Last update: 25-Nov-2014

Live adventure - from my PC to Oracle Remote Database

Kirill Loifman
adidas Group AG
Herzogenaurach

Keywords:

Oracle, database, security, remote connection, firewall, tunnelling, SSH, wallet, proxy

Introduction

Best practices and live demo on remote database connection technics including remote database connection setup, troubleshooting and undocumented tips. Together with audience we break a firewall, establish SSH RSA server authentication and receive DB access without an obvious user. Further connection techniques like Secure Password Store and Proxy Authentication will be reviewed.

Our journey scenario and test goals

In our test scenario we will achieve following goals:

- Establish remote connection from a Windows Client PC to a Remote Oracle database via Firewall
- Connect to any database user without password
- Simplify further database connection attempts
- Explore a few security points along the way

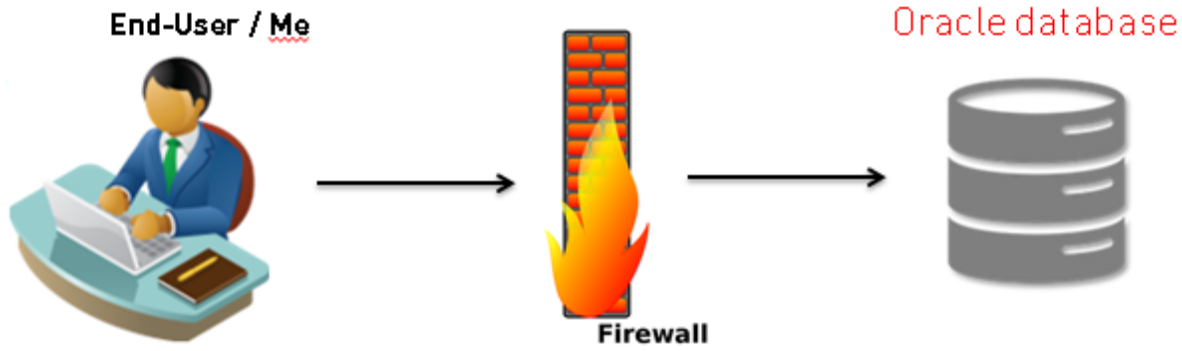


Illustration 1: Test scenario

Host name: dadbm-host
Host name virtual: dadbm-vip
DB name: orcl
Port: 1521

Test preparation

The following picture depicts the main components that are required for remote database connection

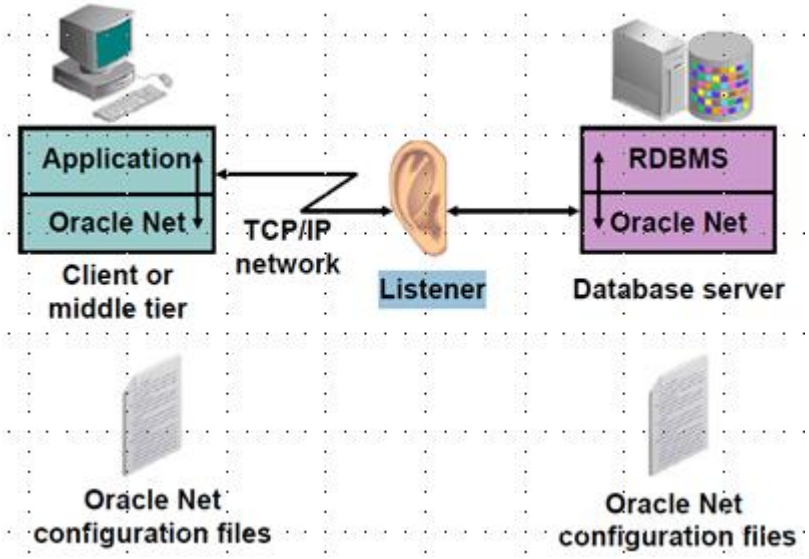


Illustration 2: Oracle NET components

- Oracle database server
- Network interface between Client & Database
- Firewall ports to be opened => can bypass this
- Oracle Listener
- Oracle Client installable => optional
- Oracle Client utility
- Oracle database user => can trick with this
- Oracle user password => can simplify this
- tnsnames.ora => optional

Following steps are required to prepare the test case

- Create and startup Oracle database on Linux box
- Configure and startup Oracle listener using tools (netmgr, lsnrctl)
- Configure Oracle Client using tools (OUI, ping, tnsping, telnet, Putty, SQL*Plus)
- Create connection string in %TNS_ADMIN%\tnsnames.ora

```
orcl=(description=(address=(protocol=tcp)(host=dadbm-
vip)(port=1521))(connect_data=(service_name=orcl))))
```

- Identify Oracle environment variables on Windows
  - OUI => ORACLE_HOME
  - regedit => ORACLE_HOME, NLS_LANG, …what is set by OUI (might be not available)
  - SQL*Plus => what is set by OUI

- set => Windows environment variables (PATH, etc.)

