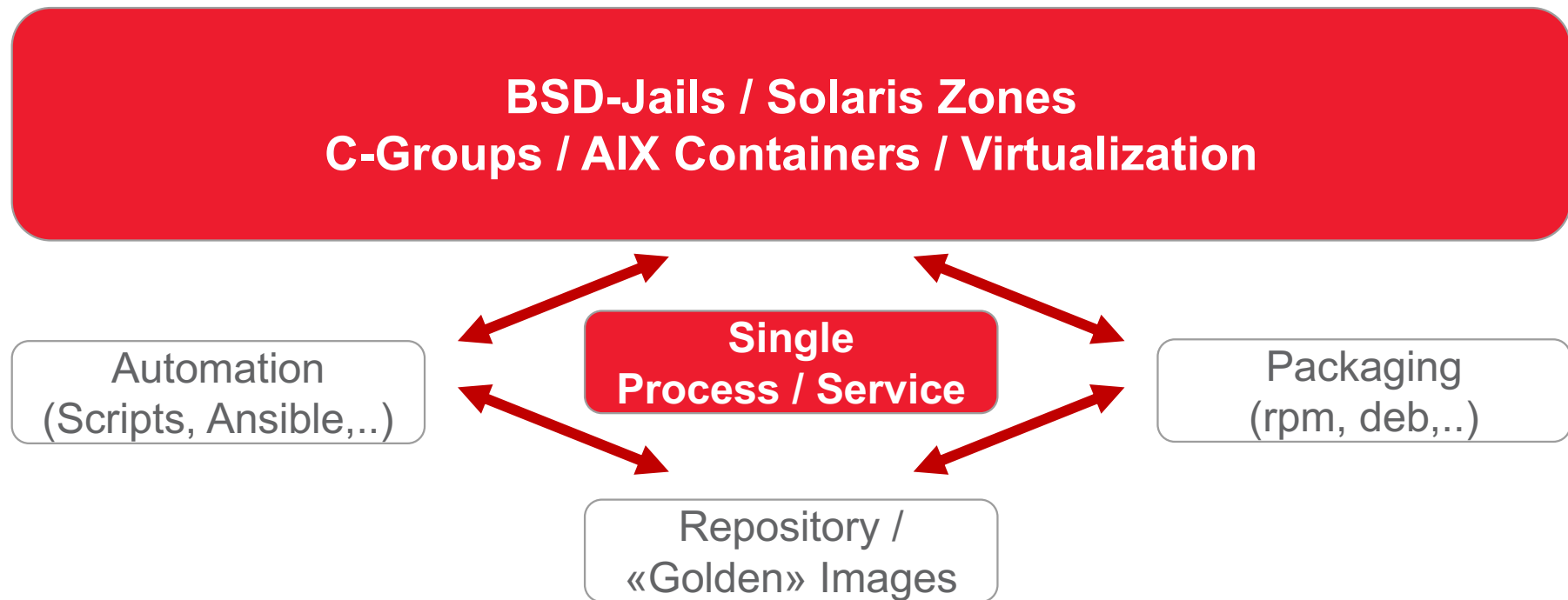
trivadis
makes IT easier. ■ ■ ■

■ Agenda

1. Docker in a Nutshell
2. OUD Installation und Voraussetzungen
3. Erstellen der OUD Images
4. Nutzung der OUD Images / Container
5. Anwendungsfälle und Demo's
6. Fazit

