Subject : cryptography

PostgreSQL authentication with Kerberos

Created by :

Jaouadi Haroun

and

Masghouni Mohamed Sadok

Introduction :

Kerberos is a protocol for authenticating users on a network, allowing them to securely establish their identities over an insecure network. The protocol is an established industry standard, with messages that are designed to resist spying and replay attacks. Numerous software applications, including Chrome, Firefox, OpenSSH, Putty, OpenLDAP, Thunderbird, and PostgreSQL, have integrated Kerberos into their systems. There are also open source implementations, such as krb5 developed by MIT, which is used by most Unix-like operating systems. To ensure a proper understanding of Kerberos, it is important to first become familiar with some key concepts before diving into the environment setup.

-Realm refers to a domain or group to which all users and servers belong, and is a required component during Kerberos installation. For instance, in this blog, the realm used is HIGHGO.CA, which can be changed as per the user's requirements.

-Principal : refers to any user or service defined in Kerberos.

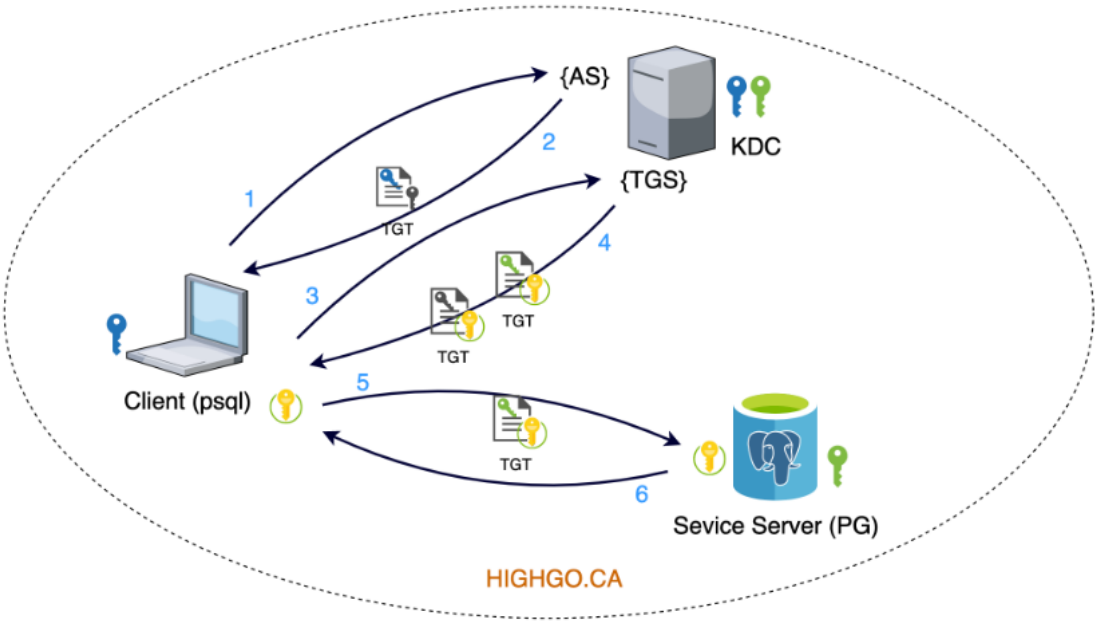
-SS (Service Server) : refers to a server that provides services. For instance, app.realm.org provides PostgreSQL database access service.

-KDC (Key Distribution Center): comprises one database of all principals and two components: AS (Authentication Server), which is responsible for the initial authentication request from users triggered by kinit, and TGS (Ticket Granting Server), which assigns the requested resource on a Service Server to the users. In this blog, both AS and TGS are deployed on the same KDC server, i.e., kdc.highgo.ca.

-TGT (Ticket Granting Ticket): is a message used to confirm the identity of the principals and to deliver session keys that are used for future secured communication among user, TGS, and SS.

