whitehorses

member of the human network

WHITEBOOK

Complete Silent install voor Oracle 12c Weblogic/OSB omgeving

December 2018

Auteur:
Maurik-Jan Veenman
INTEGRATIESPECIALIST

Inleiding

Mijn klant vroeg mij om voorbereidingen te treffen voor de migratie van de Oracle Weblogic/ OSB omgevingen van versie 11g naar 12c.

Ze gaven daarbij aan dat ze graag een geautomatiseerde installatie wilden in plaats van een handmatige installatie.

Voordelen van een geautomatiseerde installatie zijn:

- het kost weinig tijd
- het gebeurt iedere keer op dezelfde manier
- de installatie en configuratie is door de hele OTAP straat hetzelfde
- het gaat foutloos
- je kunt regelmatig een omgeving even ‘resetten’ door hem opnieuw te installeren
- snelle disaster recovery

Om deze redenen is het ook hip en wordt er veel om gevraagd.

Er zijn natuurlijk pakketten voor zoals Myst, en vanuit de Enterprise Manager is ook provisioning mogelijk. Deze pakketten en de Enterprise Manager zijn niet goedkoop, en dus niet altijd voorhanden. Oracle levert zelf ook template files voor installaties, en op het internet zijn ook diverse blogs te vinden hoe dit aan te pakken. Daarom heb ik besloten het zelf te doen.

Bij mijn vooronderzoek op het internet kwam ik veel informatie tegen over een geautomatiseerde installatie van de software, al wat minder over het patchen van die software, maar bijna niets over de domeincreatie, waarin je juist je omgeving zoals gewenst instelt. Dit bracht mij op het idee om zo’n complete installatie eens op een rijtje te zetten en te beschrijven. Daarover gaat dit Whitebook.

Wat gaan we doen?

De wens van de klant was om een OTAP-straat te vernieuwen. De scripts behelzen een installatie van een Ontwikkel-, Test- , Acceptatie- en Productie-omgeving. De acceptatie- en productie-omgeving bevatten twee nodes.

De installatie is voor een Linux omgeving. Een en ander is ontwikkeld en getest op een Redhat Enterprise Linux 7 omgeving.

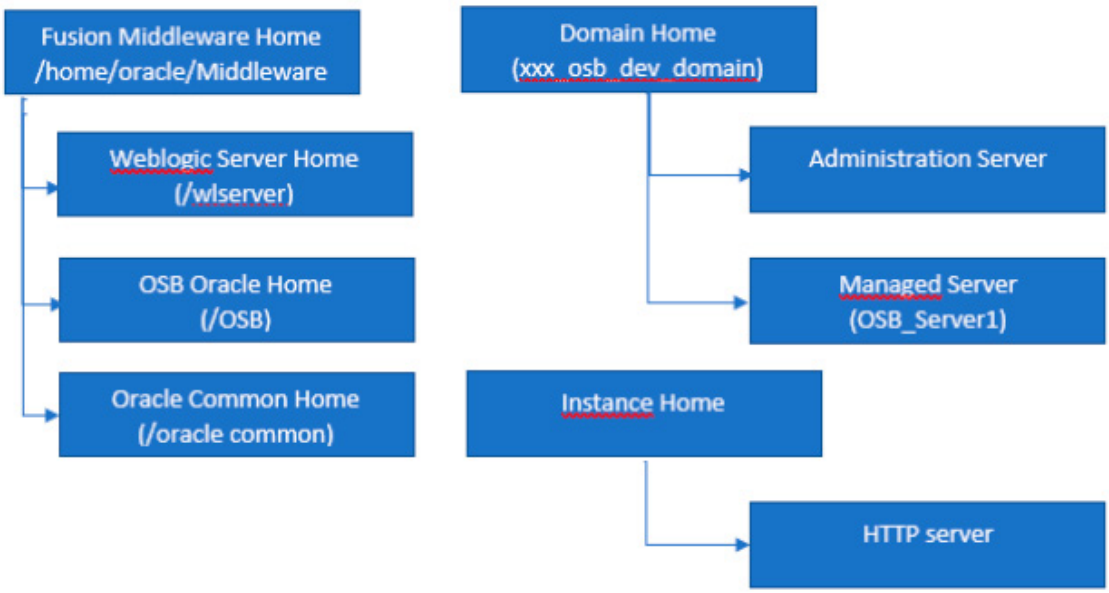
Er wordt gebruik gemaakt van Python.

De meest recente Oracle FMW omgeving wordt gebruikt (versie 12.2.1.3.0) met Oracle HTTP server

Hierbij hoort Jdk 1.8.0_161

Daarnaast worden de specifieke patches geïnstalleerd.