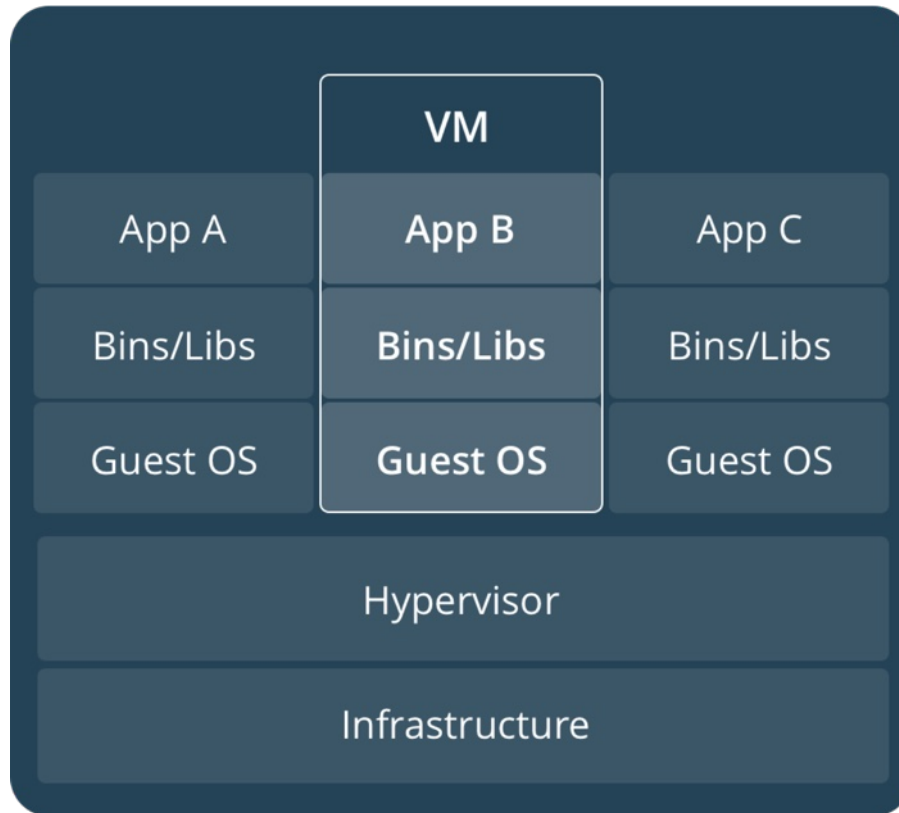
Docker in a Nutshell

■ Container



Austauschbarkeit und Reproduzierbarkeit

■ Virtualisierung

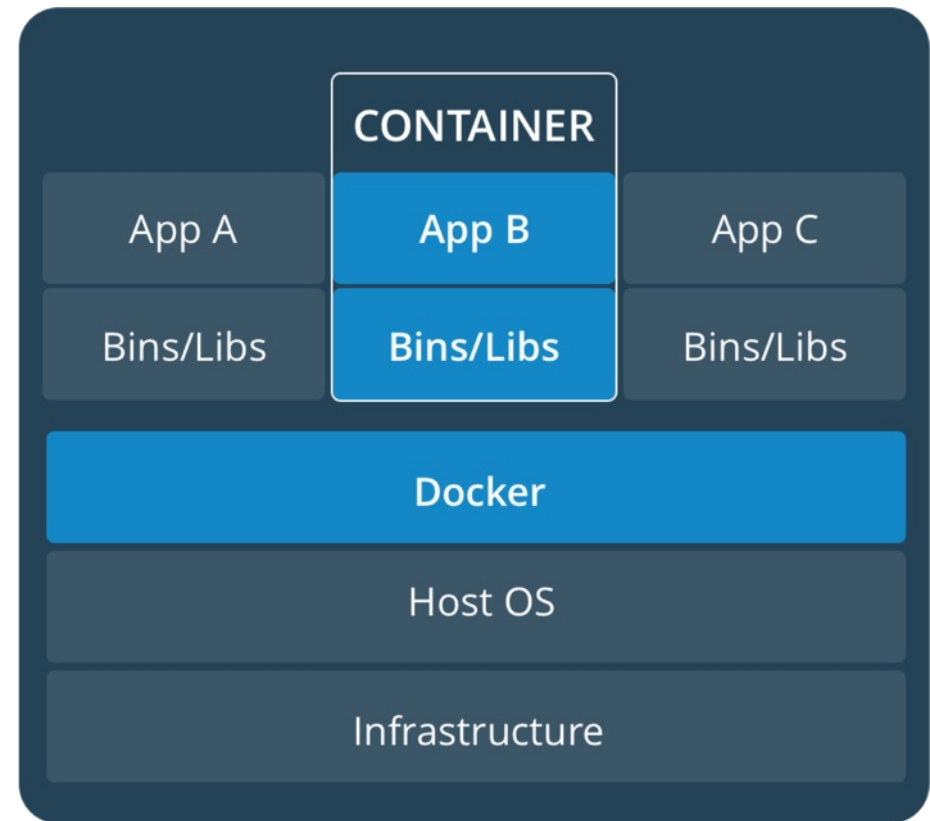


Source: Docker Inc (February 2018)

- Nutzung gemeinsamer Infrastruktur
- Jede VM ist ein «Server»
 - Guest OS
 - Software und Bibliotheken
 - Anwendung
- Redundanz
- Aufwand für den Aufbau der VM's

