EUS, Kerberos, SSL and OUD a guideline

Demo Scripts, Examples and Exercises

Stefan Oehrli

2019 October 16

# Demos EUS, Kerberos, SSL and OUD a guideline

A couple of demo’s for the TechEvent presentation *EUS, Kerberos, SSL and OUD a guideline*. Be aware, that the code can not be used copy/past in all environments due to limitations on the line breaks.

Demos are shown on an Oracle 18c Docker based database.

docker run --detach --name te2018\_eusdb \  
 --volume /data/docker/volumes/te2018\_eusdb:/u01 \  
 -e ORACLE\_SID=TE18EUS \  
 -p 1521:1521 -p 5500:5500 \  
 --hostname te2018\_eusdb.postgasse.org \  
 --dns 192.168.56.70 \  
 --dns-search postgasse.org \  
 oracle/database:18.3.0.0

Create user and roles

CREATE ROLE tvd\_connect;  
GRANT CREATE SESSION TO tvd\_connect;  
GRANT select ON v\_$session TO tvd\_connect;  
CREATE USER SOE\_KERBEROS IDENTIFIED EXTERNALLY AS 'soe@POSTGASSE.ORG';  
GRANT tvd\_connect TO SOE\_KERBEROS;

## Password Verifier

Clean up and remove the old users.

DROP USER user\_10g;  
DROP USER user\_11g;  
DROP USER user\_12c;  
DROP USER user\_all;

Create 4 dedicated test user and grant them *CREATE SESSION*.

GRANT CREATE SESSION TO user\_10g IDENTIFIED BY manager;  
GRANT CREATE SESSION TO user\_11g IDENTIFIED BY manager;  
GRANT CREATE SESSION TO user\_12c IDENTIFIED BY manager;  
GRANT CREATE SESSION TO user\_all IDENTIFIED BY manager;

Reset all passwords using *IDENTIFIED BY VALUES* to explicitly set a particular password verifier.

ALTER USER user\_10g IDENTIFIED BY VALUES '808E79166793CFD1';  
ALTER USER user\_11g IDENTIFIED BY VALUES 'S:22D8239017006EBDE054  
 108BF367F225B5E731D12C91A3BEB31FA28D4A38';  
ALTER USER user\_12c IDENTIFIED BY VALUES 'T:C6CE7A88CC5D0E048F32  
 A564D2B6A7BDC78A2092184F28D13A90FC071F804E5EA09D4D2A3749AA79  
 BFD0A90D18DEC5788D2B8754AE20EE5C309DBA87550E8AA15EAF2746ED43  
 1BF4543D2ABE33E22678';

See what we do have in *dba\_users*.

set linesize 160 pagesize 200  
col username for a25  
SELECT username,password\_versions FROM dba\_users WHERE username LIKE 'USER\_%' ORDER BY 1;  
  
USERNAME PASSWORD\_VERSIONS  
------------------------- -----------------  
USER\_10G 10G  
USER\_11G 11G  
USER\_12C 12C  
USER\_ALL 10G 11G 12C

See what we do have in *user$*.

set linesize 160 pagesize 200  
col name for a20  
col password for a20  
col spare4 for a65  
SELECT name,password,spare4 FROM user$ WHERE name LIKE 'USER\_%' ORDER BY 1;  
  
NAME PASSWORD SPARE4  
---------- ----------------- --------------------------------------  
USER\_10G 808E79166793CFD1  
USER\_11G S:22D8239017006EBDE054108BF367F225B5E7  
 31D12C91A3BEB31FA28D4A38  
USER\_12C T:C6CE7A88CC5D0E048F32A564D2B6A7BDC78A  
 2092184F28D13A90FC071F804E5EA09D4D2A37  
 49AA79BFD0A90D18DEC5788D2B8754AE20EE5C  
 309DBA87550E8AA15EAF2746ED431BF4543D2A  
 BE33E22678  
  
USER\_ALL BFD595809B6149CB S:804A87EA761505458FDED9B057A77FCF53DA  
 3DDBD6EDB168501EDF5C0B10;T:7950DF0D54D  
 EA24F1764EBC34A262D784E18F4292510B8A2E  
 0D0F7ADFEC1C6F1E22D841A9D91BAF0B9B0563  
 2F6D4898C6F4AE1EEF1509339EBCE261A1F36E  
 834A5E2DD9F1E772AB2D6413CCAB5EB0B23

Check what we do have in *sqlnet.ora*.

host grep -i ALLOWED /u00/app/oracle/network/admin/sqlnet.ora  
#SQLNET.ALLOWED\_LOGON\_VERSION\_CLIENT=12a  
SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER=11  
  
host sed -i "s|^SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER.\*|SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER=11|" \  
 /u00/app/oracle/network/admin/sqlnet.ora  
host sed -i "s|^SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER.\*|SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER=12|" \  
 /u00/app/oracle/network/admin/sqlnet.ora  
host sed -i "s|^SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER.\*|SQLNET.ALLOWED\_LOGON\_VERSION\_SERVER=12a|" \  
 /u00/app/oracle/network/admin/sqlnet.ora