Potential connection ways to our database

```
sqlplus user/pass@orcl                          # tnsnames.ora

sqlplus ususer/pass@'(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dadbm-vip)
(1521=1521))(CONNECT_DATA=(SERVICE_NAME=orcl)))'

sqlplus user@'(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dadbm-vip)
(HOST=1521))(CONNECT_DATA=(SID=orcl)))'

sqlplus user/pass@//dadbm-vip:1521/orcl       # EZCONNECT (sqlnet.ora)
sqlplus user@'//dadbm-vip:1521/orcl'          # without password
sqlplus user@'//dadbm-vip/orcl'               # default port

jdbc:oracle:thin:@dadbm-vip:1521/orcl         # custom JDBC url
jdbc:oracle:thin:user@dadbm-vip:1521:orcl
jdbc:oracle:oci:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=dadbm-
vip)(1521=1521))(CONNECT_DATA=(SERVICE_NAME=orcl)))
```

See Demo 1…

Resolving Firewall issue using tunnelling

Following test identifies a potential issue with Firewall

- `ping <dadbm-host>` => ping works!
- `tnsping orcl` => TNS-12535: TNS:operation timed out
- `sqlplus any_user/any_pass@orcl` => TNS-03505: Failed to resolve name
- `telnet <dadbm-host> 1521` => Could not open connection to the host, on port 1521: Connect failed

The proper solution to this problem would be asking Security team to open a database listener port 1521 in Firewall to be able to communicate between Oracle client and target host via the listener port.

Source IP	Source Zone	Destination IP	Destination Zone	Destination Port
<Client PC IP>	LAN/WAN	<Oracle Server IP>	DMZ	1521

In our Demo alternatively we will use SSH tunneling to break a firewall. See the solution steps below:

- SSH port 22 is usually opened in Firewalls by default
- Request a Linux user on database host
- Use Putty Firewall tunneling feature to add a tunnel:
  - Putty Menu: *Connection -> SSH -> Tunnels*
  - Tunnel listener port 1521 to port 8822 (see Illustration 3)
- Adjust DB connection string in tnsnames.ora as following:

```
orcl=(description=(address=(protocol=tcp)(127.0.0.1)(port=8822))(con
nect_data=(service_name=orcl))))
```

