HOW TO ACCESS YOUR CLOUD POSTGRESQL SERVERS FROM HIGHLY SECURED CORPORATE ENVIRONMENT

TATIANA KRUPENYA AND SERGE RIDER

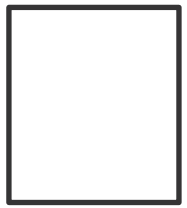
dbeaver.com

INTRO

- Application
- psql
- DBeaver

PostgreSQL server

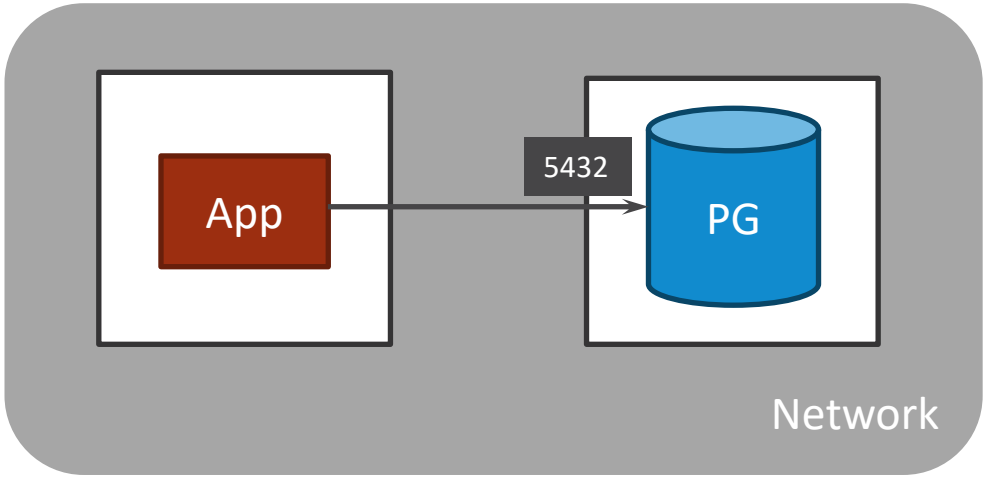
Network



Host

Port number

DIRECT ACCESS



1. Configure remote access from your client IP address.

pg_hba.conf

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     trust
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
hostssl all             all             127.0.0.1/32            md5
# IPv6 local connections:
host    all             all             ::1/128                 md5
host    all             all             143.202.92.44/32        md5
```

2. Configure PG server to listen real IP address

postgresql.conf

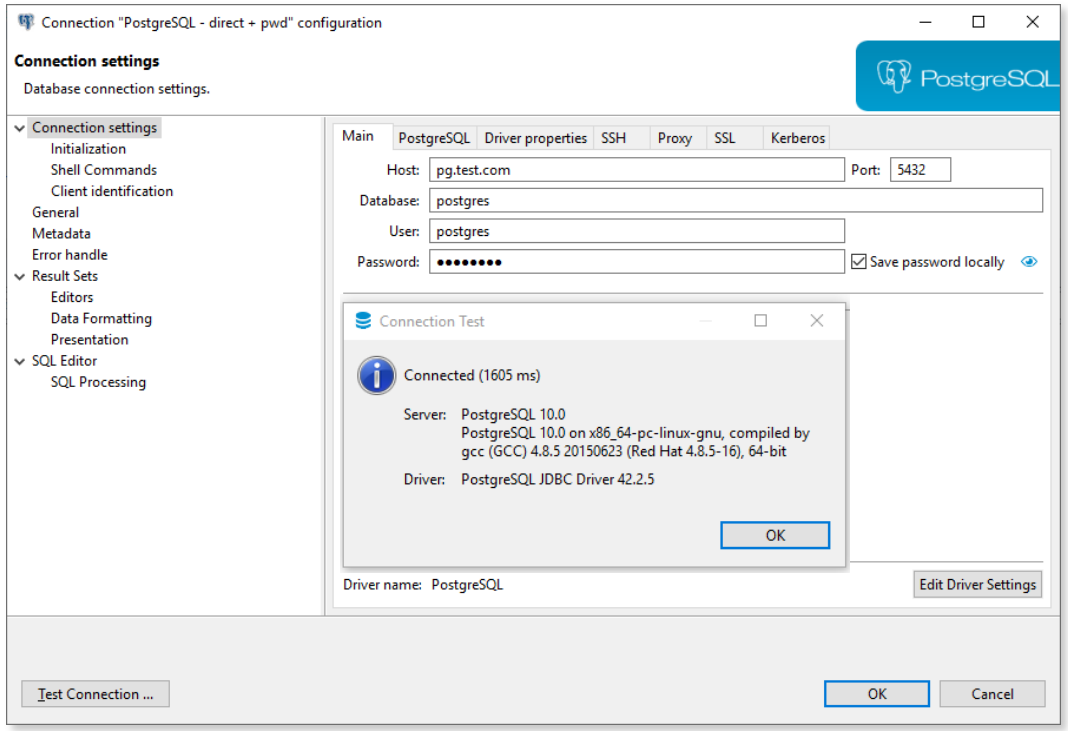
```
# - Connection Settings -

#listen_addresses = 'localhost'         # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
listen_addresses = 'localhost,pg-remote.test.net'
```

DIRECT ACCESS

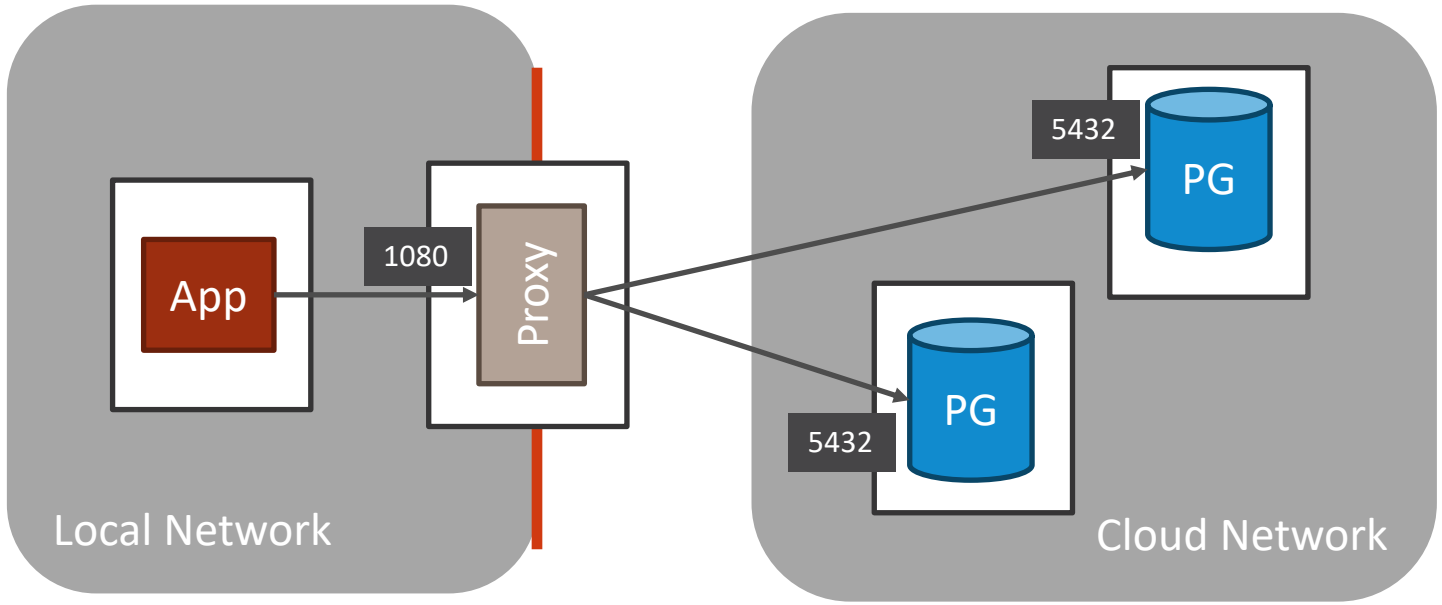
PSQL VIEW