-Keytab : is a file extracted from the KDC principal database that contains the encryption key for a service or host. For instance, postgres.keytab is the keytab file that will be used on the PostgreSQL server

-Client : refers to a workstation that needs to access a Service Server. For example, psql running on a client machine that wants to connect to the PostgreSQL server.



We will use three ubuntu virtual machines :

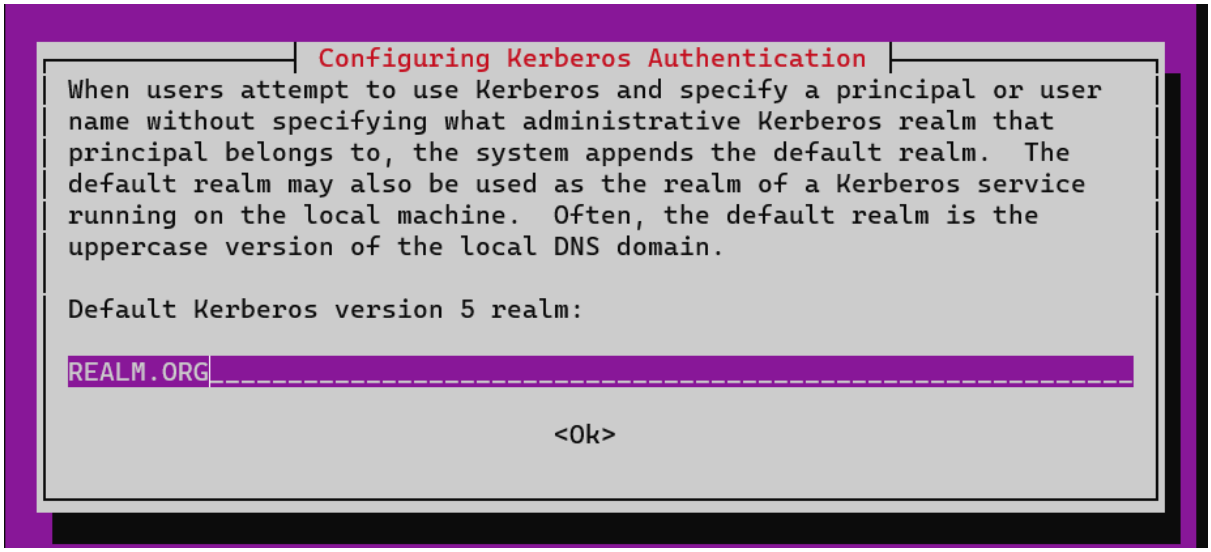
- App: the machine hosting the postgresql service
- Server : the kdc server
- Client : the client that will try to access to the postgresql service