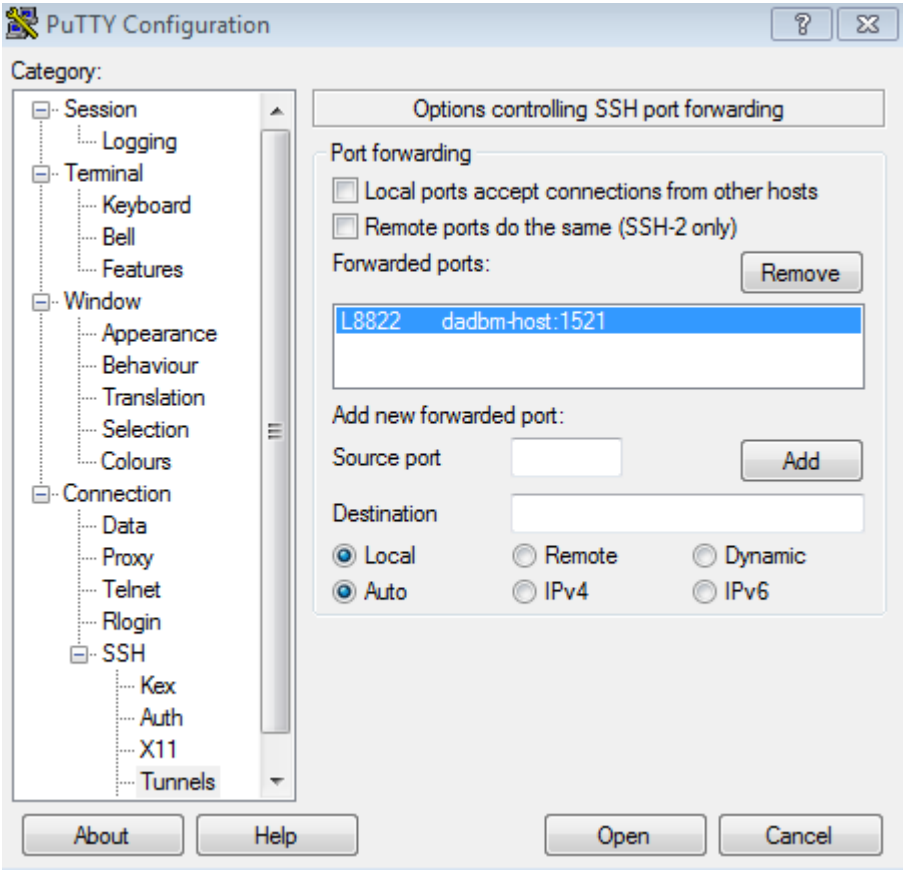


Illustration 3: Putty tool – SSH tunneling feature

As a result of SHH tunnelling local client tools can access the remote database by connecting to <mycomputer>:8822.

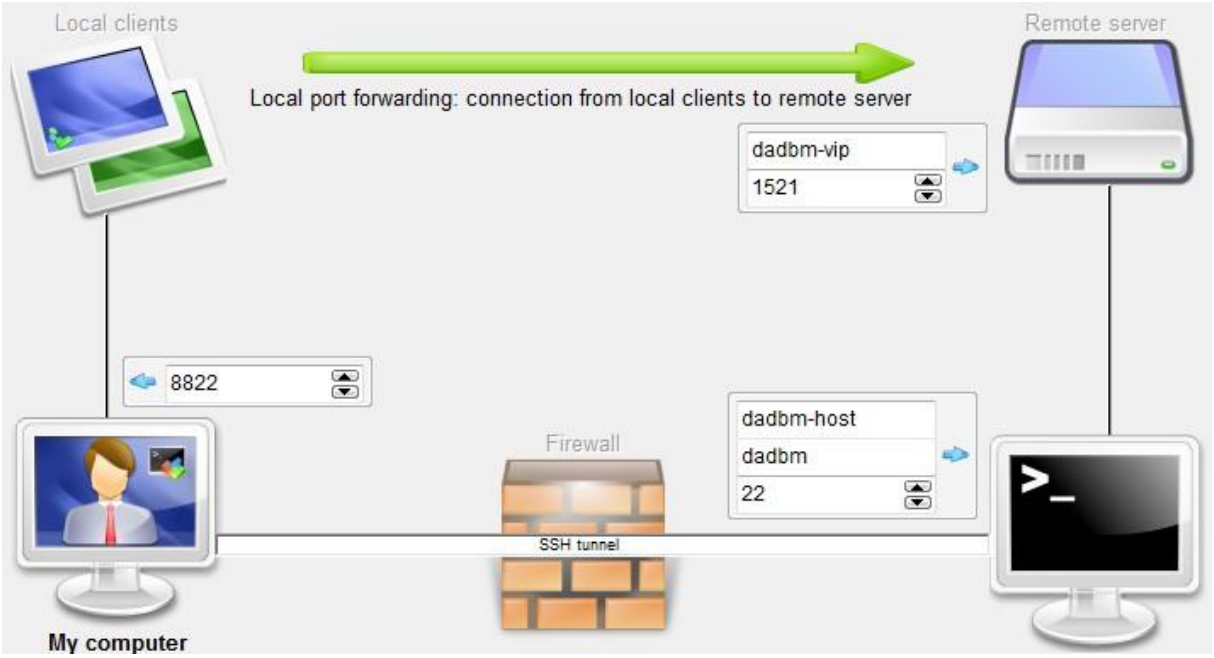


Illustration 4: How SSH tunneling works

Database connection - Ping host and database

Below are different methods of testing oracle database connection after firewall tunneling is completed

```
ping <dadbm-host-ip>

ping <dadbm-host>

telnet 127.0.0.1 8822

tnsping
(description=(address=(protocol=tcp)(host=127.0.0.1)(port=8822))(con
nect_data=(service_name=orcl)))

tnsping 127.0.0.1:8822/orcl

tnsping orcl
```

See Demo 2 …

Connect as SYSDBA without having a user in a database

```
DBAUSER@SQL+> select * from dba_users where username = 'DADBM';
no rows selected

sqlplus dadbm@orcl as sysdba
```

```
Password
SQL>
```

Create a database user with a grant trick

```
SQL+> grant OEM_MONITOR to dadbm;
```
=> ORA-01917: user or role 'DADBM' does not exist

```
SQL+> grant OEM_MONITOR to dadbm identified by ora4u;
```
=> Grant succeeded.