```
user@host:~$ psql -h pg.test.com -p 5432 -U postgres
Password for user postgres:
psql (12.1 (Ubuntu 12.1-1.pgdg18.04+1), server 10.0)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384,
bits: 256, compression: off)
Type "help" for help.

postgres=#
```

DBEAVER VIEW



PROXY (SOCKS5)

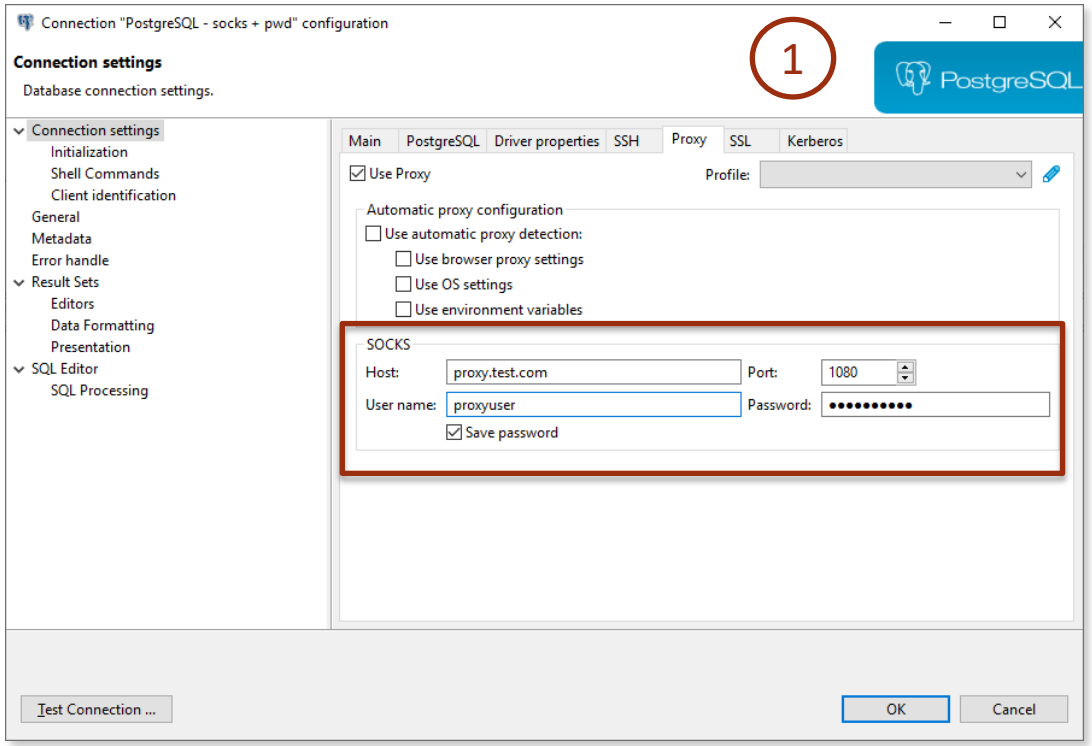


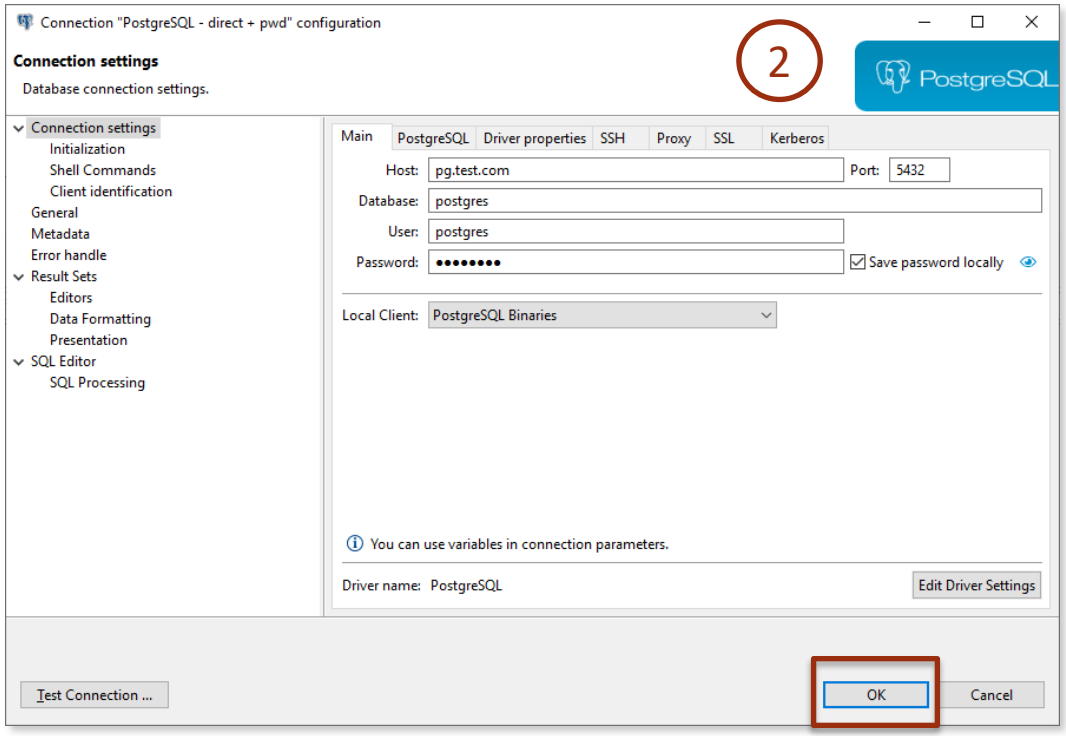
tsocks.conf

```
local = 192.168.0.0/255.255.255.0
local = 10.0.0.0/255.0.0.0
local = proxy.test.com/255.0.0.0

path {
      reaches = pg.test.com/255.0.0.0
      server = proxy.test.com
      server_type = 5
      default_user = user
      default_pass = password
}