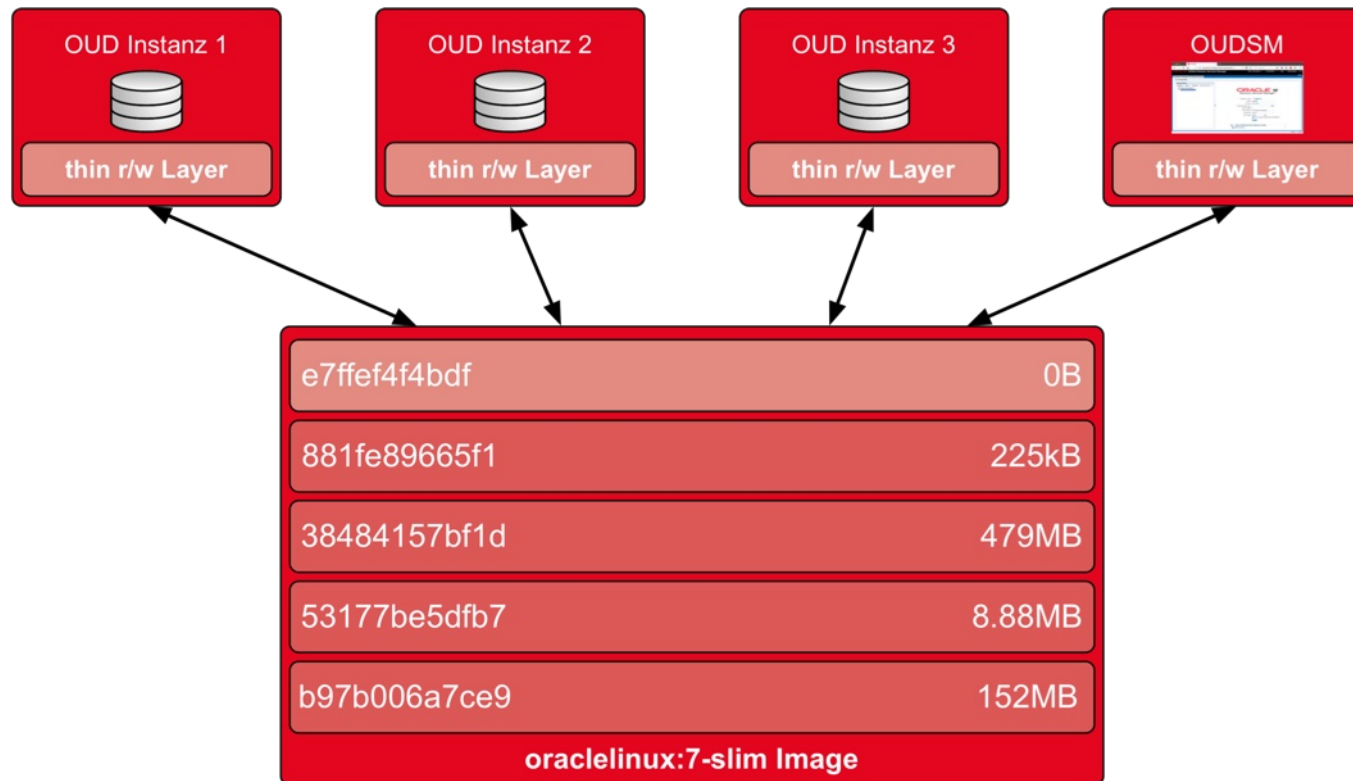
■ Docker Container

- Nutzung gemeinsamer Infrastruktur
- Ein Host OS
- Anwendungen sind «paketiert»
 - Nur nötige Bibliotheken und Software Komponenten
- Nutzen von bestehenden Images
- Schlank
- Reproduzierbar
 - Weil automatisiert erstellt



Source: Docker Inc (February 2018)

■ Docker Container und Layer



■ Was sonst noch über Container

- Es gibt nicht “den Container”
- Single Prozess / Service
 - Kein Ersatz für eine VM
- Geringer Platzbedarf
- Verstehen, wie man Container baut, aber ...
 - ... nicht jeder Container muss individuell angepasst werden.
 - ... verfügbare und vertrauenswürdige Container-Repositorys verwenden
- Achten Sie auf den Umgang mit Identitäten und sensiblen Daten.
- Umgang mit Datenpersistenz



OOD Installation und Voraussetzungen

■ Support und Lizenzierung

- Aktuell keine offiziellen Docker Images oder Build Skripte für Oracle Unified Directory
 - Allenfalls bald im entsprechenden GitHub Repository von Oracle siehe [Issue 656](#)
- Oracle unterstützt Oracle Unified Directory auf Docker grundsätzlich
 - Verifiziert via Service Request
 - Beitrag von Oracle Support in der Community
[Oracle FMW products certification on Docker](#)
 - Zertifizierung der Komponenten beachten (z.B. Oracle JDK)
- Lizenzierung ist ähnlich «optimal» wie bei VM Umgebungen
 - Lizenzierung der entsprechenden CPU's
 - Im Fall von OUD bei User Lizenzierung nicht all zu kompliziert

