#!/bin/bash

######SOA Domain Environment Configuration Script

# Prepared for 2-nodes installation (execute on both nodes)

# 1- Creates Linux aliases for ease of administration

# 2- Creates Weblogic domain administration scripts

# 3- Creates directory structure necessary for the Weblogic Domain.

# 4- Creates private & public ssl keys and certificates used by Weblogic Server for inter process communication.

# 5- Creates VirtualHost configuration files for OHS

# 6- Creates NodeManager configuration file

##### SET VARIABLES BEFORE EXECUTE

export COMPDOMAIN=mydom.com

export SERVER1HOST=testserver1

export SERVER2HOST=testserver2

export FRONTENDURL=soa.mydom.com

export USERNAME=oracle

export USERHOME=/home/${USERNAME}

export SCRIPTSHOME=${USERHOME}/scripts

export JAVA\_HOME=/opt/11.1.1.7/jrockit

export MW\_HOME=/opt/11.1.1.7/middleware

export WTIER\_HOME=${MW\_HOME}/Oracle\_WT1

export WL\_HOME=${MW\_HOME}/wlserver\_10.3

export SOA\_HOME=${MW\_HOME}/Oracle\_SOA1

export DOMAIN\_NAME=soa\_domain

export ADOMAIN\_HOME=/opt/11.1.1.7/admin/domains/${DOMAIN\_NAME}/aserver/${DOMAIN\_NAME}

export MDOMAIN\_HOME=/opt/11.1.1.7/admin/domains/${DOMAIN\_NAME}/mserver/${DOMAIN\_NAME}

export INSTANCE\_HOME=/opt/11.1.1.7/admin/instances/instance1

export SOA\_PORT=8001

export WSM\_PORT=7003

export BAM\_PORT=9001

export NM\_USER=weblogic

export NM\_PWD=xxxxx

export NM\_HOST=`hostname` # do not modify

export NM\_PORT=5556

export NM\_NETMASK='255.255.255.0'

export NM\_INTERFACE='eth0'

##### SSL Keys configuration

#

export ADMINHOST=adminsoa-prod

export ADMINIP=192.168.1.103

export SOAVIP1=soaprod-vip1

export SOAVIP2=soaprod-vip2

export BAMVIP1=bamprod-vip1

export KEYSTORE\_PWD=xxxx

##### END OF VARIABLES

#############################

# Directory structure

#############################

# Directorios locales

echo "Creating local folders..."

mkdir ${SCRIPTSHOME}/

mkdir -p ${MDOMAIN\_HOME}

mkdir ${MDOMAIN\_HOME}/../applications

# Shared disk must be already mounted

if [ ! -d ${ADOMAIN\_HOME} ]; then

 echo "Creating shared folders"

 mkdir -p ${ADOMAIN\_HOME}

 mkdir -p ${ADOMAIN\_HOME}/../certs

 mkdir -p ${ADOMAIN\_HOME}/../SOA\_CLUSTER/jms

 mkdir -p ${ADOMAIN\_HOME}/../BAM\_CLUSTER/jms

 mkdir -p ${ADOMAIN\_HOME}/../WSM\_CLUSTER/jms

 mkdir -p ${ADOMAIN\_HOME}/../SOA\_CLUSTER/tlogs

 mkdir -p ${ADOMAIN\_HOME}/../BAM\_CLUSTER/tlogs

 mkdir -p ${ADOMAIN\_HOME}/../WSM\_CLUSTER/tlogs

 mkdir -p ${ADOMAIN\_HOME}/../SOA\_CLUSTER/dp

 mkdir -p ${ADOMAIN\_HOME}/../SOA\_CLUSTER/fadapter

else

 echo "Folders already exist"

fi

###############################

# ALIASES

###############################

echo "Creating Aliases..."