server = proxy.test.com
server_type = 5
server_port = 1080
```

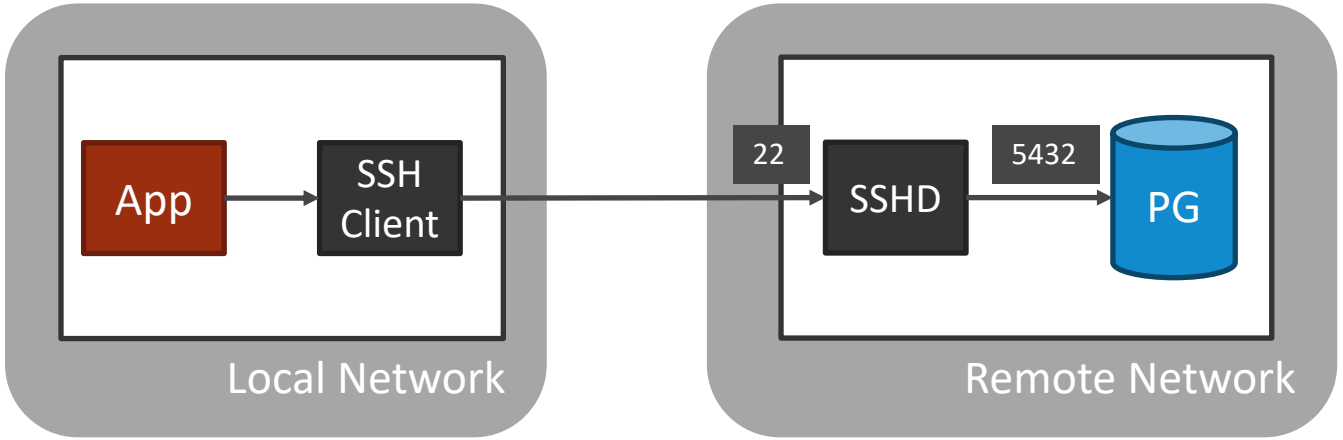
```
$ tsocks psql -U postgres -h pg.test.com postgres
```

PROXY (SOCKS5)



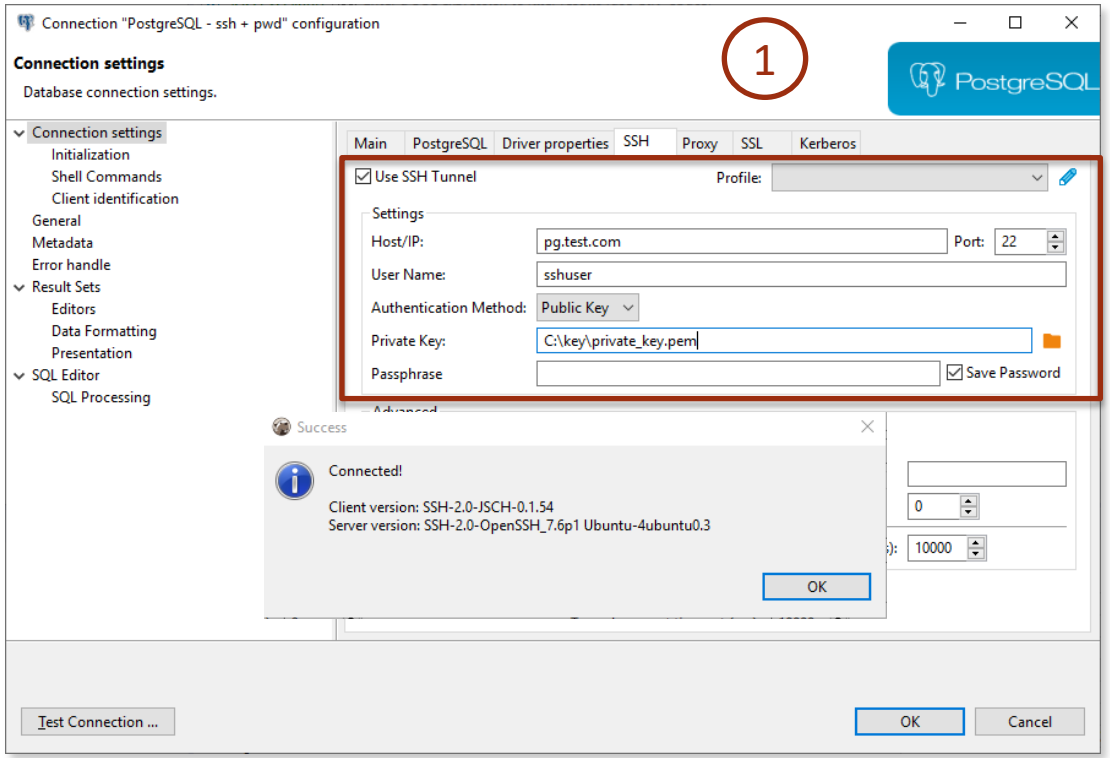


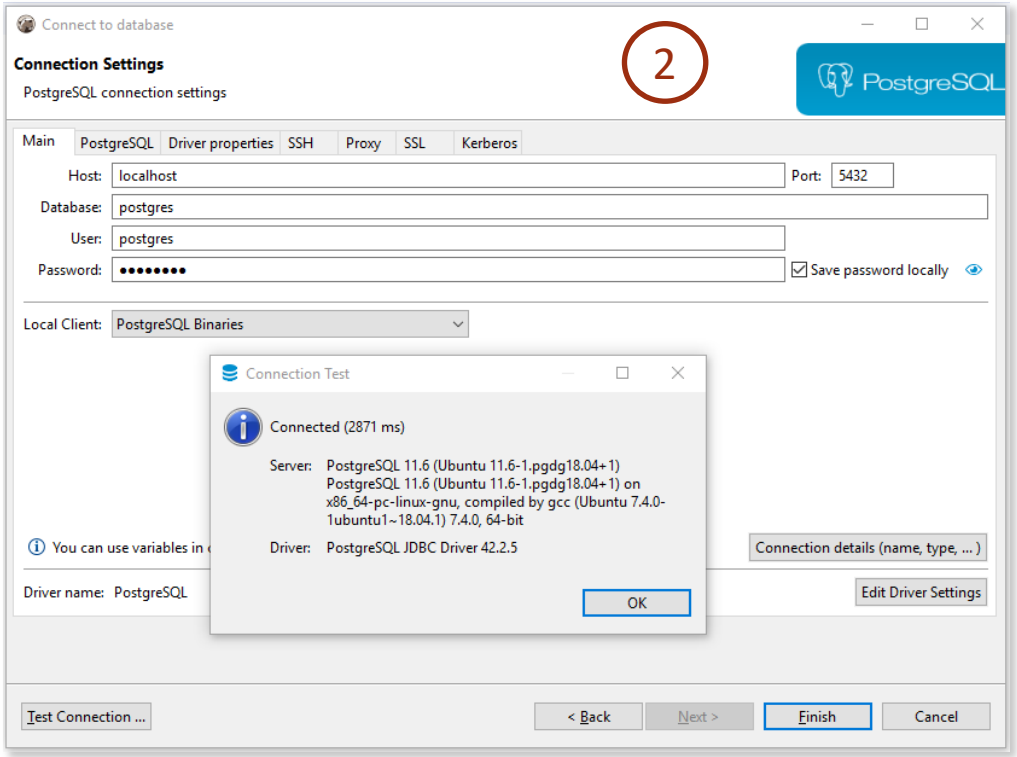
SSH TUNNEL



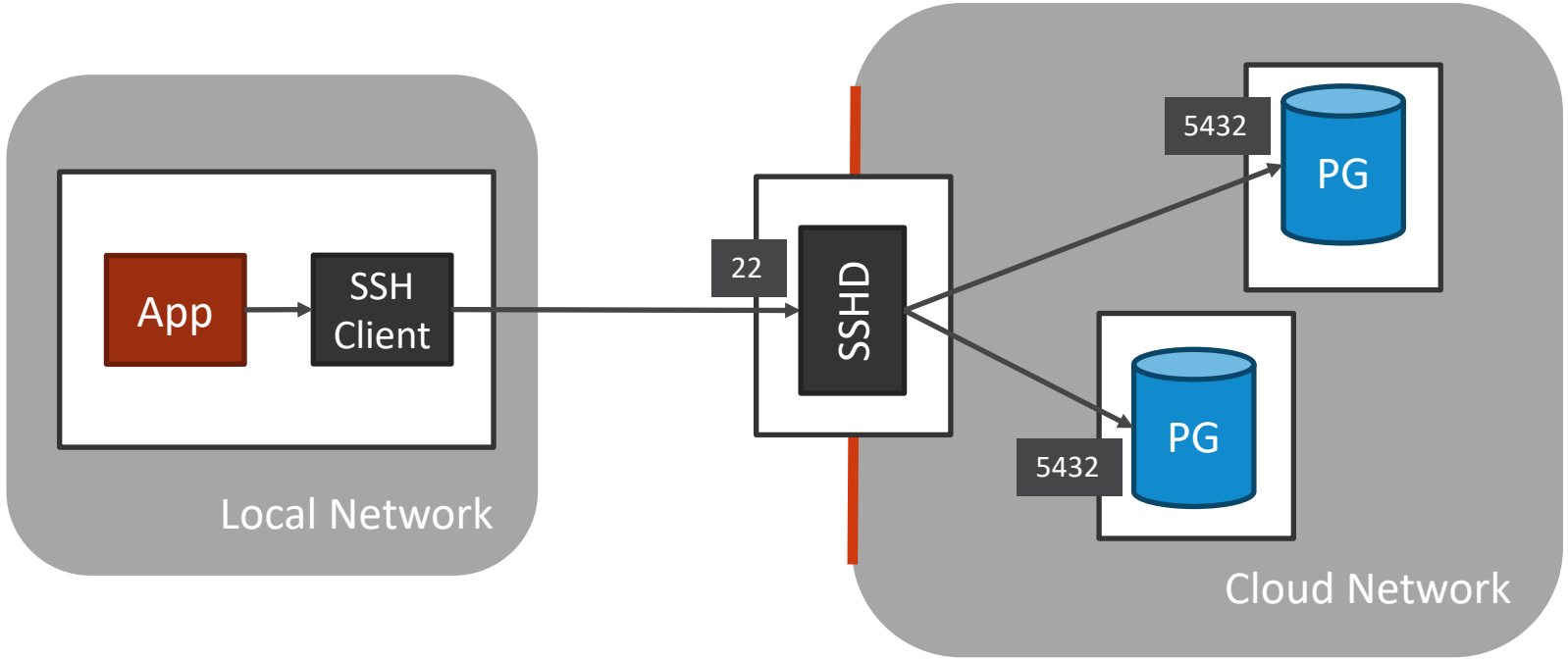