Establish a database connection with different ways

```
sqlplus dadbm@orcl

sqlplus
dadbm@'(description=(address=(protocol=tcp)(host=127.0.0.1)(port=882
2))(connect_data=(service_name=orcl)))'

sqlplus dadbm@'//127.0.0.1:8822/orcl'

jdbc:oracle:thin:@127.0.0.1:8822/orcl
jdbc:oracle:oci:@127.0.0.1:8822/orcl
```

See Demo 3…

Database connection troubleshooting

Following connection errors may arise:

```
ORA-12154: TNS:could not resolve the connect identifier specified
ORA-12198: TNS:could not find path to destination
ORA-12203: TNS:unable to connect to destination
ORA-12533: TNS:illegal ADDRESS parameters
TNS-12541: TNS:no listener
…
```

Use a following post on dadbm.com to troubleshoot database connection:

http://www.dadbm.com/how-to-troubleshoot-oracle-remote-database-connection/

Avoid database password using Secure Password Store

1) Configure Oracle Client environment for a Wallet

tnsnames.ora file:

```
orcl =
```

```
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 127.0.0.1)(PORT = 8822))
    (CONNECT_DATA = (SERVICE_NAME = orcl))
  )
```

sqlnet.ora file

```
WALLET_LOCATION =
   (SOURCE =
      (METHOD = FILE)
      (METHOD_DATA = (DIRECTORY =
C:\oracle\product\11.2.0\client_1\network\admin))
)
SQLNET.WALLET_OVERRIDE = TRUE
```

2) Create a wallet on the client

```
mkstore -wrl <wallet_location> -create
mkstore -wrl C:\oracle\product\11.2.0\client_1\network\admin -create
Enter password: <= Choose wallet password and remember it
Enter password again:
```

Directory content of C:\oracle\product\11.2.0\client_1\network\admin:

```
12-Jun-14  12:39         3,589 cwallet.sso       # Auto login wallet file
12-Jun-14  12:39         3,512 ewallet.p12       # PKCS#12 wallet file
12-Jun-14  12:26           644 sqlnet.ora
04-Jun-14  22:49        15,237 tnsnames.ora
```

3) Create database connection credentials in the wallet

```
mkstore -wrl %TNS_ADMIN% -createCredential <alias> <user> <passwd>
mkstore -wrl %TNS_ADMIN% -createCredential orcl    dadbm  ora4u
Enter password: <= Wallet password
```

4) Managing External Password Store Credentials

```
mkstore -wrl %TNS_ADMIN% -listCredential
mkstore -wrl %TNS_ADMIN% -createCredential orcl dadbm9 pass
mkstore -wrl %TNS_ADMIN% -modifyCredential orcl dadbm9 newpassword
mkstore -wrl %TNS_ADMIN% -deleteCredential orcl
…
```

5) Connect to a database without a password

```
sqlplus /@orcl
```

5) More Notes on Secure External Password Store

- DB does not know about your secure password store and vice versa
- <DB alias> is unique in Wallet -> for different user create another alias

- Wallet can be generated somewhere else and copied to you client
- Extended Wallet management with orapki utility
- More details in MOS DOC ID 340559.1

See Demo 4 …

Connect to another user without a password – Proxy Authentication

We will enable proxy Authentication to connect to another user without a password using following steps

1) Create more database users for a test:

```
grant connect,resource to dadbm1 identified by adJIk3909sdfj;
grant connect,resource to dadbm2 identified by g7fk3kZfdhl05;
alter user dadbm1 grant connect through dadbm;
```