De omgeving ziet er na installatie als volgt uit:



De Weblogic, OSB en Oracle HTTP software worden geïnstalleerd in de Fusion Middleware Home (dat is de plek waar de Fusion Middleware componenten worden geïnstalleerd, als SOA of BPM ook geïnstalleerd zouden worden, dan gebeurt dat hier).

Vervolgens wordt er een Domein gecreëerd met daarin een Administration server en een cluster waarin de Managed OSB server draait.

Tevens wordt er een Oracle HTTP server (OHS) geïnstalleerd voor de loadbalancing en reverse proxy.

De set scripts wordt op de Linux server geplaatst in een aan te maken directory /apps/stage.

De scripts hebben een temp directory nodig waar de Oracle user als wie de installatie doet kan schrijven.

Deze heet /apps/oracle/temp

Opzet van de scripts

De set scripts bestaat uit een hoofdscript, van waaruit diverse subscripts en Oracle installatietools worden aangeroepen.

Het hoofdscript wordt aangeroepen met het trigram (dev / tst / acc / prd) van de naam van de omgeving: dus ./Silentinstall.sh dev voor de development omgeving.

Het trigram zorgt ervoor dat er omgeving specifieke scripts worden aangeroepen, en dat ook de omgeving specifieke database wordt aangeroepen voor de repository installatie.

Wanneer het hoofdscript wordt gestart worden achtereenvolgens de volgende stappen uitgevoerd:

1. Verwijderen oude repository omgeving (optionele stap)
2. Installatie Weblogic infrastructuur
3. Installatie Oracle Service Bus
4. Installatie Oracle HTTP server
5. Opatch patchen naar versie 13.9.4
6. Weblogic patchen met patch 27912627
7. Installatie repository
8. Creatie domein

Stappen 2, 3 en 4 werken allemaal met het aanroepen van Java en een uit te voeren Jar, die gebruik maakt van een response file voor de inrichting. Een response file bevat de input voor variabele waarden voor de inrichting (zoals servernamen, domeinnamen, poorten en dat soort zaken). Het hoofdscript 'bevraagt' deze file voor de waarden van bepaalde variabelen, vandaar dat het een response file heet.

Dit ziet er als volgt uit:

Hoofdscript en Responsefiles

Dit is het hoofdscript:

```
rem execute Weblogic installer in silent mode
```

```
rem M.J.H. Veenman
```

```
#Drop Repository
```

```
/apps/oracle/Middleware_${1}/oracle_common/bin/rcu -silent
-dropRepository -databaseType ORACLE -connectString <database connect
string van jouw omgeving> -dbUser SYS -dbRole SYSDBA -schemaPrefix
DEV -selectDependentsForComponents true -variables SOA_PROFILE_
TYPE=MED,HEALTHCARE_INTEGRATION=NO -component MDS -component IAU -component
IAU_APPEND -component IAU_VIEWER -component OPSS -component UCSUMS
-component WLS -component STB -component ESS -component SOAINFRA -f < /
apps/stage/passwordfile${1}.txt
```

```
#infrastructure
/apps/oracle/java/jdk1.8.0_161/bin/java -jar -d64 -Djava.io.tmpdir=/apps/
oracle/temp /apps/stage/osb/fmw_12.2.1.3.0_infrastructure.jar -silent
-responseFile /apps/stage/ResponseFile${1}.xml
```

```
#Oracle Service Bus
/apps/oracle/java/jdk1.8.0_161/bin/java -jar -d64 -Djava.io.tmpdir=/apps/
oracle/temp /apps/stage/osb/fmw_12.2.1.3.0_osb.jar -silent -responseFile /
apps/stage/OSBResponseFile${1}.xml
```

```
#Oracle HTTP Server
/apps/stage/fmw_12.2.1.3.0_ohs_linux64.bin -J-Djava.io.tmpdir=/apps/oracle/
temp -silent -responseFile /apps/stage/OHSResponseFile${1}.xml
```

```
#patch OPatch naar 13.9.4
/apps/oracle/java/jdk1.8.0_161/bin/java -jar -d64 -Djava.io.tmpdir=/apps/
oracle/temp /apps/stage/6880880/opatch_generic.jar -silent oracle_home=/
apps/oracle/Middleware_${1}

#Patch WLS met 27912627
/apps/oracle/Middleware_${1}/OPatch/opatch apply -jre /apps/oracle/java/
jdk1.8.0_161 /apps/stage/27912627

# Set environment.
export MW_HOME=/apps/oracle/Middleware_${1}
export WLS_HOME=$MW_HOME/wlserver
export WL_HOME=$WLS_HOME
export JAVA_HOME=/apps/oracle/java/jdk1.8.0_161
export PATH=$JAVA_HOME/bin:$PATH
export CONFIG_JVM_ARGS=-Djava.security.egd=file:/dev/./urandom

. $WLS_HOME/server/bin/setWLSEnv.sh

case ${1} in

dev)
echo rcu installatie en domeincreatie voor development omgeving
#rcu
/apps/oracle/Middleware_${1}/oracle_common/bin/rcu -silent
-createRepository -databaseType ORACLE -connectString <database connect
string van jouw omgeving> -dbUser SYS -dbRole SYSDBA -schemaPrefix DEV
-useSamePasswordForAllSchemaUsers true -selectDependentsForComponents true
-variables SOA_PROFILE_TYPE=MED,HEALTHCARE_INTEGRATION=NO -component MDS
-component IAU -component IAU_APPEND -component IAU_VIEWER -component OPSS
-component UCSUMS -component WLS -component STB -component SOAINFRA -f < /
apps/stage/passwordfile${1}.txt


#Create the domain.
$JAVA_HOME/java -d64 -Djava.io.tmpdir=/apps/oracle/temp  weblogic.WLST /
apps/stage/wls_domain_creator${1}.py
;;

tst)
echo rcu installatie en domeincreatie voor test omgeving
#rcu
#/apps/oracle/Middleware_${1}/oracle_common/bin/rcu -silent
-createRepository -databaseType ORACLE -connectString <database connect
```

```
string van jouw omgeving> -dbUser SYS -dbRole SYSDBA -schemaPrefix TST
-useSamePasswordForAllSchemaUsers true -selectDependentsForComponents true
-variables SOA_PROFILE_TYPE=MED,HEALTHCARE_INTEGRATION=NO -component MDS
-component IAU -component IAU_APPEND -component IAU_VIEWER -component OPSS
-component UCSUMS -component WLS -component STB -component SOAINFRA -f < /
apps/stage/passwordfile${1}.txt


#Create the domain.
$JAVA_HOME/java -d64 -Djava.io.tmpdir=/apps/oracle/temp  weblogic.WLST /
apps/stage/wls_domain_creator${1}.py
;;

acc)
echo rcu installatie en domeincreatie voor acceptatie omgeving
#rcu
#/apps/oracle/Middleware_${1}/oracle_common/bin/rcu -silent
-createRepository -databaseType ORACLE -connectString
<jouwconnectstring> -dbUser SYS -dbRole SYSDBA -schemaPrefix ACC
-useSamePasswordForAllSchemaUsers true -selectDependentsForComponents true
-variables SOA_PROFILE_TYPE=MED,HEALTHCARE_INTEGRATION=NO -component MDS
-component IAU -component IAU_APPEND -component IAU_VIEWER -component OPSS
-component UCSUMS -component WLS -component STB -component SOAINFRA -f < /
apps/stage/passwordfile${1}.txt


#Create the domain.
#$JAVA_HOME/java -d64 -Djava.io.tmpdir=/apps/oracle/temp  weblogic.WLST /
apps/stage/wls_domain_creator${1}.py
;;

prd)
echo rcu installatie en domeincreatie voor productie omgeving
#rcu
#/apps/oracle/Middleware_${1}/oracle_common/bin/rcu -silent
-createRepository -databaseType ORACLE -connectString
<jouwDBconnectstring> -dbUser SYS -dbRole SYSDBA -schemaPrefix PRD
-useSamePasswordForAllSchemaUsers true -selectDependentsForComponents true
-variables SOA_PROFILE_TYPE=MED,HEALTHCARE_INTEGRATION=NO -component MDS
-component IAU -component IAU_APPEND -component IAU_VIEWER -component OPSS
-component UCSUMS -component WLS -component STB -component SOAINFRA -f < /
apps/stage/passwordfile${1}.txt
```

```
#Create the domain.
#$JAVA_HOME/java -d64 -Djava.io.tmpdir=/apps/oracle/temp  weblogic.WLST /
apps/stage/wls_domain_creator${1}.py
;;

* )
esac
```