```
user@host:~$ ssh -L 54321:127.0.0.1:5432 user@pg.test.com -N -f
user@db1.dev.test.com's password:
user@host:~$ psql -h localhost -p 54321 -U postgres postgres
Password for user postgres:
psql (12.1 (Ubuntu 12.1-1.pgdg18.04+1), server 10.0)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.
postgres=# select version();
version
------------------------------------------------------------------------------------------------------------
PostgreSQL 10.0 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-16), 64-bit
(1 row)
postgres=#
```

SSH TUNNEL


Connection "PostgreSQL - ssh + pwd" configuration
1
Connection settings
Database connection settings.
PostgreSQL
Connection settings
Initialization
Shell Commands
Client identification
General
Metadata
Error handle
Result Sets
Editors
Data Formatting
Presentation
SQL Editor
SQL Processing
Main
PostgreSQL
Driver properties
SSH
Proxy
SSL
Kerberos
Use SSH Tunnel
Profile:
Settings
Host/IP:
pg.test.com
Port:
22
User Name:
sshuser
Authentication Method:
Public Key
Private Key:
C:\key\private_key.pem
Passphrase
Save Password
Success
Connected!
Client version: SSH-2.0-JSCH-0.1.54
Server version: SSH-2.0-OpenSSH_7.6p1 Ubuntu-4ubuntu0.3
OK
Test Connection ...
OK
Cancel

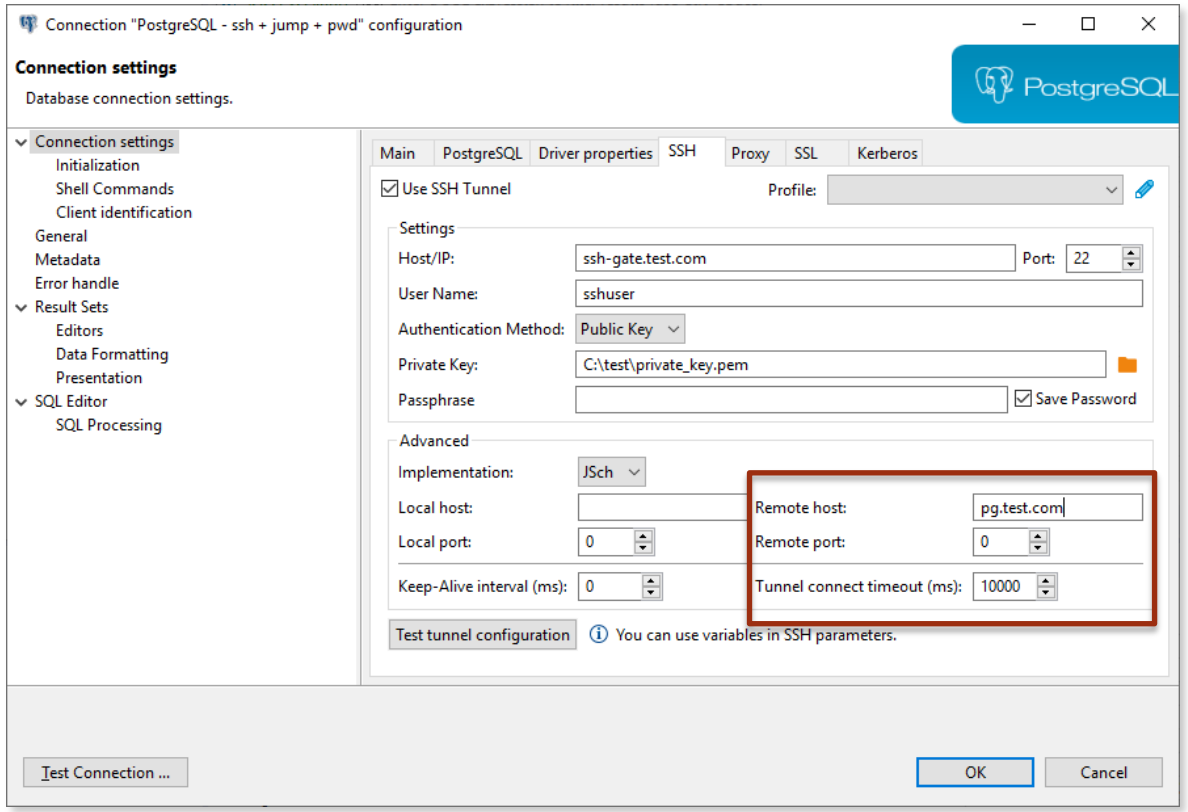

Connect to database
2
Connection Settings
PostgreSQL connection settings
PostgreSQL
Main
PostgreSQL
Driver properties
SSH
Proxy
SSL
Kerberos
Host:
localhost
Port:
5432
Database:
postgres
User:
postgres
Password:
Save password locally
Local Client:
PostgreSQL Binaries
Connection Test
Connected (2871 ms)
Server:
PostgreSQL 11.6 (Ubuntu 11.6-1.pgdg18.04+1)
PostgreSQL 11.6 (Ubuntu 11.6-1.pgdg18.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0, 64-bit
Driver:
PostgreSQL JDBC Driver 42.2.5
OK
You can use variables in
Connection details (name, type, ...)
Driver name: PostgreSQL
Edit Driver Settings
Test Connection ...
< Back
Next >
Finish
Cancel

SSH TUNNEL WITH JUMP SERVER



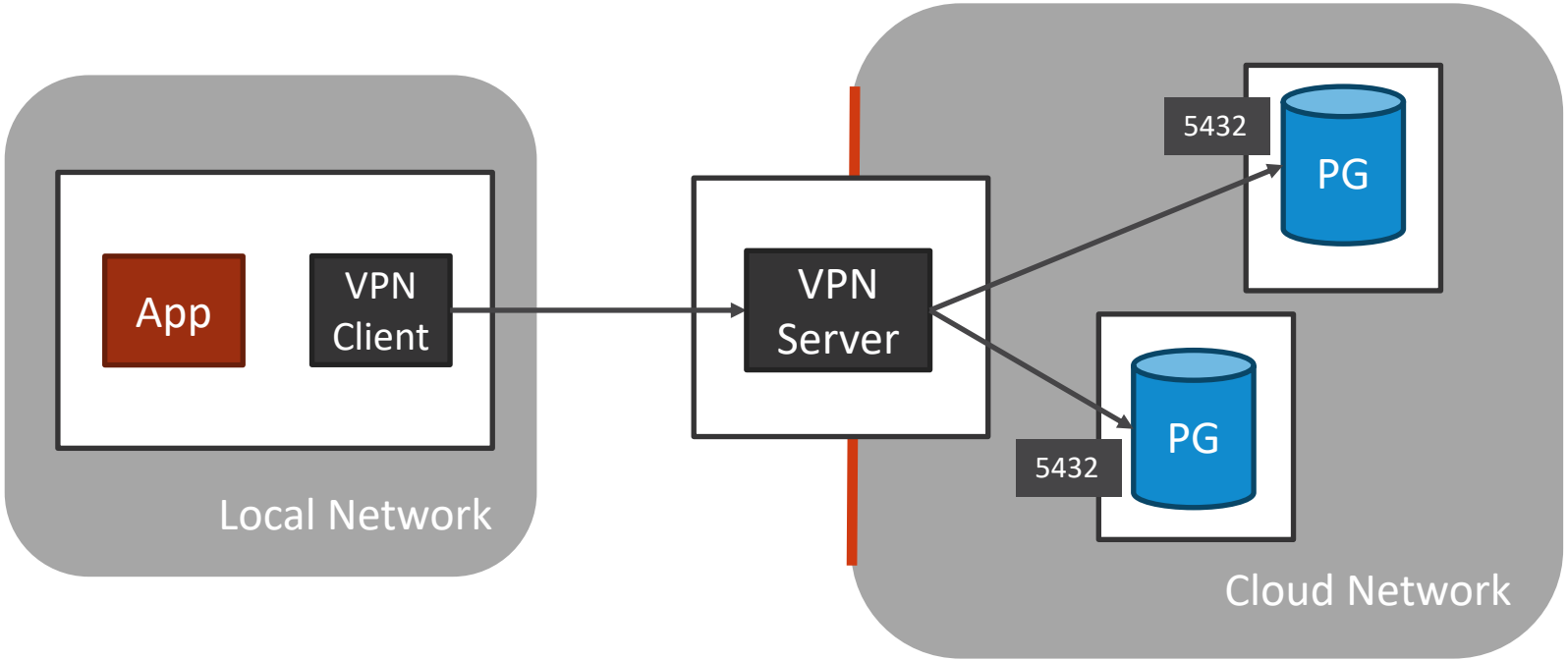
SSH TUNNEL WITH JUMP SERVER