■ Voraussetzungen

- Docker Umgebung
 - Docker Community Edition 18.05.0
- Basis Image für das Oracle Unified Directory Image
 - Betriebssystem Oracle Enterprise Linux 7.5 (slim)
 - Oracle Server JRE 8 Update 172 ([p27412890_180172_Linux-x86-64.zip](#))
- Oracle Unified Directory Installation
 - Oracle Unified Directory 12.2.1.3.0 ([p26270957_122130_Generic.zip](#))
 - Aktuelle Patch Set Updates vom April 2018
 - OUD Docker Build Skripte (GitHub [oehrlis/docker](#))
 - OUD Umgebungsskripte (GitHub [oehrlis/oudbase](#))

■ Voraussetzungen für OUDSM

- Zusätzlich für Oracle Unified Directory Services Manager
 - Oracle Fusion Middleware Infrastructure 12.2.1.3.0
([p26269885_122130_Generic.zip](#))
 - Aktuelle Patch Set Updates vom April 2018

■ OUD Installation (1)

Mit OUD 12c hat Oracle verschiedene Implementierungsmethode eingeführt

■ **Standalone** OUD Server

- Nur OUD Software ohne Fusion Middleware und somit ohne OUDSM
- Geringer Platzbedarf d.h rund 400MB für OUD
- Administration mit remote OUDSM oder Kommandozeile, LDAP Browser etc.
- Verwendung für OUD Docker Images

■ OUD Installation (2)

■ **Collocated** OUD Server mit OUD und OUDSM in **separaten Domains**

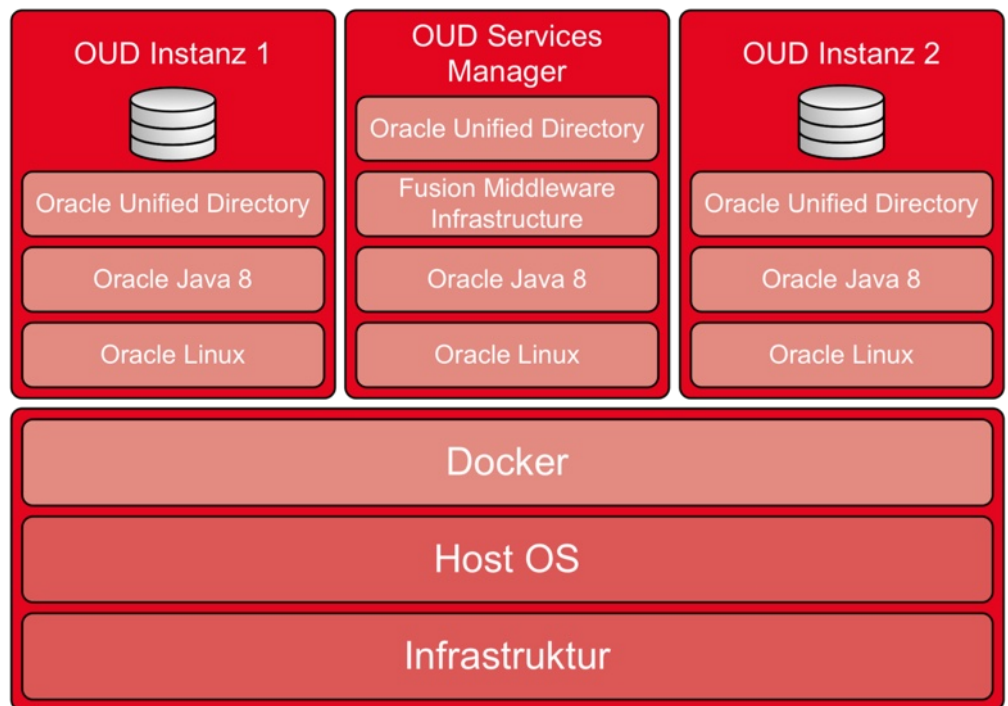
- Benötigt wesentlich mehr Platz für Oracle Fusion Middleware Infrastructure
- Administration mit OUDSM oder Kommandozeile, LDAP Browser etc
- Konfiguration von Weblogic und entsprechender Domains
- Verwendung für OUDSM Docker Images

■ **Collocated** OUD Server mit OUD und OUDSM in **einer Domain**

- Zusätzlich werden die Oracle Database Binaries benötigt
- Administration von OUD via Weblogic respektive wlst
- komplexer und benötigt zusätzliche Installationsschritte