In dit hoofdscript zit de aanroep:
/apps/oracle/java/jdk1.8.0_161/bin/java -jar -d64 -Djava.io.tmpdir=/apps/oracle/temp /apps/stage/osb/fmw_12.2.1.3.0_infrastructure.jar -silent -responseFile /apps/stage/ResponseFile${1}.xml
In het gedeelte 'ResponseFile${1}.xml' wordt de ${1} door het aanroepen van het hoofdscript met 'dev' vervangen voor 'dev'.
Er bestaat dan ook een Responsefiledev.xml:

```
[ENGINE]

#DO NOT CHANGE THIS.
Response File Version=1.0.0.0.0

[GENERIC]

#The oracle home location. This can be an existing Oracle Home or a new
Oracle Home
ORACLE_HOME=/apps/oracle/Middleware_dev

#Set this variable value to the Installation Type selected. e.g. Fusion
Middleware Infrastructure, Fusion Middleware Infrastructure With Examples.
INSTALL_TYPE=Fusion Middleware Infrastructure

#Provide the My Oracle Support Username. If you wish to ignore Oracle
Configuration Manager configuration provide empty string for user name.
MYORACLESUPPORT_USERNAME=

#Provide the My Oracle Support Password
MYORACLESUPPORT_PASSWORD=<SECURE VALUE>

#Set this to true if you wish to decline the security updates. Setting this
to true and providing empty string for My Oracle Support username will
ignore the Oracle Configuration Manager configuration
DECLINE_SECURITY_UPDATES=true
```

```
#Set this to true if My Oracle Support Password is specified
SECURITY_UPDATES_VIA_MYORACLESUPPORT=false

#Provide the Proxy Host
PROXY_HOST=

#Provide the Proxy Port
PROXY_PORT=

#Provide the Proxy Username
PROXY_USER=t

#Provide the Proxy Password
PROXY_PWD=<SECURE VALUE>

#Type String (URL format) Indicates the OCM Repeater URL which should be of
the format [scheme[Http/Https]]://[repeater host]:[repeater port]
COLLECTOR_SUPPORTHUB_URL=
```

In deze responsefile wordt als minimale en belangrijkste waarde aangegeven waar de Middleware home precies komt te staan en wat voor type installatie het is.

Wanneer deze onderdelen uitgevoerd worden vanuit het hoofdscript verschijnen er regels in het scherm die in procenten de voortgang van het proces aangeven.

Wanneer de installatie is afgerond wordt dit getoond in je console.

Voor de OSB installatie en OHS installatie worden ook responsefiles aangeleverd, deze zie je hieronder:

OSB responsefile:

```
[ENGINE]

#DO NOT CHANGE THIS.
Response File Version=1.0.0.0.0

[GENERIC]

#The oracle home location. This can be an existing Oracle Home or a new
Oracle Home
ORACLE_HOME=/apps/oracle/Middleware_dev
```

```
#Set this variable value to the Installation Type selected. e.g. Fusion
Middleware Infrastructure, Fusion Middleware Infrastructure With Examples.
INSTALL_TYPE=Service Bus

#Provide the My Oracle Support Username. If you wish to ignore Oracle
Configuration Manager configuration provide empty string for user name.
MYORACLESUPPORT_USERNAME=

#Provide the My Oracle Support Password
MYORACLESUPPORT_PASSWORD=<SECURE VALUE>

#Set this to true if you wish to decline the security updates. Setting this
to true and providing empty string for My Oracle Support username will
ignore the Oracle Configuration Manager configuration
DECLINE_SECURITY_UPDATES=true

#Set this to true if My Oracle Support Password is specified
SECURITY_UPDATES_VIA_MYORACLESUPPORT=false

#Provide the Proxy Host
PROXY_HOST=

#Provide the Proxy Port
PROXY_PORT=

#Provide the Proxy Username
PROXY_USER=

#Provide the Proxy Password
PROXY_PWD=<SECURE VALUE>

#Type String (URL format) Indicates the OCM Repeater URL which should be of
the format [scheme[Http/Https]]://[repeater host]:[repeater port]
COLLECTOR_SUPPORTHUB_URL=
```

De responsefile voor de Oracle HTTP service ziet er als volgt uit:

```
[ENGINE]

#DO NOT CHANGE THIS.
Response File Version=1.0.0.0.0

[GENERIC]
```

```
#The oracle home location. This can be an existing Oracle Home or a new
Oracle Home
ORACLE_HOME=/apps/oracle/Middleware_dev

#Set this variable value to the Installation Type selected. e.g. Fusion
Middleware Infrastructure, Fusion Middleware Infrastructure With Examples.
INSTALL_TYPE=Collocated HTTP Server (Managed through WebLogic server)

#Provide the My Oracle Support Username. If you wish to ignore Oracle
Configuration Manager configuration provide empty string for user name.
MYORACLESUPPORT_USERNAME=

#Provide the My Oracle Support Password
MYORACLESUPPORT_PASSWORD=<SECURE VALUE>

#Set this to true if you wish to decline the security updates. Setting this
to true and providing empty string for My Oracle Support username will
ignore the Oracle Configuration Manager configuration
DECLINE_SECURITY_UPDATES=true

#Set this to true if My Oracle Support Password is specified
SECURITY_UPDATES_VIA_MYORACLESUPPORT=false

#Provide the Proxy Host
PROXY_HOST=

#Provide the Proxy Port
PROXY_PORT=

#Provide the Proxy Username
PROXY_USER=

#Provide the Proxy Password
PROXY_PWD=<SECURE VALUE>

#Type String (URL format) Indicates the OCM Repeater URL which should be of
the format [scheme[Http/Https]]://[repeater host]:[repeater port]
COLLECTOR_SUPPORTHUB_URL=
```

Ook voor deze twee response files geldt dat de voornaamste input die je geeft de Middleware home is, en het type van de installatie dat je uitgevoerd wilt hebben.