Do some login tests

SQL> connect user\_10g/manager  
ERROR:  
ORA-01017: invalid username/password; logon denied  
  
  
Warning: You are no longer connected to ORACLE.  
  
connect user\_11g/manager

## Setup Kerberos

Check the configuration scripts in *sqlnet.ora*.

grep -i -A 11 -B 2 "Kerberos Configuration" $TNS\_ADMIN/sqlnet.ora  
  
################################################################  
# Kerberos Configuration  
################################################################  
SQLNET.AUTHENTICATION\_SERVICES = (BEQ,KERBEROS5)  
SQLNET.FALLBACK\_AUTHENTICATION = TRUE  
SQLNET.KERBEROS5\_KEYTAB = /u00/app/oracle/network/admin/urania.keytab  
SQLNET.KERBEROS5\_REALMS = /u00/app/oracle/network/admin/krb.realms  
SQLNET.KERBEROS5\_CC\_NAME = /u00/app/oracle/network/admin/krbcache  
SQLNET.KERBEROS5\_CONF = /u00/app/oracle/network/admin/krb5.conf  
SQLNET.KERBEROS5\_CONF\_MIT=TRUE  
SQLNET.AUTHENTICATION\_KERBEROS5\_SERVICE = oracle

Check the configuration scripts in *krb5.conf*.

cat $TNS\_ADMIN/krb5.conf  
  
####krb5.conf DB Server  
[logging]  
default = FILE:/u00/app/oracle/network/log/krb5lib.log  
kdc=FILE:/u00/app/oracle/network/log/krb5kdc.log  
admin\_server=FILE:/u00/app/oracle/network/log/kadmind.log  
  
[libdefaults]  
 default\_realm = POSTGASSE.ORG  
 clockskew=300  
 ticket\_lifetime = 24h  
 renew\_lifetime = 7d  
 forwardable = true  
  
[realms]  
 POSTGASSE.ORG = {  
 kdc = mneme.postgasse.org  
 admin\_server = mneme.postgasse.org  
}  
  
[domain\_realm]  
.postgasse.org = POSTGASSE.ORG  
postgasse.org = POSTGASSE.ORG

lookup hostname’s and check DNS configuration

cat /etc/resolv.conf  
# Generated by NetworkManager  
search aux.lan postgasse.org  
nameserver 192.168.56.70  
nameserver 10.154.0.1

nslookup mneme.postgasse.org  
Server: 192.168.56.70  
Address: 192.168.56.70#53  
  
Name: mneme.postgasse.org  
Address: 192.168.56.70  
Name: mneme.postgasse.org  
Address: 10.0.2.19

nslookup te2018\_eusdb.postgasse.org  
Server: 192.168.56.70  
Address: 192.168.56.70#53  
  
Name: urania.postgasse.org  
Address: 192.168.56.90

Create a service principle in MS AD

Create the keytab file

ktpass.exe -princ oracle/te2018\_eusdb.postgasse.org@POSTGASSE.ORG \  
 -mapuser te2018\_eusdb.postgasse.org -pass manager \  
 -crypto ALL -ptype KRB5\_NT\_PRINCIPAL \  
 -out C:\u00\app\oracle\network\te2018\_eusdb.keytab

Connect as kerberos User

Vulnerability

|  |  |
| --- | --- |
| Vulnerability | Risk [^1] |
| V01 | high [^99] |
| V02 | gelb |

## Setup OUD AD Proxy

### Requirements

Before you can start you may need a few things.

* Docker environment (eg. Docker community edition)
* OUD Docker Images in particular one for OUD 12.2.1.3 with the latest OUD base see [oehrlis/docker](https://github.com/oehrlis/docker) soon you may also get the Dockerfiles from the Oracle Repository see [pull request 911](https://github.com/oracle/docker-images/pull/911)
* An MS AD Directory server or at lease a few credential to access one

### Environment Variable

To type less you just have to define a few environment variables. Basically you will define the local Docker volume path, container name, container hostname and the OUD instance name.

export MY\_CONTAINER="te2018\_oud"  
export MY\_VOLUME\_PATH="/data/docker/volumes/$MY\_CONTAINER"  
export MY\_HOST="$MY\_CONTAINER.postgasse.org"  
export MY\_OUD\_INSTANCE="oud\_adproxy"

### Create the container

Just create a container without starting it. Adjust ports, base DN etc.

docker container create --name $MY\_CONTAINER \  
 --volume $MY\_VOLUME\_PATH:/u01 \  
 -p 1389:1389 -p 1636:1636 -p 4444:4444 \  
 -e OUD\_CUSTOM=TRUE \  
 -e BASEDN="dc=postgasse,dc=org" \  
 -e OUD\_INSTANCE=$MY\_OUD\_INSTANCE \  
 --hostname $MY\_HOST \  
 --dns 192.168.56.70 \  
 --dns-search postgasse.org \  
 oracle/oud:12.2.1.3.180626

Get and configure your create scripts out of the container from the OUD base. Alternatively you may also get it directly from GitHub [oehrlis/oudbase](https://github.com/oehrlis/oudbase).