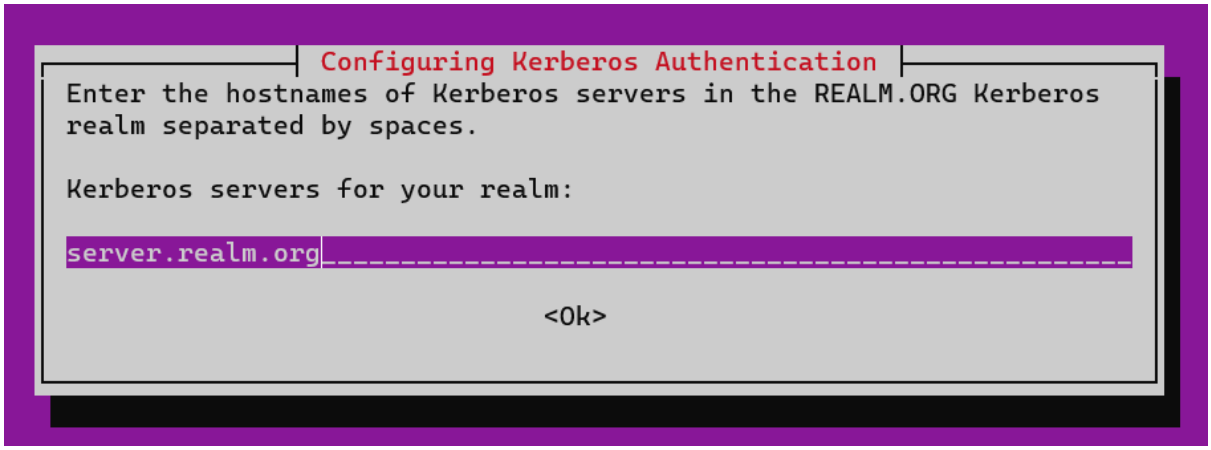
First we must configure our /etc/hosts file :

```
root@server:~# cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       ubuntu.myguest.virtualbox.org   ubuntu

192.168.134.186 client.realm.org client
192.168.134.174 app.realm.org app
192.168.134.124 server.realm.org server

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Install packages needed