```
user@host:~$ ssh -L 54321:ssh-gate.test.com:5432 user@pg.test.com -N -f
user@db1.dev.test.com's password:
user@host:~$ psql -h localhost -p 54321 -U postgres postgres
Password for user postgres:
psql (12.1 (Ubuntu 12.1-1.pgdg18.04+1), server 10.0)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384,
bits: 256, compression: off)
Type "help" for help.
postgres=# select version();
version
------------------------------------------------------------------------
PostgreSQL 10.0 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5
20150623 (Red Hat 4.8.5-16), 64-bit
(1 row)
postgres=#
```



VPN

- ✓ A VPN client has to be installed on a workstation




```
sudo openvpn --config client-config.ovpn
```

SSL CERTIFICATE

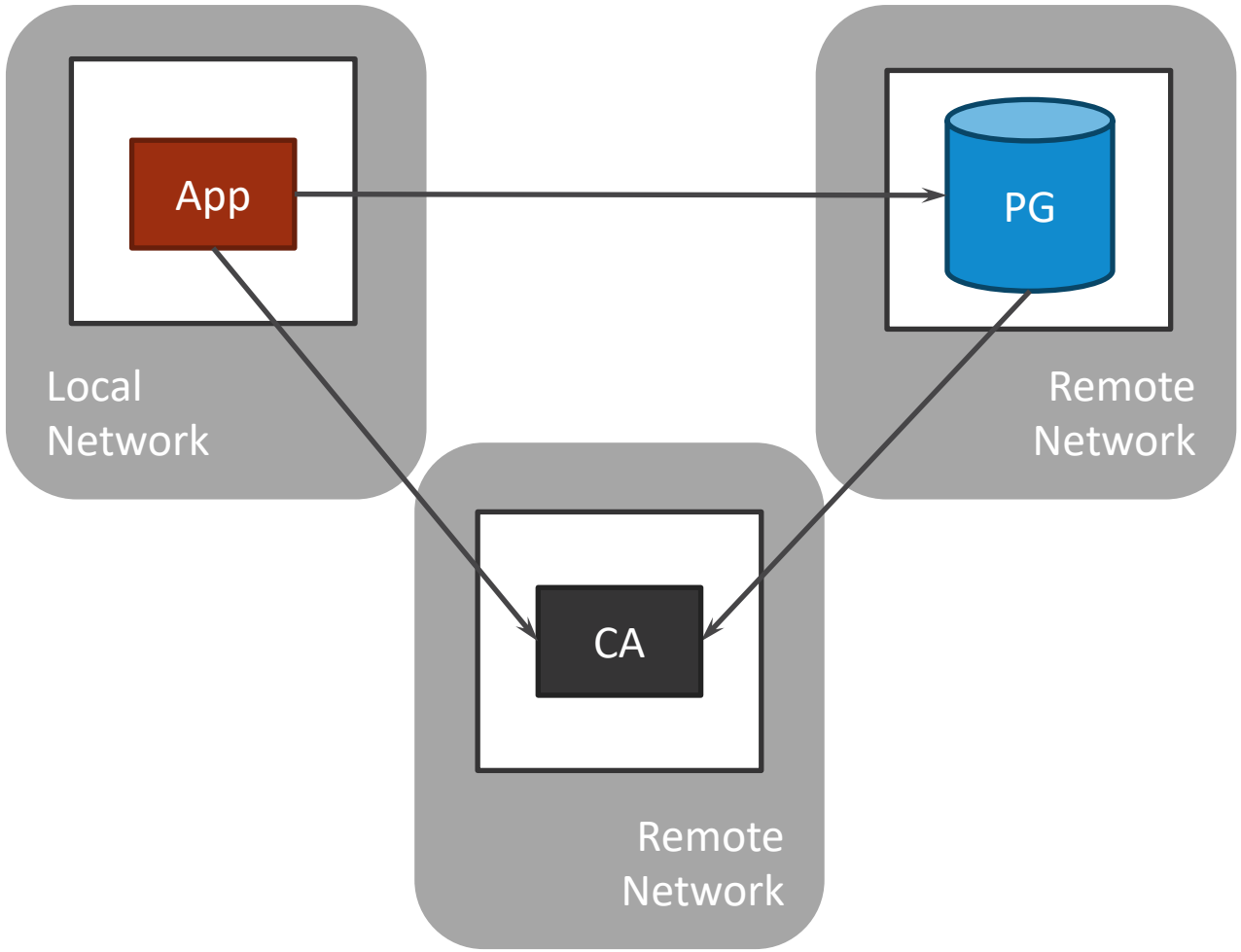
SSL USAGE

- Traffic encryption
- User authentication
- Both options

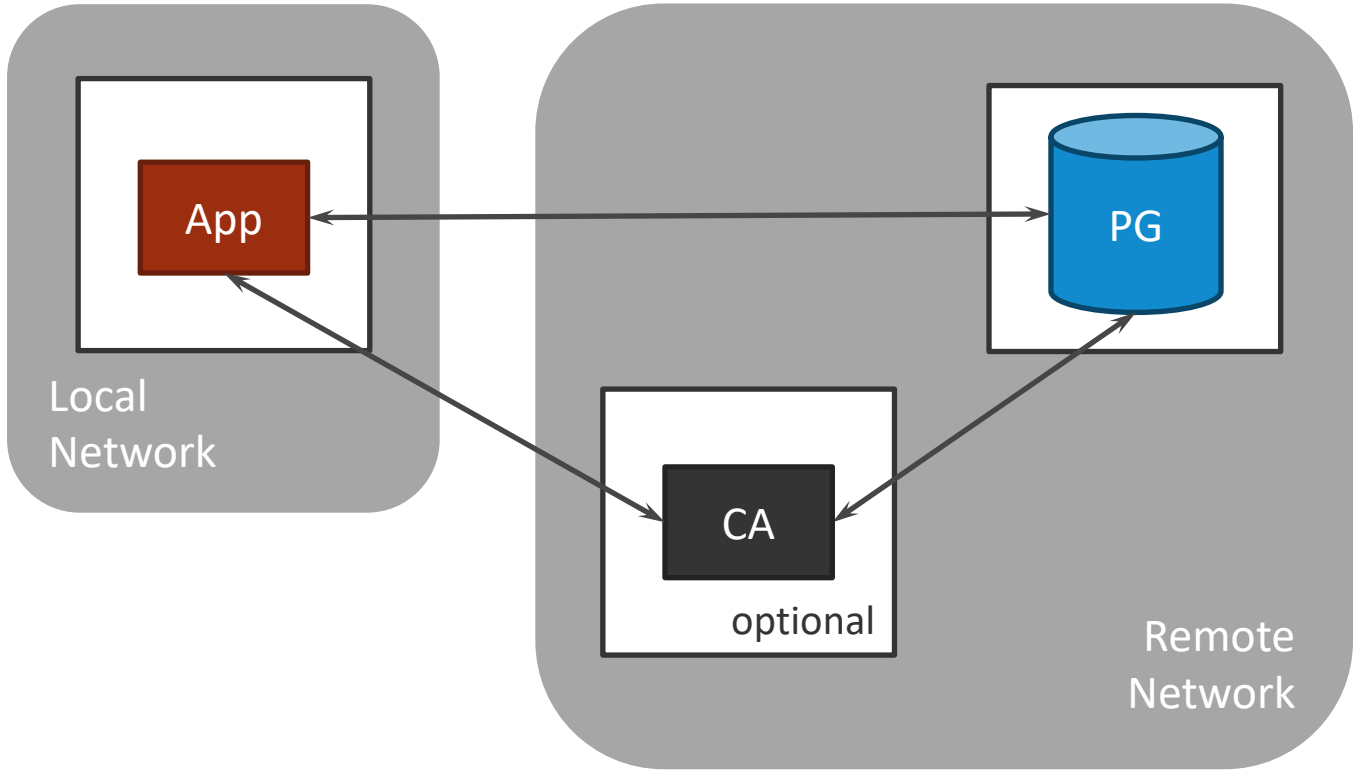
SSL MODES

- Non-validating mode (not safe)
- Server root certificate provided explicitly
- “Official” CA certificate is used

SSL: OFFICIAL CERTIFICATE



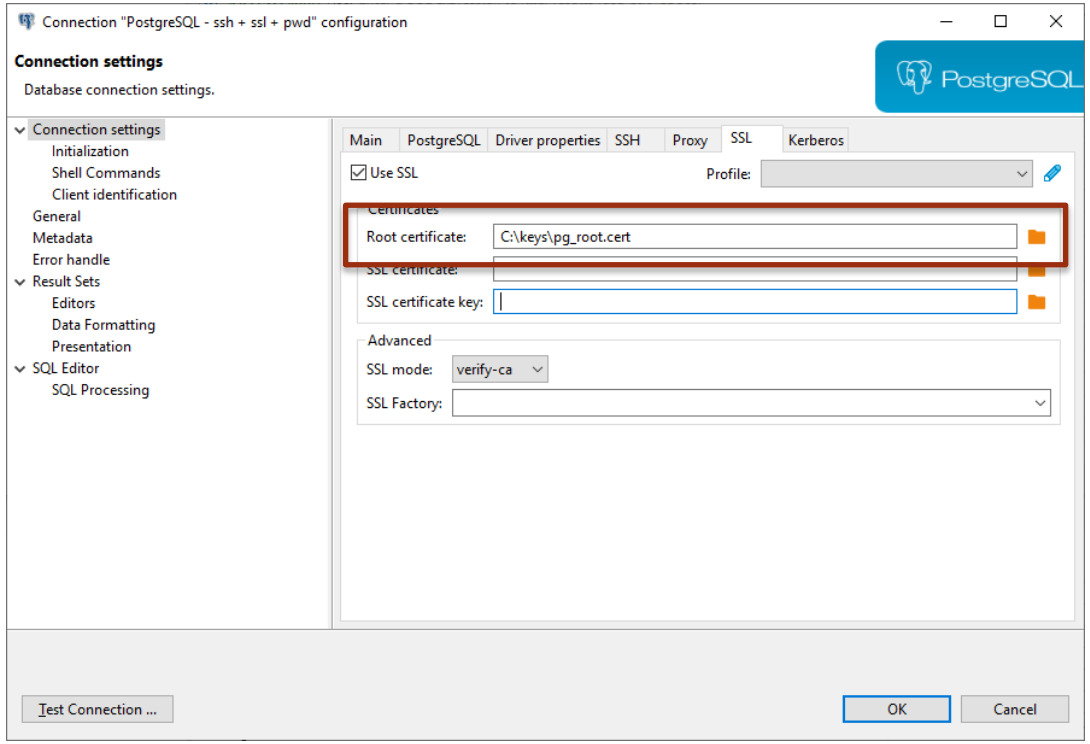
SSL: LOCAL SERVER CERTIFICATION



SSL: LOCAL SERVER CERTIFICATION

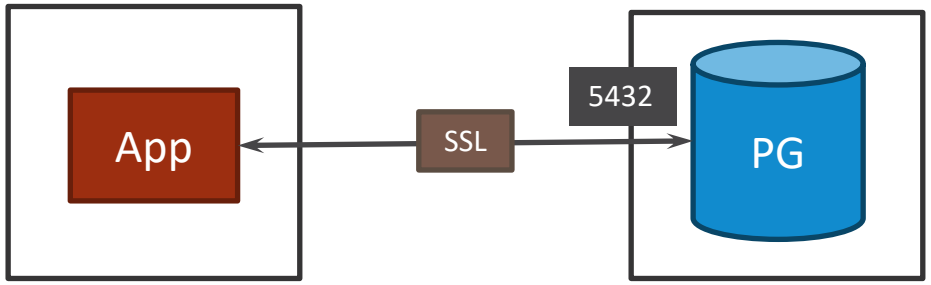
- ✓ Place SSL certificate for pg.test.com to ~/.postgresql/root.crt