Get the OUD EUS AD templates from the Docker container created before.

mkdir -p $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE  
docker cp \  
 $(docker ps -aqf "name=$MY\_CONTAINER"):/u00/app/oracle/local/oudbase/templates/create/oud12c\_eus\_ad\_proxy \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE  
mv $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/oud12c\_eus\_ad\_proxy $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create  
mkdir -p $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/etc  
echo "manager" >$MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/etc/${MY\_OUD\_INSTANCE}\_pwd.txt

Update the *00\_init\_environment* according to your environment. In particular the variables AD\_PDC\_HOST,AD\_PDC\_PORT, AD\_PDC\_USER, AD\_PDC\_PASSWORD and BASEDN, GROUP\_DN, USER\_DN

vi $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
  
sed -i -e "s|<PDC\_HOSTNAME>|mneme.postgasse.org|g" \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
sed -i -e 's|<USER\_DN>|CN=OUD\\ Admin,CN=Users,dc=postgasse,dc=org|g' \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
sed -i -e "s|<PASSWORD>|manager|g" \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
  
sed -i -e 's|^export BASEDN.\*|export BASEDN="dc=postgasse,dc=org"|g' \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
sed -i -e 's|^export GROUP\_OU.\*|export GROUP\_OU="ou=Groups,dc=postgasse,dc=org"|g' \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
sed -i -e 's|^export USER\_OU.\*|export USER\_OU="ou=People,dc=postgasse,dc=org"|g' \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
sed -i -e "s|dc=example,dc=com|dc=postgasse,dc=org|g" \  
 $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment  
  
cat $MY\_VOLUME\_PATH/admin/$MY\_OUD\_INSTANCE/create/00\_init\_environment

Lets go. Start the container and let the scripts create the OUD instance.

docker start $MY\_CONTAINER

Enjoy the log and see how your OUD EUS AD proxy is created

docker logs -f $MY\_CONTAINER

## Setup EUS

dbca -configureDatabase -sourceDB $ORACLE\_SID \  
 -registerWithDirService true \  
 -dirServiceUserName "cn=eusadmin" -dirServicePassword manager \  
 -walletPassword TVD04manager -silent

Create a global DB User

DROP USER eus\_users;  
CREATE USER eus\_users IDENTIFIED GLOBALLY;   
GRANT tvd\_connect TO eus\_users;

Define a EUS mapping to the shared schema created before

eusm createMapping database\_name="$ORACLE\_SID" \  
 realm\_dn="dc=postgasse,dc=org" map\_type=SUBTREE \  
 map\_dn="ou=People,dc=postgasse,dc=org" schema=EUS\_USERS \  
 ldap\_host="te2018\_oud.postgasse.org" ldap\_port=1389 \  
 ldap\_user\_dn="cn=eusadmin" \  
 ldap\_user\_password="manager"

eusm listMappings database\_name="$ORACLE\_SID" \  
 realm\_dn="dc=postgasse,dc=org" \  
 ldap\_host="te2018\_oud.postgasse.org" ldap\_port=1389 \  
 ldap\_user\_dn="cn=eusadmin" \  
 ldap\_user\_password="manager"

Passwords are in docker logs or in the password files in MY\_OUD\_INSTANCE/etc

check EUS connection

SQL> conn dinu/manager  
Connected.  
SQL> @sousrinf  
Database Information  
--------------------  
- DB\_NAME : TDB122A  
- DB\_DOMAIN :  
- INSTANCE : 1  
- INSTANCE\_NAME : TDB122A  
- SERVER\_HOST : urania  
-  
Authentification Information  
----------------------------  
- SESSION\_USER : EUS\_USERS  
- PROXY\_USER :  
- AUTHENTICATION\_METHOD : PASSWORD  
- IDENTIFICATION\_TYPE : GLOBAL SHARED  
- NETWORK\_PROTOCOL :  
- OS\_USER : oracle  
- AUTHENTICATED\_IDENTITY: DINU  
- ENTERPRISE\_IDENTITY : cn=Martin Berger,ou=People,dc=postgasse,dc=org  
-  
Other Information  
-----------------  
- ISDBA : FALSE  
- CLIENT\_INFO :  
- PROGRAM : sqlplus@urania (TNS V1-V3)  
- MODULE : SQL\*Plus  
- IP\_ADDRESS :  
- SID : 33  
- SERIAL# : 17568  
- SERVER : DEDICATED  
- TERMINAL : pts/1  
  
PL/SQL procedure successfully completed.