2) Connect to a database with proxy user and secure password store

```
-- Usual way
sqlplus proxy_user[proxy_client]@orcl

-- Proxy connect with secure password store (no password for proxy user required)
sqlplus [dadbm1]@orcl

-- Similar
sqlplus /@orcl
SQL>conn [dadbm1]@orcl
```

3) Check who you are on the database

```
alter session set current_schema=dadbm2;

select sys_context('USERENV','PROXY_USER') PROXY_USER,
sys_context('USERENV','SESSION_USER') SESSION_USER,
sys_context('USERENV','CURRENT_SCHEMA') CURRENT_SCHEMA
from dual;

PROXY_USER           SESSION_USER         CURRENT_SCHEMA
-------------------- -------------------- --------------------
DADBM                DADBM1               DADBM2
```

4) More information

```
select * from dba_proxies;
select * from proxy_roles;
```

Simplify connection to a database

1) Create 2 files sql.bat and proxy.sql on the client with following contect

```
# sql.bat
 SET LOCAL=%1
 sqlplus /@%LOCAL% %2

# proxy.sql
set verify off
alter user &&1 grant connect through dadbm;
connect [&&1]@&_CONNECT_IDENTIFIER
show user
```

2) Ask for extra grant to my user dadbm :

```
grant alter user to dadbm;
```

3) Connect with user dadbm to any TNS alias using sql.bat and to any DB user with proxy.sql
using combination of Proxy Authentication and Secure Password Store (no dadbm password required)

```
sql orcl
DADBM@orcl+>@proxy dadbm1
USER is "DADBM1"
DADBM1@orcl+>
```

Simplify connection to Linux box(es)

To simplify a remote terminal connection to a Linux box (where our database is running) we will configure Public-key SSH RSA authentication using Putty tool set:

- Create an empty file on Linux box: $HOME/.ssh/authorized_keys
- Use puttygen.exe to generate SSH2-RSA keys (public and private) using *Passphrase*
- Copy and paste generated public key into authorized_keys file on Linux
- Save private key into dadbm.ppk file on locally on the client
- Adjust the Putty terminal session by specifying Private key file (dadbm.ppk) in SSH-> Auth Menu
- Adjust putty_auth_agent.bat file with proper path to Putty agent utility
- Execute putty_auth_agent.bat file and enter *Passphrase* just once
- Use Putty.exe to establish Linux connection without using a password (see Illustration 5)