echo "export MW\_HOME=${MW\_HOME}

export JAVA\_HOME=${JAVA\_HOME}

export PATH=$JAVA\_HOME/bin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin

cd \$MW\_HOME" > ${USERHOME}/.mwhome

echo "export MW\_HOME=${MW\_HOME}

export JAVA\_HOME=${JAVA\_HOME}

export WL\_HOME=${WL\_HOME}

export PATH=${JAVA\_HOME}/bin:${WL\_HOME}/common/bin:${WL\_HOME}/server/bin:${WL\_HOME}/common/nodemanager:/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin

cd \$WL\_HOME" > ${USERHOME}/.wlogic

echo "export MW\_HOME=${MW\_HOME}

export JAVA\_HOME=${JAVA\_HOME}

export ORACLE\_HOME=${SOA\_HOME}

export PATH=${JAVA\_HOME}/bin:${SOA\_HOME}/common/bin:${SOA\_HOME}/bin:${SOA\_HOME}/OPatch:/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin

cd \$ORACLE\_HOME" > ${USERHOME}/.soahome

echo "export MW\_HOME=${MW\_HOME}

export JAVA\_HOME=${JAVA\_HOME}

export ORACLE\_HOME=${WTIER\_HOME}

export PATH=${JAVA\_HOME}/bin:${WTIER\_HOME}/bin:${WTIER\_HOME}/OPatch:/sbin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin

cd \$ORACLE\_HOME" > ${USERHOME}/.wtier

echo "export DOMAIN\_HOME=${ADOMAIN\_HOME}

export PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:${ADOMAIN\_HOME}/bin

cd \$DOMAIN\_HOME" > ${USERHOME}/.soadom

echo "export DOMAIN\_HOME=${MDOMAIN\_HOME}

export PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:${MDOMAIN\_HOME}/bin:${WL\_HOME}/common/bin:${WL\_HOME}/common/nodemanager

cd \$DOMAIN\_HOME" > ${USERHOME}/.msoadom

echo "export INSTANCE\_HOME=${INSTANCE\_HOME}

export PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:${INSTANCE\_HOME}/bin

cd \$INSTANCE\_HOME" > ${USERHOME}/.soainst

##### Add aliases to .bashrc file

echo "alias soahome='. ${USERHOME}/.soahome'

alias mwhome='. ${USERHOME}/.mwhome'

alias wlogic='. ${USERHOME}/.wlogic'

alias wtier='. ${USERHOME}/.wtier'

alias soadom='. ${USERHOME}/.soadom'

alias msoadom='. ${USERHOME}/.msoadom'

alias ohs='. ${USERHOME}/.soainst'

export PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:${MDOMAIN\_HOME}/bin/server\_migration:${WL\_HOME}/common/bin:${WL\_HOME}/common/nodemanager" >> ${USERHOME}/.bashrc

###################################

# Weblogic Domain Operation Scripts

###################################

echo "Creando scripts de operacion"

cd ${SCRIPTSHOME}

echo "count=\$(ping -c 4 ${ADMINHOST}.${COMPDOMAIN} | grep 'received' | awk -F',' '{ print \$2 }' | awk '{ print \$1 }')

if [ \$count -eq 0 ]; then

 # 100% failed

 echo 'Host: ${ADMINHOST}.${COMPDOMAIN} is down'

 echo 'Bringing VIP up'

 sudo /sbin/ifconfig ${NM\_INTERFACE}:1 ${ADMINIP} netmask ${NM\_NETMASK}

 sudo /sbin/arping -q -U -c 3 -I ${NM\_INTERFACE} ${ADMINIP}

else

 echo 'WARNING! Host ${ADMINHOST}.${COMPDOMAIN} is already UP'

fi" > adminVipUP.sh

echo "${WL\_HOME}/common/bin/wlst.sh <<EOF

startNodeManager(verbose='true', NodeManagerHome='${WL\_HOME}/common/nodemanager', ListenPort='${NM\_PORT}', ListenAddress='${NM\_HOST}')

exit()

EOF" > startNodeManager.sh

echo "${WL\_HOME}/common/bin/wlst.sh <<EOF

nmConnect('${NM\_USER}','${NM\_PWD}','${NM\_HOST}','${NM\_PORT}','${DOMAIN\_NAME}','${ADOMAIN\_HOME}')

nmStart('AdminServer')

exit()