to configure the client and the app(service postgresql) machine:

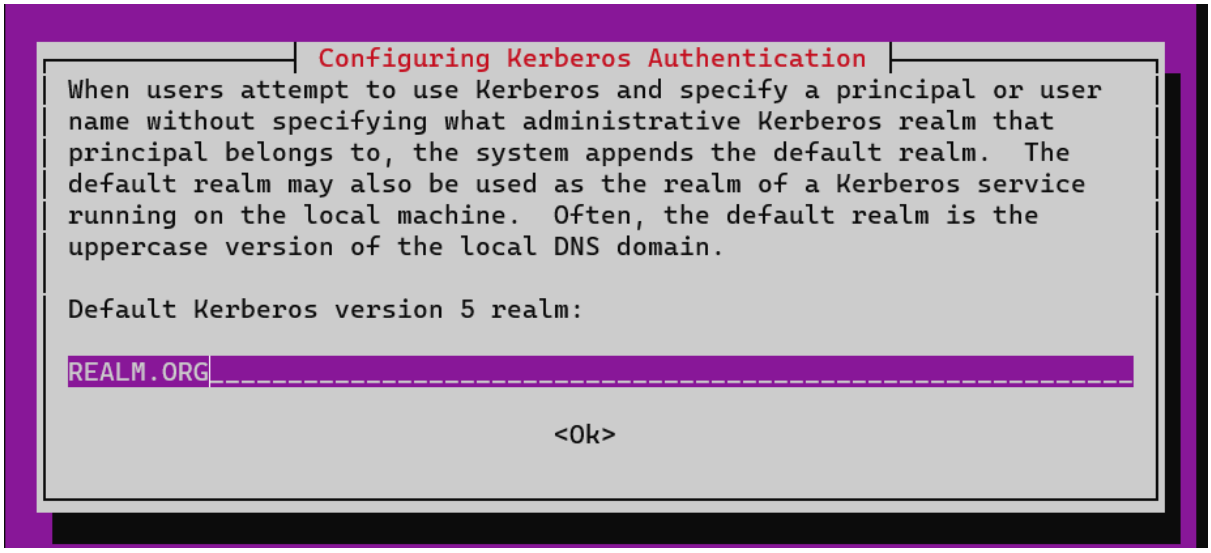
```
root@app:~# sudo apt install krb5-user libpam-krb5 libpam-ccreds auth-client
-config
```

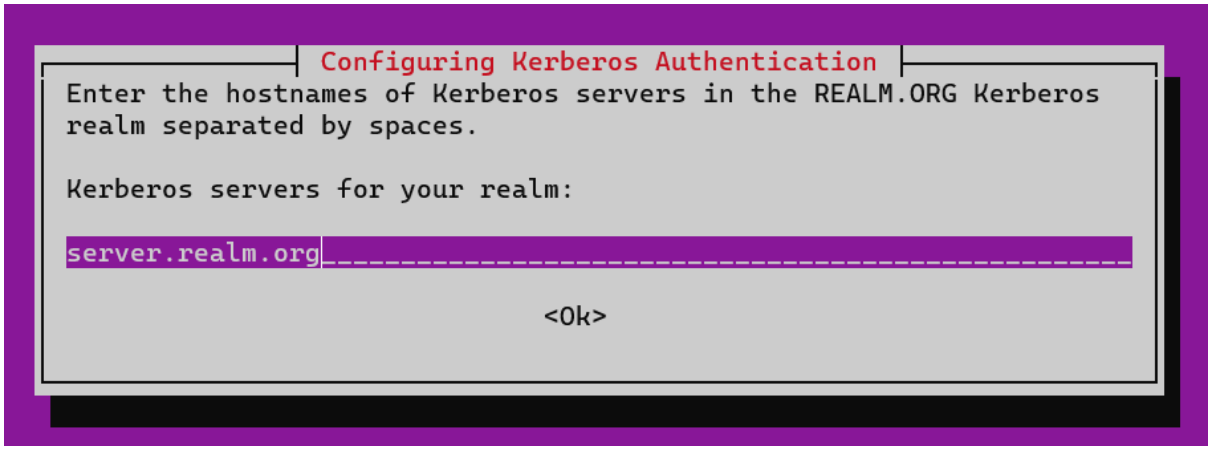


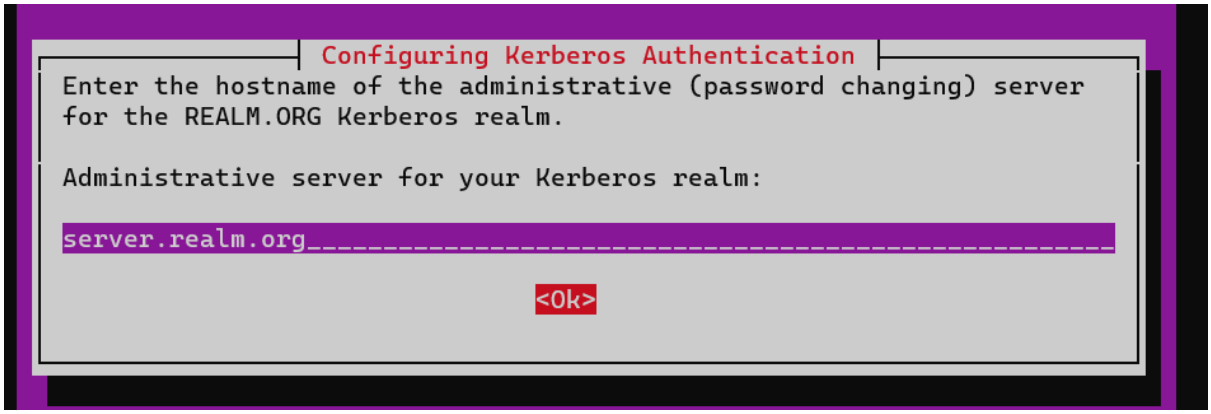


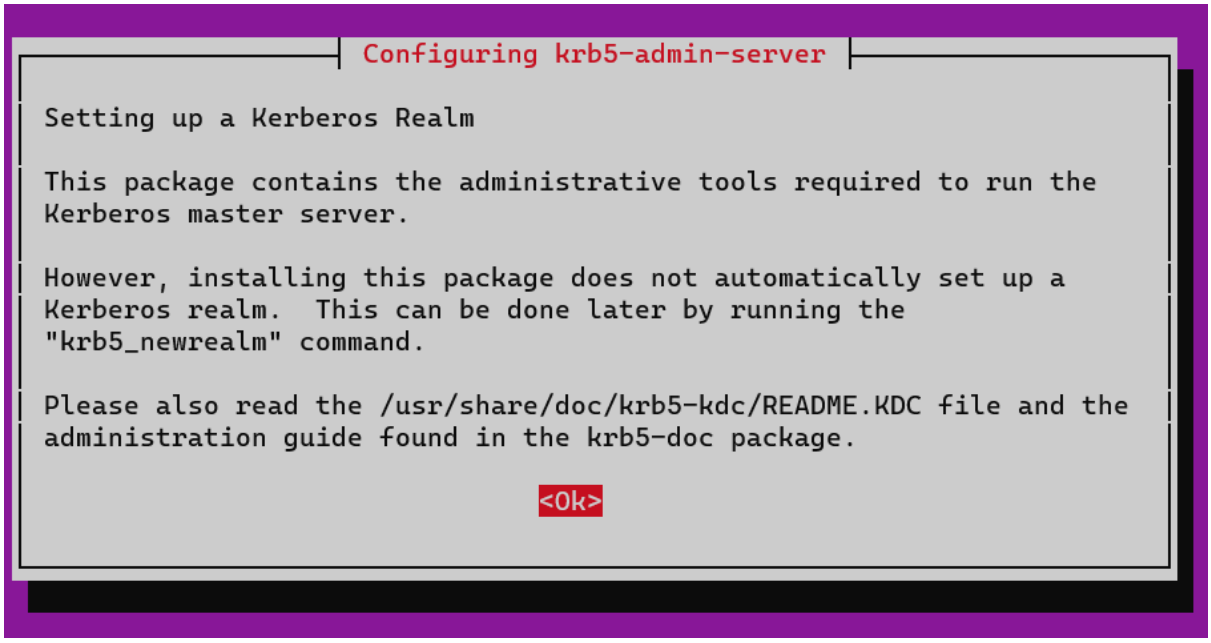
Install packages needed to configure the KDC machine (server.realm.org) :