■ Übersicht OUD Docker Container

- Mehrere OUD Container
- Ein OUDSM Container
- Zentrale Verwaltung
- Gleiche Layer werden «geteilt»



Erstellen der OUD Images

■ OUD Basis Image

- OUD benötigt als Basis ein Docker Image mit Oracle JDK (TAG *oracle/serverjre:8*)
 - Download aus der Oracle Container Registry
 - Erstellen analog Oracle Docker GitHub Repository
 - Erstellen analog meinem GitHub Repository
 - Selber eine Image erstellen

```
docker login container-registry.oracle.com
```

```
docker pull container-registry.oracle.com/java/serverjre:8
```

```
docker tag container-registry.oracle.com/java/serverjre:8 \
oracle/serverjre:8
```

■ Erstellen des OUD Image

- GitHub Repository oehrlis/docker clonen
- Bereitstellend der Software (OUD und Patches) im Build Verzeichnis

```
cp p26270957_122130_Generic.zip docker/OracleOUD/12.2.1.3.0  
cp p27742743_122130_Generic.zip docker/OracleOUD/12.2.1.3.0
```

- Mit dem Aufruf von *docker build* das Image erstellen

```
cd docker/OracleOUD/12.2.1.3.0  
docker build -t oracle/oud:12.2.1.3.180322 .
```


■ OUD Docker File (1)

- Definition des Base Images
- Setzen des Maintainers
- Setzen der verschiedenen Umgebungsvariablen
 - ORACLE_ROOT
 - ORACLE_BASE
 - ORACLE_DATE

```
FROM oracle/serverjre:8

# Maintainer
# -----
LABEL maintainer="stefan.oehrli@trivadis.com"

# Arguments for Oracle Installation
ARG ORACLE_ROOT
ARG ORACLE_DATA
ARG ORACLE_BASE
ARG ORAREPO

# Environment variables required for this build (do NOT change)
# -----
ENV ORAREPO=${ORAREPO:-orarepo} \
  DOWNLOAD="/tmp/download" \
  DOCKER_SCRIPTS="/opt/docker/bin" \
  START_SCRIPT="start_oud_instance.sh" \
  CHECK_SCRIPT="check_oud_instance.sh" \
  INSTALL_SCRIPT="setup_oud.sh" \
  USER_MEM_ARGS="-Djava.security.egd=file:/dev/./urandom" \
  ORACLE_HOME_NAME="fmw12.2.1.3.0" \
  ORACLE_ROOT=${ORACLE_ROOT:-/u00} \
  ORACLE_DATA=${ORACLE_DATA:-/u01} \
  OUD_INSTANCE=${OUD_INSTANCE:-oud_docker} \
  PORT=${PORT:-1389} \
  PORT_SSL=${PORT_SSL:-1636} \
  PORT_REP=${PORT_REP:-8989} \
  PORT_ADMIN=${PORT_ADMIN:-4444} \
  FMW_OUD_PKG="p26270957_122130_Generic.zip" \
  OUD_PATCH="p27742743_122130_Generic.zip"

# Use second ENV so that variable get substituted
ENV ORACLE_BASE=${ORACLE_BASE:-$ORACLE_ROOT/app/oracle} \
  OUD_INSTANCE_BASE=${OUD_INSTANCE_BASE:-$ORACLE_DATA/instances}
```

■ OUD Docker File (2)

- Task als Benutzer root
- Benutzer und Gruppen anlegen
- Verzeichnisse erstellen
- Optional bei Bedarf fehlende Pakete nachinstalliert
 - tar, gzip und libaio
- Anpassen der Java Security
 - Workaround für EUS siehe MOS Note [2397791.1](#)

```
# RUN as user root
# -----
# - create group oracle and oinstall
# - create user oracle
# - setup subdirectory to install OUDpackage and container-scripts
# - create softlink for the OUD setup scripts
# - adjust ownership of download folder
# - relax java.security and allow 3DES_EDE_CBC
# -----
RUN groupadd --gid 1000 oracle && \
  groupadd --gid 1010 oinstall && \
  useradd --create-home --gid oracle --groups oracle,oinstall \
  --shell /bin/bash oracle && \
  install --owner oracle --group oracle --mode=775 --verbose --directory \
  ${ORACLE_ROOT} \
  ${ORACLE_BASE} \
  ${ORACLE_DATA} \
  ${DOWNLOAD} \
  ${DOCKER_SCRIPTS} && \
  ln -s ${ORACLE_DATA}/scripts /docker-entrypoint-initdb.d && \
  chown oracle:oinstall ${DOWNLOAD} && \
  sed -i 's/, 3DES_EDE_CBC/' $(find /usr/java -name java.security)
```