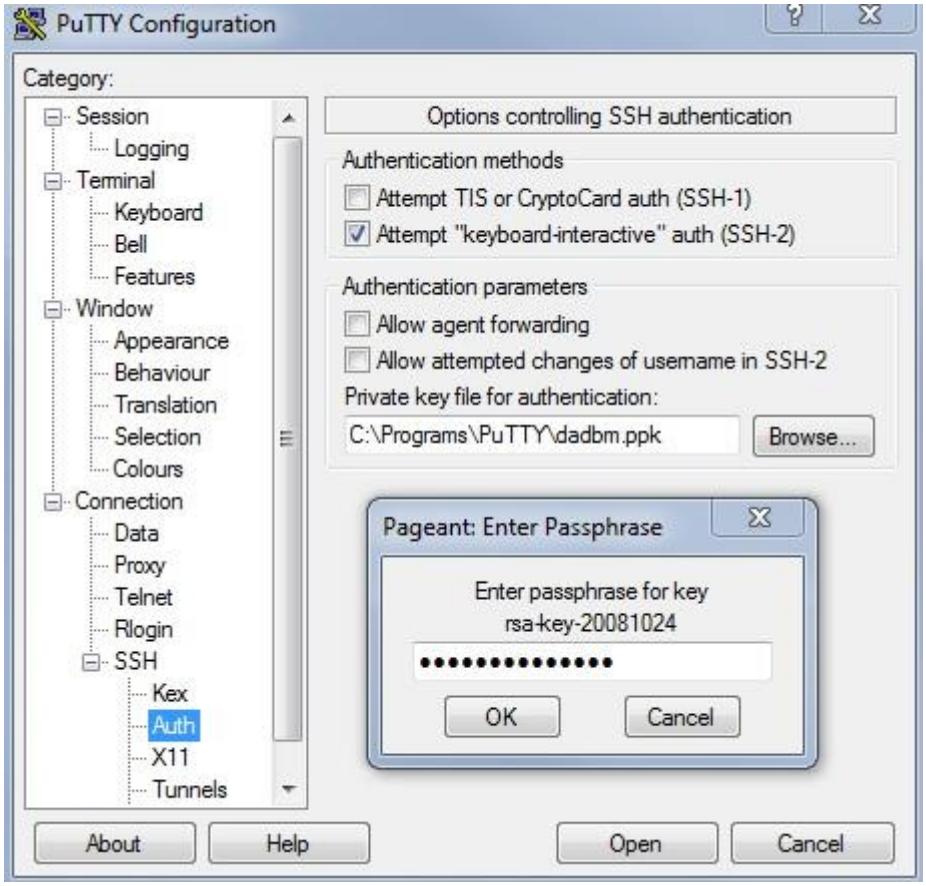
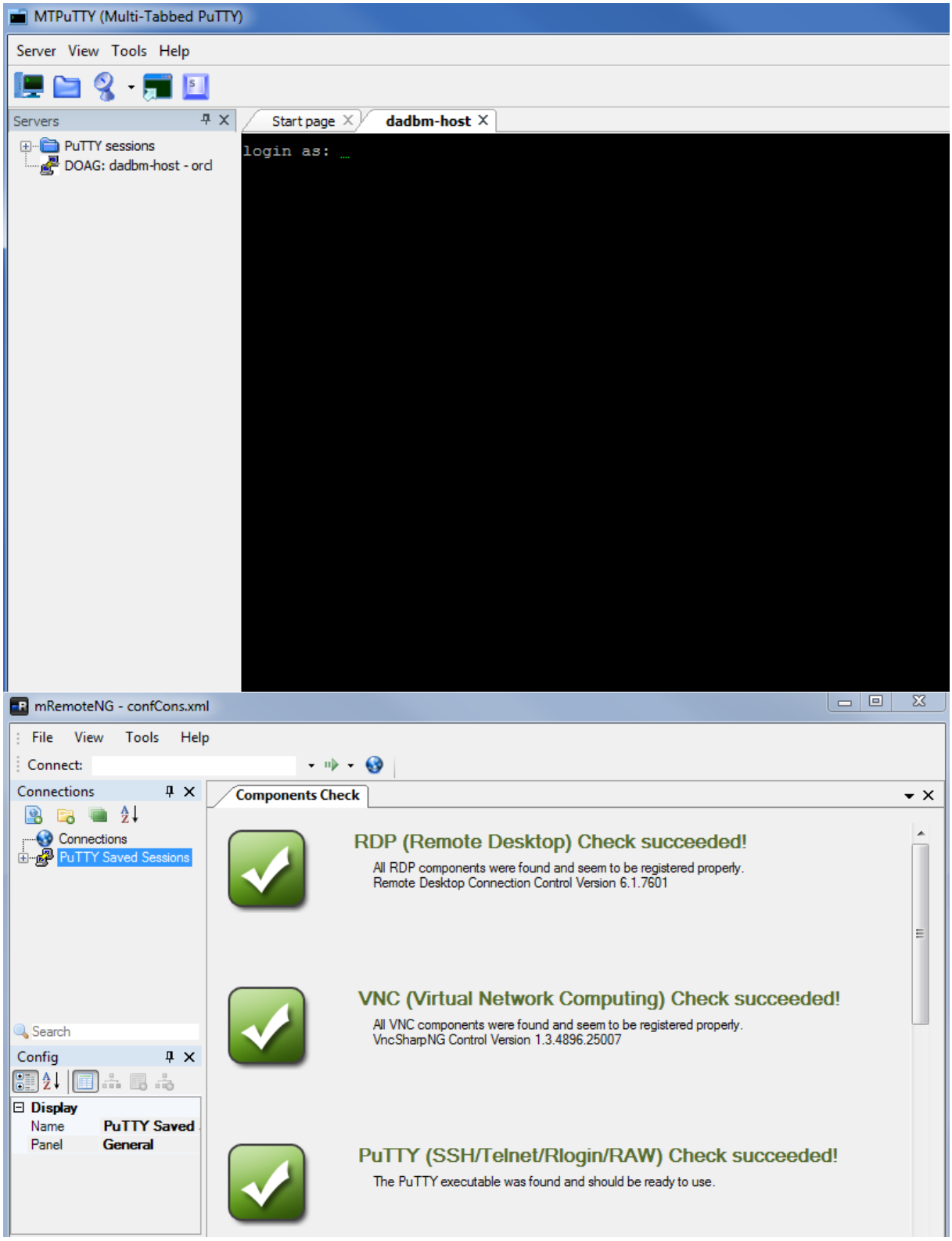