```
psql -h pg.test.com -U postgres -W "sslmode=verify-full dbname=postgres"
```



SSL: NO CERTIFICATE VALIDATION

Not recommended for production systems



AUTHENTICATION

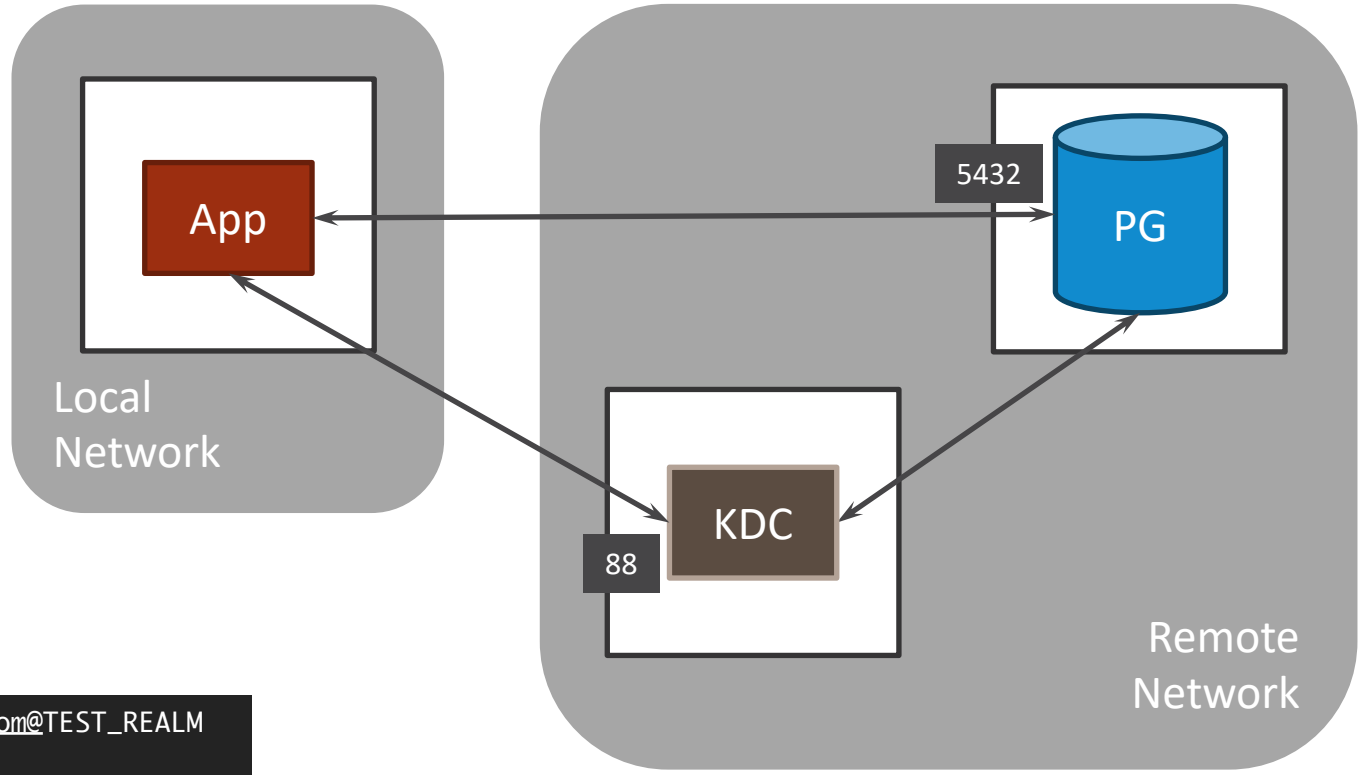
AUTHENTICATION SCHEMAS

- Trust: local network hosts
- Peer: local machine
- Indent: OS-based, local network hosts
- Password based
  - LDAP
  - RADIUS
  - PAM (Pluggable Authentication Modules)
- SSL certificates
- SSO: GSSAPI, SSPI, Kerberos

KERBEROS

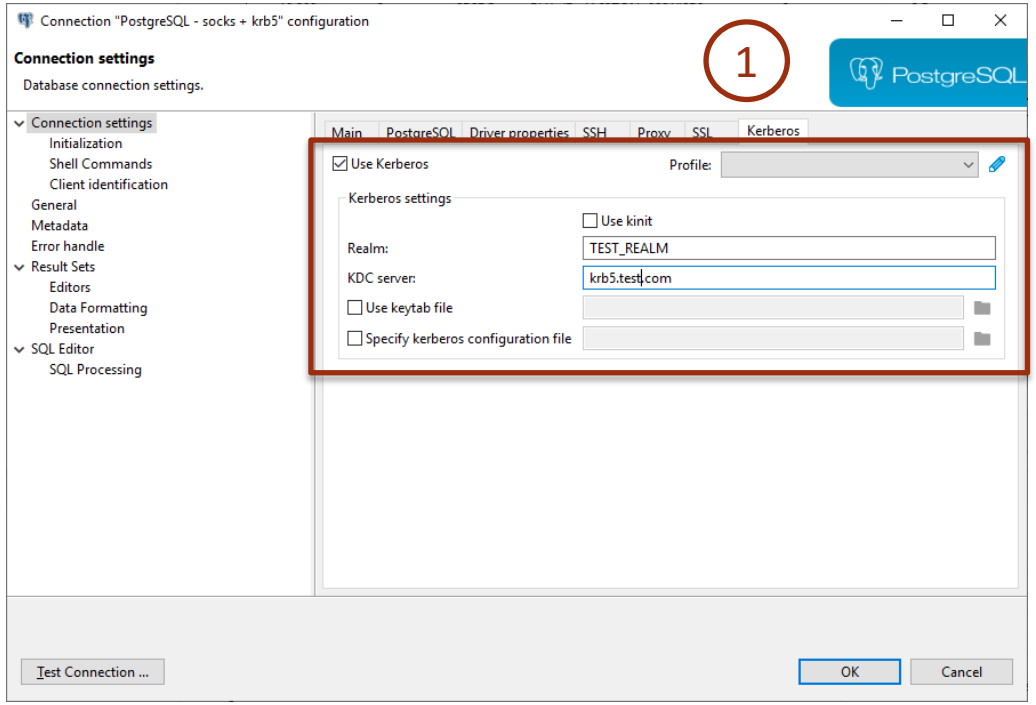
krb5.conf

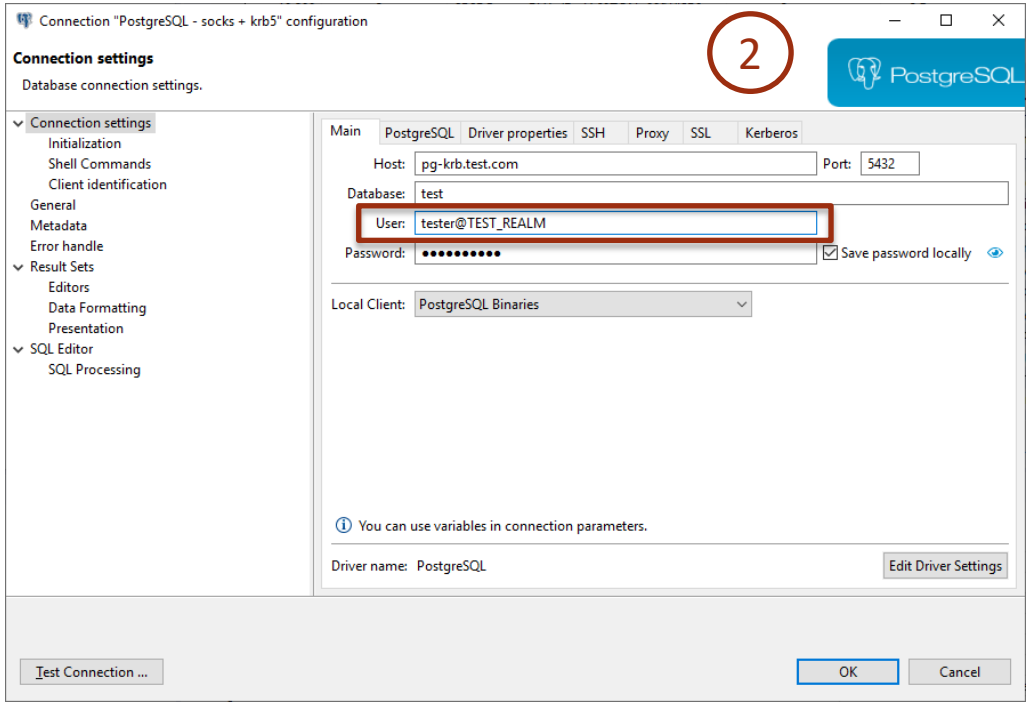
```
[libdefaults]
    default_realm = TEST_REALM
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ticket_lifetime = 48h
    renew_lifetime = 7d
    forwardable = yes