■ OUD Docker File (3)

- Kopieren der Skripte
- Kopieren der Software
- Start der OUD Installation mit setup_oud.sh
- Start der OUD Base Installation mit setup_oudbase.sh

```
# Copy scripts and software
# -----
# copy all setup scripts to DOCKER_BIN
COPY scripts/* "${DOCKER_SCRIPTS}/"

# COPY oud/software and response files
COPY *zip* install.rsp oraInst.loc "${DOWNLOAD}/"

# RUN as oracle
# Switch to user oracle, oracle software as to be installed with regular user
# -----
USER oracle
RUN "${DOCKER_SCRIPTS}/${INSTALL_SCRIPT}" ${FMW_OUD_PKG} ${OUD_PATCH}

# get the latest OUD base from GitHub and install it
RUN "${DOCKER_SCRIPTS}/setup_oudbase.sh"
```

■ OUD Docker File (4)

- Festlegen der Ports für den Export
- Health Check Script
- Verzeichnis \$ORACLE_DATA als Volume definieren
- start_oud_instance.sh als standard Kommando festlegen

```
# Finalize image
# -----
# expose the OUD admin, replication and ldap ports
EXPOSE ${PORT} ${PORT_SSL} ${PORT_REP} ${PORT_ADMIN}

# run container health check
HEALTHCHECK --interval=1m --start-period=5m \
  CMD "${DOCKER_SCRIPTS}/${CHECK_SCRIPT}" >/dev/null || exit 1

# Oracle data volume for OUD instance and configuration files
VOLUME ["${ORACLE_DATA}"]

# set working directory
WORKDIR "${ORACLE_BASE}"

# Define default command to start OUD instance
CMD exec "${DOCKER_SCRIPTS}/${START_SCRIPT}"
```

■ OUD Docker Image Grösse

- Die so erstellten Images sind unnötig gross
 - Images enthalten aufgrund der Docker Build Layers die Software in den Layer
- Verwendung des Parameter `--squash` bei `docker build`
 - Alle Layer werden zusammengeführt
 - Intermediate Layer der gross ist «verschwindet»
 - Layer History geht verloren
- Alternative lokaler Software Download beim Erstellen der Docker Images
 - `setup_oud.sh` unterstützt dies und kann fehlende Software mit `curl` herunterladen

■ Build mit download (1)

- Erstellen eines lokalen HTTP Servers (ja auch ein Docker Container ☺)

```
docker run -dit --hostname orarepo \  
-p 8080:80 --name orarepo \  
-v /Volumes/orarepo:/usr/local/apache2/htdocs/ \  
httpd:alpine
```

- Kopieren der Software in das Volume Verzeichnis vom HTTP Server

```
cd OracleOUD/12.2.1.3.0  
cp p26270957_122130_Generic.zip /Volumes/orarepo  
rm p26270957_122130_Generic.zip
```

■ Build mit download (2)

- IP Adresse des HTTP Servers in eine Variable speichern

```
orarepo_ip=$(docker inspect -f '{{range  
.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' orarepo)
```

- Bei *docker build* entsprechend den Host orarepo angeben

```
docker build --add-host=orarepo:${orarepo_ip} \  
-t oracle/oud:12.2.1.3.180322 .
```


Nutzung der OUD Images / Container

■ Starten eines OUD Containers

■ Mit *docker run* wird ein neuer OUD Container erstellt und gestartet

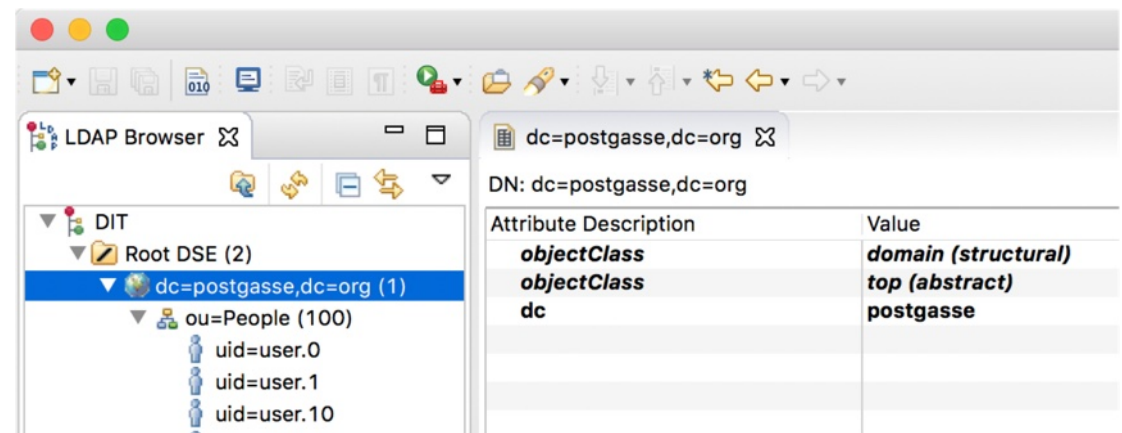
```
docker run --detach \  
--volume /Data/vm/docker/volumes/oudeng:/u01 \  
-p 1389:1389 -p 1636:1636 -p 4444:4444 \  
--hostname oudeng --name oudeng \  
oracle/oud:12.2.1.3.0
```