```
sudo apt install krb5-kdc krb5-admin-server
```






Configuring Kerberos Authentication
Enter the hostname of the administrative (password changing) server
for the REALM.ORG Kerberos realm.
Administrative server for your Kerberos realm:
server.realm.org
<Ok>


Configuring krb5-admin-server
Setting up a Kerberos Realm
This package contains the administrative tools required to run the
Kerberos master server.
However, installing this package does not automatically set up a
Kerberos realm. This can be done later by running the
"krb5_newrealm" command.
Please also read the /usr/share/doc/krb5-kdc/README.KDC file and the
administration guide found in the krb5-doc package.
<Ok>

Now we will check the file /etc/krb5.conf :

```
root@server:~# cat /etc/krb5.conf
[libdefaults]
        default_realm = REALM.ORG

# The following krb5.conf variables are only for MIT Kerberos.
        kdc_timesync = 1
        ccache_type = 4
        forwardable = true
        proxiable = true

# The following encryption type specification will be used by MIT Kerberos
# if uncommented.  In general, the defaults in the MIT Kerberos code are
# correct and overriding these specifications only serves to disable new
# encryption types as they are added, creating interoperability problems.
#
# The only time when you might need to uncomment these lines and change
# the enctypes is if you have local software that will break on ticket
# caches containing ticket encryption types it doesn't know about (such as
# old versions of Sun Java).

#       default_tgs_enctypes = des3-hmac-sha1
#       default_tkt_enctypes = des3-hmac-sha1
#       permitted_enctypes = des3-hmac-sha1

# The following libdefaults parameters are only for Heimdal Kerberos.
        fcc-mit-ticketflags = true

[realms]
        REALM.ORG = {
                kdc = server.realm.org
                admin_server = server.realm.org
        }
        ATHENA.MIT.EDU = {
                kdc = kerberos.mit.edu
                kdc = kerberos-1.mit.edu
                kdc = kerberos-2.mit.edu:88
```

Also we must check the /etc/krb5kdc/kdc.conf to verify the kdc configuration :

```
root@server:~# cat /etc/krb5kdc/kdc.conf
[kdcdefaults]
    kdc_ports = 750,88

[realms]
    REALM.ORG = {
        database_name = /var/lib/krb5kdc/principal
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        acl_file = /etc/krb5kdc/kadm5.acl
        key_stash_file = /etc/krb5kdc/stash
        kdc_ports = 750,88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des3-hmac-sha1
        #supported_enctypes = aes256-cts:normal aes128-cts:normal
        default_principal_flags = +preauth
    }
root@server:~#
```

Now we create our realm : to set up the master key for KDC database.