Patchen

Er worden twee patches uitgevoerd. De eerste patch, nummer 6880880 patcht Opatch (de patch tool van Oracle) zelf, en wordt aangeroepen middels Java.

Vervolgens wordt de gepatchte Opatch van de Middleware home zelf gebruikt om patch 27912627 uit te voeren.

Repository installatie

Vervolgens wordt de Repository Creation Utility uit de zojuist geïnstalleerde Middleware home aangesproken om de repository in de vooraf gecreëerde database te installeren.
In deze aanroep dien je de database url van je eigen database op te geven tussen de <> tekens.

Tevens dien je een passwordfiledev.txt gemaakt te hebben met de volgende inhoud (AdminServer wachtwoord, en vervolgens 11 keer het wachtwoord voor de verschillende aan te maken schema's) Let op! Als je meer of minder onderdelen installeert in je aanroep uit het hoofdscript, moet het aantal wachtwoorden natuurlijk ook navenant aangepast worden! Het zou eleganter zijn wanneer je, net als in de handmatige installatie, met een vinkje zou kunnen aangeven dat het wachtwoord voor alle schema's gelijk mag zijn, zodat je het wachtwoord maar één keer hoeft op te geven, maar helaas bestaat deze mogelijkheid bij een silent install niet.

```
Welcome123456!
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
Welcome987
```

Het script installeert de repository, en je ziet een voortgangsindicator in procenten getoond worden.

Creatie van het domein

Na de creatie van de repository kunnen we een domein gaan aanmaken.

Dit geschiedt door de aanroep van een domeincreatiescript ‘wls_domain_creator_dev.py’, welke gevoed wordt door een domain.properties file.

In deze domain.properties file geef je aan wat je geïnstalleerd wilt hebben.

Het domain creator script is een universeel script, waar je maar weinig in hoeft aan te passen.

Je hoeft alleen het volgende naar wens te veranderen:

1. Je moet op regel 11 de naam van je properties file aanpassen naar de naam van je zelf gemaakte file.
2. Op regel 68 en verder moet je de templates aangeven die je ingeladen wilt hebben. In dit voorbeeld wordt de standaard Weblogic template ingeladen, de Enterprise Manager template en de OSB template. Maar als jij behoefte hebt aan bijvoorbeeld de jax-rpc template, dan dien je die toe te voegen tussen de laatste template en het loadTemplates() statement.
3. Op regel 267 een nieuwe unieke naam voor je logfile opgeven.