■ Parametrisierung

- Volume Verzeichnis für Ablage der OUD Instanz «ausserhalb» des Containers
- Definition der Parameter für das Port Forwarding
- Angabe eines Hostnamens und Container Namen

■ Anpassung des OUD Container

- Anlegen einer Beispiel Instanz
- Spezifische Instanzen durch
 - Parametrisierung
 - Umgebungsvariablen
 - Konfigurationsdateien



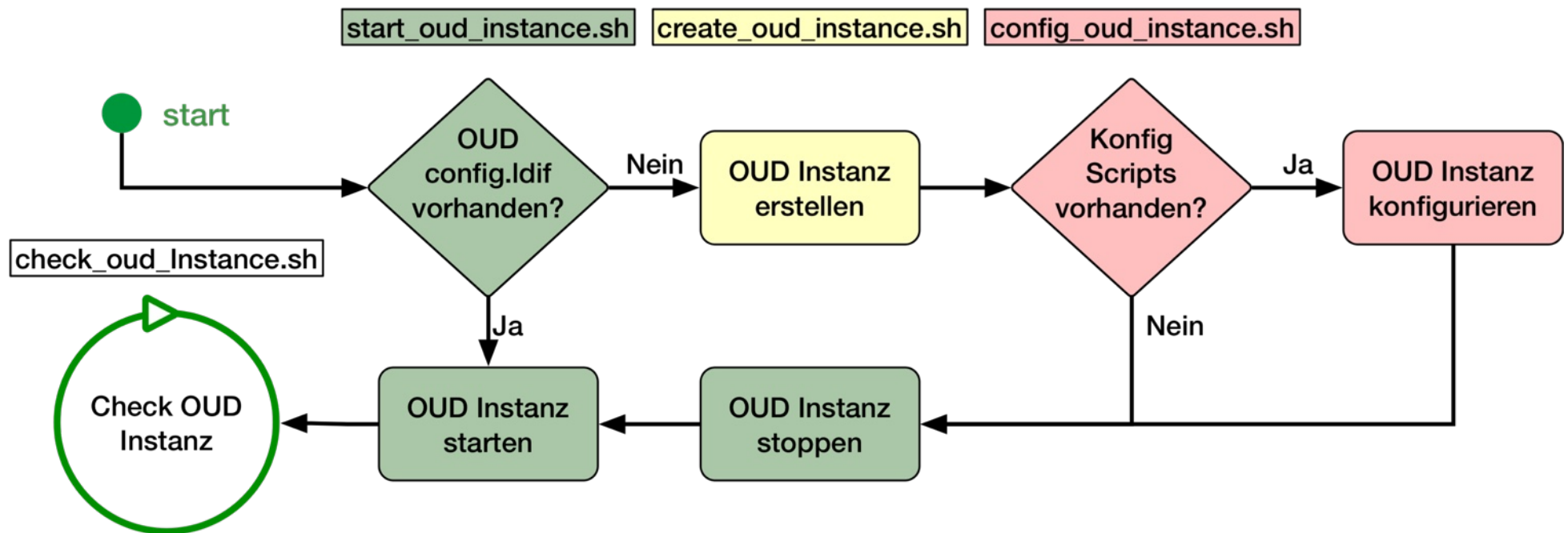
■ Container Variablen

Variable	Verwendung
OID_INSTANCE	OID Instanz Name. Standardwert oid_docker
CREATE_INSTANCE	Flag zum Erstellen der OID Instanz beim Starten. Standardwert TRUE.
OID_INSTANCE_BASE	Basis Verzeichnis für die OID Instanzen. Standardwert \$ORACLE_DATA/instances
OID_INSTANCE_HOME	OID Instanz Home Verzeichnis. Standardwert \${OID_INSTANCE_BASE}/\${OID_INSTANCE}
OID_INSTANCE_ADMIN	OID Instanz Admin Verzeichnis. Standardwert \${ORACLE_DATA}/admin/\${OID_INSTANCE}
ADMIN_USER	OID Directory Root Benutzer. Standardwert cn=Directory Manager
ADMIN_PASSWORD	Password für den OID Directory Root Benutzer. Wird automatisch generiert und in \$PWD FILE abgelegt.