EOF" > startWeblogic.sh

echo "SRV=\$1

${WL\_HOME}/common/bin/wlst.sh <<EOF

connect('${NM\_USER}','${NM\_PWD}','t3://${ADMINHOST}.${COMPDOMAIN}:7001')

start('\${SRV}','Server')

exit()

EOF" > startManagedServer.sh

echo "

. ${USERHOME}/scripts/startWeblogic.sh

. ${USERHOME}/scripts/startManagedServer.sh WLS\_WSM2 &

. ${USERHOME}/scripts/startManagedServer.sh WLS\_WSM1

. ${USERHOME}/scripts/startManagedServer.sh WLS\_SOA2 &

. ${USERHOME}/scripts/startManagedServer.sh WLS\_SOA1

. ${USERHOME}/scripts/startManagedServer.sh WLS\_BAM2 &

. ${USERHOME}/scripts/startManagedServer.sh WLS\_BAM1" > start\_soa.sh

echo "${WL\_HOME}/common/bin/wlst.sh <<EOF

connect('${NM\_USER}','${NM\_PWD}','t3://${ADMINHOST}.${COMPDOMAIN}:7001')

shutdown(force='true')

exit()

EOF" > stopWeblogic.sh

echo "${WL\_HOME}/common/bin/wlst.sh <<EOF

nmConnect('${NM\_USER}','${NM\_PWD}','${NM\_HOST}','${NM\_PORT}','${DOMAIN\_NAME}','${ADOMAIN\_HOME}')

stopNodeManager();

exit()

EOF" > stopNodeManager.sh

echo "SRV=\$1

${WL\_HOME}/common/bin/wlst.sh <<EOF

connect('${NM\_USER}','${NM\_PWD}','t3://${ADMINHOST}.${COMPDOMAIN}:7001')

shutdown('\${SRV}','Server','true',1000,block='false')

exit()

EOF" > stopManagedServer.sh

echo ". ${USERHOME}/scripts/stopManagedServer.sh WLS\_BAM2 &

. ${USERHOME}/scripts/stopManagedServer.sh WLS\_BAM1

. ${USERHOME}/scripts/stopManagedServer.sh WLS\_SOA2 &

. ${USERHOME}/scripts/stopManagedServer.sh WLS\_SOA1

. ${USERHOME}/scripts/stopManagedServer.sh WLS\_WSM2 &

. ${USERHOME}/scripts/stopManagedServer.sh WLS\_WSM1

. ${USERHOME}/scripts/stopWeblogic.sh

. ${USERHOME}/scripts/stopNodeManager.sh" > stop\_soa.sh

chmod +x \*.sh

###################################

# SSL Keys and certs

###################################

echo "Creating Identity and Trust Keystores..."

if [ ! -f ${ADOMAIN\_HOME}/../certs/appIdentityKeystore.jks ]

then

 echo "Creating SSL certs and Identity Keystore..."

 cd ${ADOMAIN\_HOME}/../certs

 . ${WL\_HOME}/server/bin/setWLSEnv.sh

 java utils.CertGen ${KEYSTORE\_PWD} ${ADMINHOST}\_cert ${ADMINHOST}\_key domestic ${ADMINHOST}.${COMPDOMAIN}

 java utils.CertGen ${KEYSTORE\_PWD} ${SERVER1HOST}\_cert ${SERVER1HOST}\_key domestic ${SERVER1HOST}.${COMPDOMAIN}

 java utils.CertGen ${KEYSTORE\_PWD} ${SERVER2HOST}\_cert ${SERVER2HOST}\_key domestic ${SERVER2HOST}.${COMPDOMAIN}

 java utils.CertGen ${KEYSTORE\_PWD} ${SOAVIP1}\_cert ${SOAVIP1}\_key domestic ${SOAVIP1}.${COMPDOMAIN}

 java utils.CertGen ${KEYSTORE\_PWD} ${SOAVIP2}\_cert ${SOAVIP2}\_key domestic ${SOAVIP2}.${COMPDOMAIN}

 java utils.CertGen ${KEYSTORE\_PWD} ${BAMVIP1}\_cert ${BAMVIP1}\_key domestic ${BAMVIP1}.${COMPDOMAIN}