Zo ziet het script eruit:

```
import sys
import os
from os.path import exists
from sys import argv


def get_script_path():
return os.path.dirname(os.path.realpath(sys.argv[0]))

def parsefile():
propfile = get_script_path()+”/domain.propertieststst”
if exists(propfile):
global fo
fo = open(propfile, ‘r+’)
lines = fo.readlines()
for line in lines:
```

```
#print line.rstrip()
if “=” in line:
line = line.rstrip()
key = line.split(‘=’)[0]
value = line.split(‘=’)[1]
_dict[key]=value


def printdomain():
print ‘-----------------------------’
print “Properties Information”
print ‘-----------------------------’
for key, val in _dict.iteritems():
print key,”=>”,val

def export_properties():
global _dict
global mwhome
global wlshome
global domainroot
global approot
global domainName
global domain_username
global domain_password
global adminPort
global adminAddress
global adminPortSSL
global adminMachine
global machines
global servers
global clusters
mwhome = _dict.get(‘mwhome’)
wlshome = _dict.get(‘wlshome’)
domainroot = _dict.get(‘domainroot’)
approot = _dict.get(‘approot’)
domainName = _dict.get(‘domain_name’)
domain_username = _dict.get(‘domain_username’)
domain_password = _dict.get(‘domain_password’)

adminPort = _dict.get(“admin.port”)
adminAddress = _dict.get(“admin.address”)
adminPortSSL = _dict.get(“admin.port.ssl”)
#adminMachine = _dict.get(“admin.machine”)
```

```
machines = _dict.get("machines").split(',')
servers = _dict.get("managedservers").split(',')
clusters = _dict.get("clusters").split(',')

def read_template():
# Load the template. Versions < 12.2
try:
selectTemplate('Basic WebLogic Server Domain','12.2.1.3')
selectTemplate('Oracle Enterprise Manager')
selectTemplate('Oracle Service Bus')
loadTemplates()

except:
print "Error Reading the Template",wlshome
print "Dumpstack: \n -------------- \n",dumpStack()
sys.exit(2)

def create_machine():
try:
cd('/')
for machine in machines:
print "Creating a New Machine with the following Configuration"

mn = create(machine,'Machine')
machine_name=_dict.get(machine+'.Name')
if (machine_name != ""):
print "\tMachine Name",machine_name
mn.setName(machine_name)
else:
print "No machine Name mentioned for",machine
except:
print "Creating Machine failed",machine
print "Dumpstack: \n -------------- \n",dumpStack()
sys.exit(2)

def create_admin():
try:
print "\nCreating AdminServer with the following Configuration"
cd('/Security/base_domain/User/' + domain_username)
cmo.setPassword(domain_password)
cd('/Server/AdminServer')
cmo.setName('AdminServer')
cmo.setListenPort(int(adminPort))
cmo.setListenAddress(adminAddress)
```

```
print "\tAdminServer ListenPort:",adminPort
print "\tAdminServer Listenaddress:",adminAddress
print "\tAdminServer SSLListenPort:",adminPortSSL

create('AdminServer','SSL')
cd('SSL/AdminServer')
set('Enabled', 'True')
set('ListenPort', int(adminPortSSL))

except:
print "Error while creating AdminServer"
print "Dumpstack: \n -------------- \n",dumpStack()

def create_managedserver():
try:
cd ('/')
for server in servers:
MSN = _dict.get(server+'.Name')
MSP = _dict.get(server+'.port')
MSA = _dict.get(server+'.address')
MSM = _dict.get(server+'.machine')
print "\nCreating A New Managed Server with following Configuration"
print "\tServerName:",MSN
print "\tServer ListenPort:",MSP
print "\tServer ListenAddress:",MSA
sobj = create(MSN,'Server')
sobj.setName(MSN)
sobj.setListenPort(int(MSP))
sobj.setListenAddress(MSA)
sobj.setMachine(MSM)
except:
print "Error While Creating ManagedServer",server
print "Dumpstack: \n -------------- \n",dumpStack()


def create_clusters():
try:
cd ('/')
for cluster in clusters:
CN = _dict.get(cluster+'.Name')
cobj = create(CN,'Cluster')
print "\nCreating a New Cluster with the following Configuration"
print "\tClusterName",CN
except:
```

```
print “Error while Creating Cluster”,cluster
print “Dumpstack: \n -------------- \n”,dumpStack()
sys.exit(2)

def commit_writedomain():
try:
# If the domain already exists, overwrite the domain
setOption(‘OverwriteDomain’, ‘true’)

setOption(‘ServerStartMode’,’prod’)
setOption(‘AppDir’, approot + ‘/’ + domainName)

writeDomain(domainroot + ‘/’ + domainName)
closeTemplate()

except:
print “ERROR: commit_writedomain Failed”
print “Dumpstack: \n -------------- \n”,dumpStack()

def print_withformat(title):
print “\n----------------------------------------------------\n”,title,”
\n----------------------------------------------------”

def print_somelines():
print “----------------------------------------------------”

def print_domainsummary():
print “DomainName:”,domainName
print “DomainUserName:”,domain_username
print “DomainPassword: ***************”
print “DomainDirectory:”,domainroot
print “ApplicationRoot:”,approot

def start_AdminServer():
try:
global managementurl
managementurl = “t3://”+adminAddress+”:”+adminPort
global AdminServerDir
AdminServerDir = domainroot+”/”+domainName+”/servers/AdminServer”
global AdminServerLogDir
AdminServerLog = AdminServerDir+”/logs/AdminServer.out”
global DomainDir
DomainDir = domainroot+”/”+domainName
```

```
print_somelines()
print "\nStarting Server with following Params"