■ Container Variablen

Variable	Verwendung
PWD_FILE	Password File mit dem clear Text Password vom Directory Root Benutzer. Standardwert <code>\${OUD_INSTANCE_ADMIN}/etc/\${OUD_INSTANCE}_pwd.txt</code>
SAMPLE_DATA	Flag für das Erstellen der Beispiel Daten. Möglich Werte sind TRUE, FALSE oder Anzahl der Datensätze. Standardwert TRUE
OUD_INSTANCE_INIT	Verzeichnis für die Instanz Konfigurationsdateien. Standardwert <code>\$ORACLE_DATA/scripts</code>
OUD_PROXY	Erstellen einer OUD Proxy Instanz anstelle eines Verzeichnisseservers. Standardwert FALSE
OUD_CUSTOM	Komplette Erstellung der Instanz mit Konfigurationsdateien aus dem Verzeichnis <code>\$OUD_INSTANCE_INIT</code> . Standardwert FALSE

■ Startup Abfolge



■ OUD Docker Skripte

- **check_oud_instance.sh** Healthcheck des Docker Containers
- **config_oud_instance.sh** konfiguration der OUD Instanz anhand von Konfigurationsdateien (*.sh, *.conf, *.ldif) in \${OUD_INSTANCE_ADMIN}/create
- **create_oud_instance.sh** erstellt die OUD Instanz
- **setup_oud.sh** wird beim Build zur Installation von OUD benötigt
- **setup_oudbase.sh** wird beim Build zur Installation der OUD Base genutzt.
- **start_oud_instance.sh** startet die OUD Instanz und definiert entsprechende Signal Handler für SIGINT, SIGTERM und SIGKILL
 - Ist keine OUD Instanz vorhanden, wird das Skript **create_oud_instance.sh** aufgerufen.

Anwendungsfälle und Demo

■ Mögliche Anwendungsfälle

- Adhoc Testinstanz für die Analyse von OUD Problemen
 - Z.b im Rahmen eines Service Requests
- Testmigration von ODSEE nach OUD
- Entwicklung eines OUD Proxy für die Integration weiteren Verzeichnissen
- Einsatz von Oracle Enterprise User Security mit OUD
- Testen verschiedener OUD Versionen
- Replikation von mehreren OUD Instanzen in verschiedenen Docker Containers
- OUDSM Container
- Und vieles mehr

■ Demo

- Erstellen einer OUD Proxy Instanz für EUS mit AD Integration

```
docker run --detach \  
--volume /Data/vm/docker/volumes/doag :/u01 \  
-p 1389:1389 -p 1636:1636 -p 4444:4444 \  
-e OUD_CUSTOM=TRUE -e BASEDN="dc=postgasse,dc=org" \  
-e OUD_INSTANCE=oud_adproxy \  
--hostname oudad.postgasse.org --name doag  
oracle/oud:12.2.1.3.180322
```

Fazit

- Der Aufbau von OUD in Docker ist relativ einfach und bietet einiges
- OUD Docker Images lassen sich in vielen Bereichen einsetzen
- Lizenzierung von Oracle Produkten in Docker ist generell Suboptimal 😊
- Allenfalls gibt es bald auch Offizielle Oracle Docker Build Skripte für OUD

■ Links und Quellen

- Oracle Container Registry <https://container-registry.oracle.com>
- Oracle Docker GitHub <https://github.com/oracle/docker-images>
- Blog Post OUD to go on Raspberry Pi Zero <http://url.oradba.ch/2AHzfbi>
- My Oracle Support OUD 12c Zertifizierung <https://url.oradba.ch/MOSoud12c>
- Stefan Oehrli GitHub Docker Repository <https://github.com/oehrlis/docker>
- OUD Umgebungsskripte <https://github.com/oehrlis/oudbase>
- Blog Post Smaller Oracle Docker Images <https://url.oradba.ch/2ut6jEH>
- OUD 12c - EUS Integration Failing with Message "no cipher suites in common"
<https://support.oracle.com/epmos/faces/DocumentDisplay?id=2397791.1>

Fragen und Antworten

Stefan Oehrli
Solution Manager / Trivadis Partner

Tel.: +41 58 459 55 55
stefan.oehrli@trivadis.com



 **@stefanoehrli**
<http://www.oradba.ch>



trivadis
makes IT easier. ■ ■ ■