```
root@server:~# krb5_newrealm
This script should be run on the master KDC/admin server to initialize
a Kerberos realm.  It will ask you to type in a master key password.
This password will be used to generate a key that is stored in
/etc/krb5kdc/stash.  You should try to remember this password, but it
is much more important that it be a strong password than that it be
remembered.  However, if you lose the password and /etc/krb5kdc/stash,
you cannot decrypt your Kerberos database.
Loading random data
Initializing database '/var/lib/krb5kdc/principal' for realm 'REALM.ORG',
master key name 'K/M@REALM.ORG'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:


Now that your realm is set up you may wish to create an administrative
principal using the addprinc subcommand of the kadmin.local program.
Then, this principal can be added to /etc/krb5kdc/kadm5.acl so that
you can use the kadmin program on other computers.  Kerberos admin
principals usually belong to a single user and end in /admin.  For
example, if jruser is a Kerberos administrator, then in addition to
the normal jruser principal, a jruser/admin principal should be
created.

Don't forget to set up DNS information so your clients can find your
KDC and admin servers.  Doing so is documented in the administration
guide.
root@server:~#
```

Next, we need to assign the appropriate access control list to the admin user in the Kerberos configuration file kadm5.acl :

```
  GNU nano 4.8                        kadm5.acl                          Modified
# This file Is the access control list for krb5 administration.
# When this file is edited run service krb5-admin-server restart to activate
# One common way to set up Kerberos administration is to allow any principa>
# ending in /admin  is given full administrative rights.
# To enable this, uncomment the following line:
# */admin *
*/admin@realm.org *
```

We need to create an admin user to manage principals, and it is recommended to use a different username :

```
root@server:~# kadmin.local
Authenticating as principal root/admin@REALM.ORG with password.
kadmin.local:  ?
Available kadmin.local requests:

add_principal, addprinc, ank
                         Add principal
delete_principal, delprinc
                         Delete principal
modify_principal, modprinc
                         Modify principal
rename_principal, renprinc
                         Rename principal
change_password, cpw     Change password
get_principal, getprinc  Get principal
list_principals, listprincs, get_principals, getprincs
                         List principals
add_policy, addpol       Add policy
modify_policy, modpol    Modify policy
delete_policy, delpol    Delete policy
get_policy, getpol       Get policy
list_policies, listpols, get_policies, getpols
                         List policies
get_privs, getprivs      Get privileges
ktadd, xst               Add entry(s) to a keytab
ktremove, ktrem          Remove entry(s) from a keytab
lock                     Lock database exclusively (use with extreme caution
!)
unlock                   Release exclusive database lock
purgekeys                Purge previously retained old keys from a principal
get_strings, getstrs     Show string attributes on a principal
set_string, setstr       Set a string attribute on a principal
del_string, delstr       Delete a string attribute on a principal
list_requests, lr, ?     List available requests.
quit, exit, q            Exit program.
kadmin.local:
```

To see all principals :

```
kadmin.local:  list_principals
K/M@REALM.ORG
kadmin/admin@REALM.ORG
kadmin/changepw@REALM.ORG
kadmin/server.realm.org@REALM.ORG
kiprop/server.realm.org@REALM.ORG
krbtgt/REALM.ORG@REALM.ORG
```

Add a principal “utilisateur” for Client, this is the login user for Client OS, and later will be used to log into database :

```
kadmin.local:  addprinc utilisateur
WARNING: no policy specified for utilisateur@REALM.ORG; defaulting to no pol
icy
Enter password for principal "utilisateur@REALM.ORG":
Re-enter password for principal "utilisateur@REALM.ORG":
Principal "utilisateur@REALM.ORG" created.
kadmin.local:  listprincs
K/M@REALM.ORG
kadmin/admin@REALM.ORG
kadmin/changepw@REALM.ORG
kadmin/server.realm.org@REALM.ORG
kiprop/server.realm.org@REALM.ORG
krbtgt/REALM.ORG@REALM.ORG
postgres/app.realm.org@REALM.ORG
postgres@REALM.ORG
root/admin@REALM.ORG
utilisateur@REALM.ORG
```

To see information about a principle :