print_somelines()
print "DomainDir",DomainDir
print "managementurl",managementurl
print_somelines()

print "\nRedirecting Startup Logs to",AdminServerLog
startServer('AdminServer',domainName,managementurl,domain_username,domain_
password,DomainDir,'true',120000,AdminServerLog)

print "AdminServer has been successfully Started"
except:
print "ERROR: Unable to Start AdminServer"
print "Dumpstack: \n -------------- \n",dumpStack()

def stop_AdminServer():

try:
global managementurl
managementurl = "t3://"+adminAddress+":"+adminPort
global AdminServerDir
AdminServerDir = domainroot+"/"+domainName+"/servers/AdminServer"
global AdminServerLogDir
AdminServerLog = AdminServerDir+"/logs/AdminServer.out"
global DomainDir
DomainDir = domainroot+"/"+domainName

print_somelines()
print "\nStarting Server with following Params"
print_somelines()
print "DomainDir",DomainDir
print "managementurl",managementurl
print_somelines()

print "\nRedirecting Startup Logs to",AdminServerLog
startServer('AdminServer',domainName,managementurl,domain_username,domain_
password,DomainDir,'true',120000)

print "AdminServer has been successfully Started"
except:
print "ERROR: Unable to Start AdminServer"
print "Dumpstack: \n -------------- \n",dumpStack()
```

```
def stop_AdminServer():
try:
print_somelines()
print "\n Shutting Down AdminServer"
print_somelines()

shutdown('AdminServer','Server','true',1200)
print "AdminServer has been successfully SHUTDOWN"
except:
print "ERROR: Unable to SHUTDOWN AdminServer"
print "Dumpstack: \n -------------- \n",dumpStack()

def connect_online():
try:
global managementurl
managementurl = "t3://"+adminAddress+":"+adminPort
print "\nConnecting to AdminServer with managementurl",managementurl
connect(domain_username,domain_password,managementurl)
print "\nSuccessfully Connected to AdminServer!!."

except:
print "ERROR: Unable to Connect to AdminServer"
sys.exit(2)

def acquire_edit_session():
edit()
startEdit()

def save_activate_session():
save()
activate()

def Enable_wlst_log_redirection():
#wlst output redirect to a logfile
redirect('./wlst_executiontst.log','false')

def Stop_wlst_log_redirection():
stopRedirect()
def map_machines():
#try:
acquire_edit_session()
for machine in machines:
print "Starting to map resources to the machine ",machine
instances = _dict.get(machine+".instances")
```

```
#print "INST",instances
if len(instances) > 1:
instances = instances.split(',')
for instance in instances:
if instance == "admin":
instname = "AdminServer"
else:
instname = _dict.get(instance+".Name")
#print "What is the instname",instname
cd ('/Servers/'+instname)
#print "WHARE AM I",pwd()
machine_name=_dict.get(machine+'.Name')
mbean_name='/Machines/'+machine_name
#print "What is Machine MBEAN",mbean_name
cmo.setMachine(getMBean(mbean_name))
else:
instname = _dict.get(instances+".Name")
#print "What is the instname",instname
cd ('/Servers/'+instname)
#print "WHARE AM I",pwd()
machine_name=_dict.get(machine+'.Name')
mbean_name='/Machines/'+machine_name
cmo.setMachine(getMBean(mbean_name))
save_activate_session()
```

```
def map_clusters():
#try:
acquire_edit_session()
for cluster in clusters:
print "\nStarting to map resources to the cluster ",cluster
members = _dict.get(cluster+".members")
#print "members",members
if len(members) > 1:
members = members.split(',')
for member in members:
if member == "admin":
membername = "AdminServer"
else:
membername = _dict.get(member+".Name")
#print "What is the memberName",membername
cd ('/Servers/'+membername)
#print "WHARE AM I",pwd()
cluster_name=_dict.get(cluster+'.Name')
mbean_name='/Clusters/'+cluster_name
```

```
#print "What is Cluster MBEAN",mbean_name
cmo.setCluster(getMBean(mbean_name))
else:
membername = _dict.get(member+".Name")
#print "What is the memberName",membername
cd ('/Servers/'+membername)
#print "WHARE AM I",pwd()
cluster_name=_dict.get(cluster+'.Name')
mbean_name='../../Clusters/'+cluster_name
cmo.setCluster(getMBean(mbean_name))
save_activate_session()
#except:
#print "Machine Creation Failed"

if __name__ != "__main__":
_dict={};
Enable_wlst_log_redirection()
print "Start of the script Execution >>"
print "Parsing the properties file..."
parsefile()
print "Exporting the Properties to variables.."
export_properties()
print "Creating Domain from Domain Template..."
read_template()
print_withformat("Creating Machines")
create_machine()
print_somelines()
print_withformat("Creating AdminServer")
create_admin()
print_somelines()
print_withformat("Creating ManagedServers")
create_managedserver()
print_somelines()
print_withformat("Creating Clusters")
create_clusters()
print_somelines()
print "\nCommit and Saving the Domain"
commit_writedomain()
print_withformat("Domain Summary")
print_domainsummary()
print_somelines()
print("Starting the AdminServer")
start_AdminServer()
connect_online()
```

```
map_machines()
print “Done Mapping the Machines”
map_clusters()
print “Done Mapping the Cluster!”
print “End of Script Execution << \nGood Bye!”
Stop_wlst_log_redirection()
sys.exit(0)

if __name__ == “__main__”:
print “This script has to be executed with weblogic WLST”
sys.exit(1)
```