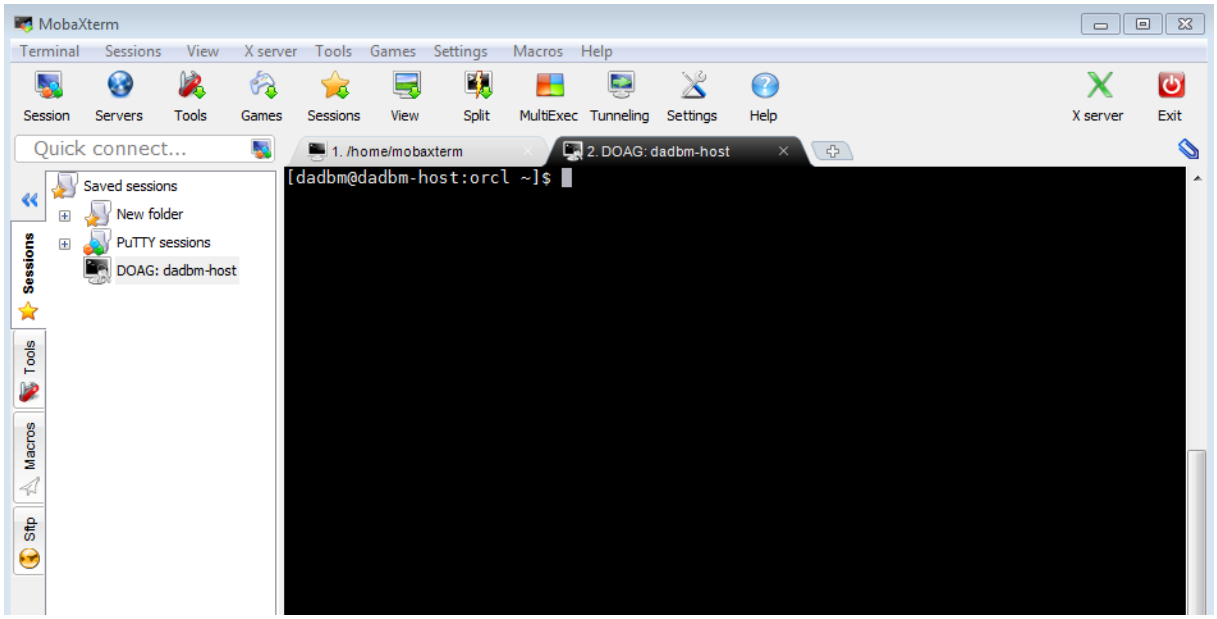
Illustration 5: Public-key SSH2-RSA authentication using Putty utility

See Demo 6…

Live Adventure exercise Summary

- Setting up the database and server connections this way can simplify the day-to-day tasks of DBAs and developers.
- But think about security! That was just a playground!
- Firewall tunneling and Secure Password Store are supported by other tools
- Get your servers listed using other fancy GUIs: MTPutty, mRemoteNG, MobaXterm, or others.


MTPuTTY (Multi-Tabbed PuTTY)
Server View Tools Help
Servers
PuTTY sessions
DOAG: dadbm-host - orcl
Start page
dadbm-host
login as:
mRemoteNG - confCons.xml
File View Tools Help
Connect:
Connections
Connections
PuTTY Saved Sessions
Search
Config
Display
Name PuTTY Saved
Panel General
Components Check
RDP (Remote Desktop) Check succeeded!
All RDP components were found and seem to be registered properly.
Remote Desktop Connection Control Version 6.1.7601
VNC (Virtual Network Computing) Check succeeded!
All VNC components were found and seem to be registered properly.
VncSharpNG Control Version 1.3.4896.25007
PuTTY (SSH/Telnet/Rlogin/RAW) Check succeeded!
The PuTTY executable was found and should be ready to use.



Contact address:

Kirill Loifman
Berliner Str. 15
91074, Herzogenaurach

Phone: +49(0)160 884 2541
Email dadbm1@gmail.com
Internet: www.dadbm.com
www.twitter.com/loifmkir
www.google.com/+dadbm
www.facebook.com/pages/DaDBm/425726914237370
www.youtube.com/dadbman