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity001 ${KEYSTORE\_PWD} ./${ADMINHOST}\_cert.pem ./${ADMINHOST}\_key.pem

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity002 ${KEYSTORE\_PWD} ./${SERVER1HOST}\_cert.pem ./${SERVER1HOST}\_key.pem

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity003 ${KEYSTORE\_PWD} ./${SERVER2HOST}\_cert.pem ./${SERVER2HOST}\_key.pem

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity004 ${KEYSTORE\_PWD} ./${SOAVIP1}\_cert.pem ./${SOAVIP1}\_key.pem

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity005 ${KEYSTORE\_PWD} ./${SOAVIP2}\_cert.pem ./${SOAVIP2}\_key.pem

 java utils.ImportPrivateKey appIdentityKeystore.jks ${KEYSTORE\_PWD} appIdentity006 ${KEYSTORE\_PWD} ./${BAMVIP1}\_cert.pem ./${BAMVIP1}\_key.pem

else

 echo "Identity Keystore already exists"

fi

if [ ! -f ${ADOMAIN\_HOME}/../certs/appTrustKeyStore.jks ]

then

 echo "Creating Trust Keystore..."

 cp $WL\_HOME/server/lib/cacerts ./appTrustKeyStore.jks

 keytool -storepasswd -new ${KEYSTORE\_PWD} -keystore appTrustKeyStore.jks -storepass changeit

 keytool -import -v -noprompt -trustcacerts -alias clientCaCert -file $WL\_HOME/server/lib/CertGenCA.der -keystore appTrustKeyStore.jks -storepass ${KEYSTORE\_PWD}

else

 echo "Trust Keystore already exist"

fi

##########################

# Extract B2B engine

##########################

echo "Extracting B2B engine..."

cd ${SOA\_HOME}/soa/thirdparty/edifecs/

tar -zxf XEngine.tar.gz

##############################################

# OHS VirtualHosts file

##############################################

# staticports.ini

cd ${USERHOME}

echo "[OHS]

#Listen port for OHS component

OHS Port = 7777

[OPMN]

#Process Manager Local port no

OPMN Local Port = 1880" > staticports.ini

# soa\_vh.conf

echo "<VirtualHost \*:7777>

 ServerName https://${FRONTENDURL}:443

 ServerAdmin you@your.address

 RewriteEngine On

 RewriteOptions inherit

<Location /soa-infra>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# SOA inspection.wsil

<Location /inspection.wsil>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# Worklist

<Location /integration>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# B2B

<Location /b2bconsole>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

<Location /b2b/services/ws/>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# UMS prefs

<Location /sdpmessaging/userprefs-ui>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# Default to-do taskflow

<Location /DefaultToDoTaskFlow>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# Workflow

<Location /workflow>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

#Required if attachments are added for workflow tasks

 <Location /ADFAttachmentHelper>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

 </Location>

# SOA composer application

 <Location /soa/composer>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

 <Location /frevvo>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# BPM

<Location /bpm/composer>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

<Location /bpm/workspace>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# BAM Web Application

<Location /OracleBAM >

 SetHandler weblogic-handler

 WebLogicCluster ${BAMVIP1}:${BAM\_PORT},${SERVER2HOST}:${BAM\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

<Location /OracleBAMWS >

 SetHandler weblogic-handler

 WebLogicCluster ${BAMVIP1}:${BAM\_PORT},${SERVER2HOST}:${BAM\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