Zo ziet de properties file eruit:

```
# Paths
mwhome=/apps/oracle/Middleware_dev
wlshome=/apps/oracle/Middleware_dev/wlserver
domainroot=/apps/oracle/Middleware_dev/user_projects/domains
approot=/apps/oracle/Middleware_dev/user_projects/applications


# Credentials
domain_name=<jouw domeinnaam>
domain_username=weblogic
domain_password=Welcome123456!


# Admin Server
admin.port=5001
admin.address=<jouwhostnaam.jouwbedrijfsnaam.nl>
admin.port.ssl=5441


#Managed Server Definition
# Add more ms based on your need
# for every ms(server) you are adding you should also specify the
properties like ms[n].port etc
managedservers=ms1

ms1.Name=osb_server1
ms1.port=5010
ms1.address=<jouwhostnaam.jouwbedrijfsnaam.nl>
```

```
# Cluster Definition
clusters=c1

c1.Name=<jouwclusternaam>

c1.members=ms1

# Define Machines
machines=m1

m1.Name=<jouwHostnaam>
m1.instances=ms1,admin
```

Nabewerkingen

Wanneer de installatie van de omgeving is afgerond dienen nog wel de volgende stappen uitgevoerd worden:

1. Starten AdminServer
2. Aanpassen wachtwoord Nodemanager in AdminServer
3. Draaien Nodemanager ‘discoverer script’
4. boot.propertes file met wachtwoord Adminserver maken
5. Oracle HTTP configuratie

Conclusie

Het is mogelijk om met goede scripts de complete installatie, inclusief domeincreatie, herhaalbaar en foutloos uit te voeren. Hierdoor krijg je een OTAP-straat die er overal hetzelfde uit ziet. Als er iets verandert in de eisen aan de servers, dan passen we het in deze scripts aan, en weten we zeker dat het in het vervolg altijd zo toegepast zal worden.

Waarom op deze manier?

Bij een handmatig uitgevoerde domeincreatie moet je vele inrichtingskeuzes maken, vaak worden die niet altijd even doordacht genomen, omdat de omgeving snel klaar moet zijn. Zo kan gemakkelijk willekeur ontstaan, met verschillend ingerichte omgevingen tot gevolg. Met deze methode is er één keer goed over nagedacht, zijn netjes naamgevingsconventies en inrichtingskeuzes toegepast en wordt het vervolgens altijd hetzelfde gedaan.

Valkuilen

Pas op met het gebruiken van scripts van het internet dat ze bedoeld zijn voor de versie die je wilt installeren. De 11g en 12c structuur ontloopt elkaar niet veel, maar er zijn wel een aantal directories die nog wel bestaan in 12c (waarschijnlijk vanwege backward compatibility), maar niet meer gebruikt worden (in 11g werd veel vanuit de DOMAIN_HOME subdirectories gedaan, in 12c gebeurt dit allemaal vanuit de WL_HOME en ORACLE_COMMON directories). Als je een verouderde directory aanroept, dan stopt je script, maar je hebt geen idee waarom. Dat merk je pas als je de aanroepen separaat gaat uitvoeren.

Voor- en nadelen in vergelijking met Myst

Nadelen zijn dat Myst niet altijd de nieuwste versies kan provisionen en het kost geld. Als je specifieke omgevingswensen hebt, moet je ook hier zelf de voor jou specifieke templates opzetten. Je kunt daarbij wel gebruik maken van een wizard die je door de keuzes heen helpt.

Voordelen zijn dat ze een aantal standaard templates aanbieden om je provisioning mee te doen. Als die jou voldoende bieden ben je snel klaar. Daarnaast wordt er bij het aanmaken van een nieuwe omgeving ook meteen behoorlijk uitgebreide documentatie aangemaakt, en dat is wel een grote pré.

Kun je nu nog meer doen?

Ja, je kunt door het overkoepelende script uit te breiden ook de benodigde certificaten geautomatiseerd inlezen, de HTTP server automatisch in laten richten et cetera. Ik sluit niet uit dat als we dat ook gaan automatiseren daar een blog over volgt, waarnaar ik zal verwijzen vanuit deze Whitebook.

Bronnen

Goede website over het werken met silent install en responsefiles:
http://www.middlewarestuff.com/2015/10/silent-installation-of-oracle-soa-suite.html

Goede website over het werken met scripts voor domeincreatie (werkt standaard niet helemaal goed met 12c, maar legt de opzet goed uit):
https://oracle-base.com/articles/web/wlst-create-domain

Oracle documentatie over WLST:
https://docs.oracle.com/middleware/1221/wls/WLSTG/domains.htm#WLSTG156

Oracle documentatie over Templates:
https://docs.oracle.com/middleware/1221/wls/WLDTR/templates.htm#WLDTR110

Template opbouw en provisioning met Myst:
https://www.youtube.com/watch?time_continue=9&v=QKKRGSoXkjQ