[realms]
    TEST_REALM = {
        kdc = krb5.test.com
        admin_server = krb5.test.com
        default_domain = test.com
    }

[domain_realm]
    test.com = TEST_REALM
```



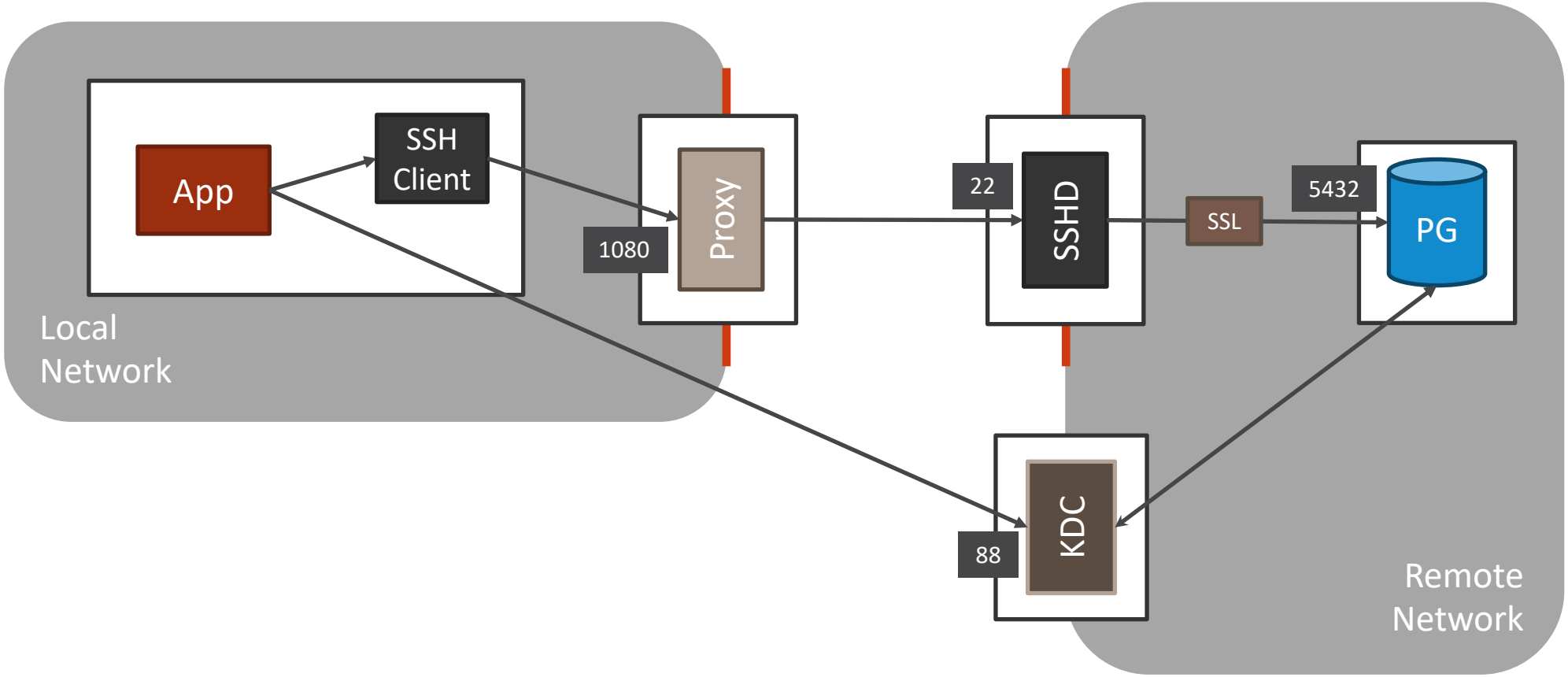

```
kinit -k -t pg.test.com-TEST_REALM.keytab postgres/pg.test.com@TEST_REALM

psql -U user/pg.test.com@TEST_REALM -h pg.test.com database_name
```

KERBEROS


Connection "PostgreSQL - socks + krb5" configuration
1
Connection settings
Database connection settings.
PostgreSQL
Connection settings
Initialization
Shell Commands
Client identification
General
Metadata
Error handle
Result Sets
Editors
Data Formatting
Presentation
SQL Editor
SQL Processing
Main
PostgreSQL
Driver properties
SSH
Proxy
SSL
Kerberos
Use Kerberos
Profile:
Kerberos settings
Use kinit
Realm:
TEST_REALM
KDC server:
krb5.test.com
Use keytab file
Specify kerberos configuration file
Test Connection ...
OK
Cancel


Connection "PostgreSQL - socks + krb5" configuration
2
Connection settings
Database connection settings.
PostgreSQL
Connection settings
Initialization
Shell Commands
Client identification
General
Metadata
Error handle
Result Sets
Editors
Data Formatting
Presentation
SQL Editor
SQL Processing
Main
PostgreSQL
Driver properties
SSH
Proxy
SSL
Kerberos
Host:
pg-krb.test.com
Port:
5432
Database:
test
User:
tester@TEST_REALM
Password:
Save password locally
Local Client:
PostgreSQL Binaries
You can use variables in connection parameters.
Driver name:
PostgreSQL
Edit Driver Settings
Test Connection ...
OK
Cancel

ALL TOGETHER



USEFUL LINKS

FOLLOW US

- Twitter: https://twitter.com/dbeaver_news
- GitHub: https://github.com/dbeaver/
- DBeaver EE: https://dbeaver.com/
- DBeaver CE: https://dbeaver.io/

CONTACT US

- General questions: tati@dbeaver.com
- Technical questions: serge@dbeaver.com