# WSM-PM

<Location /wsm-pm>

 SetHandler weblogic-handler

 WebLogicCluster ${SERVER1HOST}:${WSM\_PORT},${SERVER2HOST}:${WSM\_PORT}

 WLProxySSL ON

 WLProxySSLPassThrough ON

</Location>

</VirtualHost>" > soa\_vh443.conf

# soa\_vh.conf

echo "<VirtualHost \*:7777>

 ServerName http://${FRONTENDURL}:80

 ServerAdmin you@your.address

 RewriteEngine On

 RewriteOptions inherit

<Location /soa-infra>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# SOA inspection.wsil

<Location /inspection.wsil>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# Worklist

<Location /integration>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# B2B

<Location /b2bconsole>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

<Location /b2b/services/ws/>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# UMS prefs

<Location /sdpmessaging/userprefs-ui>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# Default to-do taskflow

<Location /DefaultToDoTaskFlow>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# Workflow

<Location /workflow>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

#Required if attachments are added for workflow tasks

 <Location /ADFAttachmentHelper>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

 </Location>

# SOA composer application

 <Location /soa/composer>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

 <Location /frevvo>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# BPM

<Location /bpm/composer>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

<Location /bpm/workspace>

 SetHandler weblogic-handler

 WebLogicCluster ${SOAVIP1}:${SOA\_PORT},${SOAVIP2}:${SOA\_PORT}

</Location>

# BAM Web Application

<Location /OracleBAM >

 SetHandler weblogic-handler

 WebLogicCluster ${BAMVIP1}:${BAM\_PORT},${SERVER2HOST}:${BAM\_PORT}

</Location>

<Location /OracleBAMWS >

 SetHandler weblogic-handler

 WebLogicCluster ${BAMVIP1}:${BAM\_PORT},${SERVER2HOST}:${BAM\_PORT}

</Location>

# WSM-PM

<Location /wsm-pm>

 SetHandler weblogic-handler

 WebLogicCluster ${SERVER1HOST}:${WSM\_PORT},${SERVER2HOST}:${WSM\_PORT}

</Location>

</VirtualHost>" > soa\_vh80.conf

###########################

# Configure NodeManager

###########################

echo "Configurando nodemanager.properties..."

if [ ${NM\_HOST} = ${SERVER1HOST}.${COMPDOMAIN} ]

then

 echo "Host: ${NM\_HOST}"

 export APPIDENTITY=appIdentity002

else

 echo "Host: ${NM\_HOST}"

 export APPIDENTITY=appIdentity003

fi

echo "DomainsFile=${WL\_HOME}/common/nodemanager/nodemanager.domains

LogLimit=0

PropertiesVersion=10.3

DomainsDirRemoteSharingEnabled=false

javaHome=${JAVA\_HOME}

AuthenticationEnabled=true

NodeManagerHome=${WL\_HOME}/common/nodemanager

JavaHome=${JAVA\_HOME}/jre

LogLevel=INFO

DomainsFileEnabled=true

StartScriptName=startWebLogic.sh

ListenAddress=${NM\_HOST}

NativeVersionEnabled=true

ListenPort=${NM\_PORT}

LogToStderr=true

SecureListener=true

LogCount=1

DomainRegistrationEnabled=false

StopScriptEnabled=true

QuitEnabled=false

LogAppend=true

StateCheckInterval=500

CrashRecoveryEnabled=false

StartScriptEnabled=true

LogFile=${WL\_HOME}/common/nodemanager/nodemanager.log

LogFormatter=weblogic.nodemanager.server.LogFormatter

ListenBacklog=50

Interface=${NM\_INTERFACE}

NetMask=${NM\_NETMASK}

UseMACBroadcast=true

KeyStores=CustomIdentityAndCustomTrust

CustomIdentityKeyStoreFileName=${ADOMAIN\_HOME}/../certs/appIdentityKeystore.jks

CustomIdentityKeyStorePassPhrase=${KEYSTORE\_PWD}

CustomIdentityAlias=${APPIDENTITY}

CustomIdentityPrivateKeyPassPhrase=${KEYSTORE\_PWD}" > ${WL\_HOME}/common/nodemanager/nodemanager.properties