```
kadmin.local:  get_principal utilisateur
Principal: utilisateur@REALM.ORG
Expiration date: [never]
Last password change: Sat Apr 29 04:44:53 WAT 2023
Password expiration date: [never]
Maximum ticket life: 0 days 10:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Sat Apr 29 04:44:53 WAT 2023 (root/admin@REALM.ORG)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, aes256-cts-hmac-sha1-96
Key: vno 1, aes128-cts-hmac-sha1-96
MKey: vno 1
Attributes: REQUIRES_PRE_AUTH
Policy: [none]
kadmin.local:
```

```
kadmin.local:  add_principal root/admin
WARNING: no policy specified for root/admin@REALM.ORG; defaulting to no poli
cy
Enter password for principal "root/admin@REALM.ORG":
Re-enter password for principal "root/admin@REALM.ORG":
Principal "root/admin@REALM.ORG" created.
kadmin.local:
```

We need to create an admin user to manage principals, and it is recommended to use a different username. In our case, root/admin. Below are the commands used for the setup.

```
root@server:~# kinit root/admin
Password for root/admin@REALM.ORG:
root@server:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: root/admin@REALM.ORG

Valid starting       Expires              Service principal
29/04/2023 06:14:16  29/04/2023 16:14:16  krbtgt/REALM.ORG@REALM.ORG
        renew until 30/04/2023 06:14:00
root@server:~#
```

```
kadmin.local:  list_principals
K/M@REALM.ORG
kadmin/admin@REALM.ORG
kadmin/changepw@REALM.ORG
kadmin/server.realm.org@REALM.ORG
kiprop/server.realm.org@REALM.ORG
krbtgt/REALM.ORG@REALM.ORG
root/admin@REALM.ORG
utilisateur@REALM.ORG
kadmin.local:
```

Add a principal postgres/app.realm.org as a principle instance for Service server :

```
kadmin.local:  add_principal postgres/app.realm.org
WARNING: no policy specified for postgres/app.realm.org@REALM.ORG; defaultin
g to no policy
Enter password for principal "postgres/app.realm.org@REALM.ORG":
Re-enter password for principal "postgres/app.realm.org@REALM.ORG":
Principal "postgres/app.realm.org@REALM.ORG" created.
kadmin.local:
```

Check principals :

```
kadmin.local:  get_principal postgres/app.realm.org@REALM.ORG
Principal: postgres/app.realm.org@REALM.ORG
Expiration date: [never]
Last password change: Sat Apr 29 05:15:03 WAT 2023
Password expiration date: [never]
Maximum ticket life: 0 days 10:00:00
Maximum renewable life: 7 days 00:00:00
Last modified: Sat Apr 29 05:15:03 WAT 2023 (root/admin@REALM.ORG)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 2
Key: vno 1, aes256-cts-hmac-sha1-96
Key: vno 1, aes128-cts-hmac-sha1-96
MKey: vno 1
Attributes: REQUIRES_PRE_AUTH
Policy: [none]
kadmin.local:
```

Extract the service principal from KDC principal database to a keytab file, which will be used to configure PostgreSQL Server. The file should be saved to current folder when run below commands.

```
root@server:~# ktutil
ktutil:  add_entry -password -p postgres/app.realm.org@REALM.ORG -k 1 -e aes
256-cts-hmac-sha1-96
Password for postgres/app.realm.org@REALM.ORG:
ktutil:
```

```
Password for postgres/app.realm.org@REALM.ORG:
ktutil:  wkt /etc/krb5kdc/postgres.keytab
ktutil:  q
root@server:~# ls /etc/krb5kdc/
kadm5.acl  kdc.conf  postgres.keytab  stash
root@server:~#
```

Install postgresql on the app.realm.org machine (app) with gssapi :

```
root@app:~# sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sourc
es.list.d/pgdg.list'
root@app:~# wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
OK
root@app:~# sudo apt-get update
Hit:1 http://tn.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://tn.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://apt.postgresql.org/pub/repos/apt focal-pgdg InRelease [116 kB]
Get:5 http://tn.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://apt.postgresql.org/pub/repos/apt focal-pgdg/main amd64 Packages [262 kB]
Fetched 714 kB in 2s (448 kB/s)
Reading package lists... Done
N: Skipping acquire of configured file 'main/binary-i386/Packages' as repository 'http://apt.postgresql.org/pub/repos/ap
t focal-pgdg InRelease' doesn't support architecture 'i386'
root@app:~# sudo apt-get -y install postgresql
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libllvm10 libpq5 libtypes-serialiser-perl postgresql-15
  postgresql-client-15 postgresql-client-common postgresql-common sysstat
Suggested packages:
  postgresql-doc postgresql-doc-15 isag
The following NEW packages will be installed:
```

Copy the keytab file from kdc to the postgress(app) machine :

```
root@server:~# scp /etc/krb5kdc/postgres.keytab postgress@app.realm.org:/hom
e/postgress
The authenticity of host 'app.realm.org (192.168.1.53)' can't be established
.
ECDSA key fingerprint is SHA256:ZD2V5mvozb6JZohG6jY6PWI7moUpNfbmCMST/y4mSBQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'app.realm.org,192.168.1.53' (ECDSA) to the list
of known hosts.
postgress@app.realm.org's password:
postgres.keytab                               100%   93    28.4KB/s   00:00
root@server:~#
```

```
root@app:/etc/postgresql/15/main# cp /home/postgress/postgres.keytab /var/lib/postgresql
```

Checking the location of the keytab file :

```
root@app:/var/lib/postgresql# ll
total 20
drwxr-xr-x  3 postgres postgres 4096 May  2 02:48 ./
drwxr-xr-x 67 root     root     4096 May  1 17:17 ../
drwxr-xr-x  3 postgres postgres 4096 Apr 30 16:10 15/
-rw-------  1 postgres postgres   38 May  1 19:05 .bash_history
-rw-------  1 postgres root       93 May  2 02:48 postgres.keytab
-rw-------  1 postgres postgres    0 May  1 17:58 .psql_history
root@app:/var/lib/postgresql# chgrp postgres postgres.keytab
root@app:/var/lib/postgresql# ll
total 20
drwxr-xr-x  3 postgres postgres 4096 May  2 02:48 ./
drwxr-xr-x 67 root     root     4096 May  1 17:17 ../
drwxr-xr-x  3 postgres postgres 4096 Apr 30 16:10 15/
-rw-------  1 postgres postgres   38 May  1 19:05 .bash_history
-rw-------  1 postgres postgres   93 May  2 02:48 postgres.keytab
-rw-------  1 postgres postgres    0 May  1 17:58 .psql_history
root@app:/var/lib/postgresql#
```

Now we will relate the keytab file to postgres in postgresql.conf :

We also need Postgres Server to allow connection from the network by change the listen_addresses :

```
# GSSAPI using Kerberos
#krb_server_keyfile = 'FILE:${sysconfdir}/krb5.keytab'
#krb_caseins_users = off
krb_server_keyfile = '/var/lib/postgresql/postgres.keytab'
listen_addresses = '*'
```

We must verify that the value of the port is 5432

And now we read the keytab file and relate it to the principal

```
root@app:/var/lib/postgresql# ktutil
ktutil:  list
slot KVNO Principal
---- ---- ---------------------------------------------------------------
ktutil:  read_kt postgres.keytab
ktutil:  list
slot KVNO Principal
---- ---- ---------------------------------------------------------------
   1    1         postgres/app.realm.org@REALM.ORG
ktutil:
```

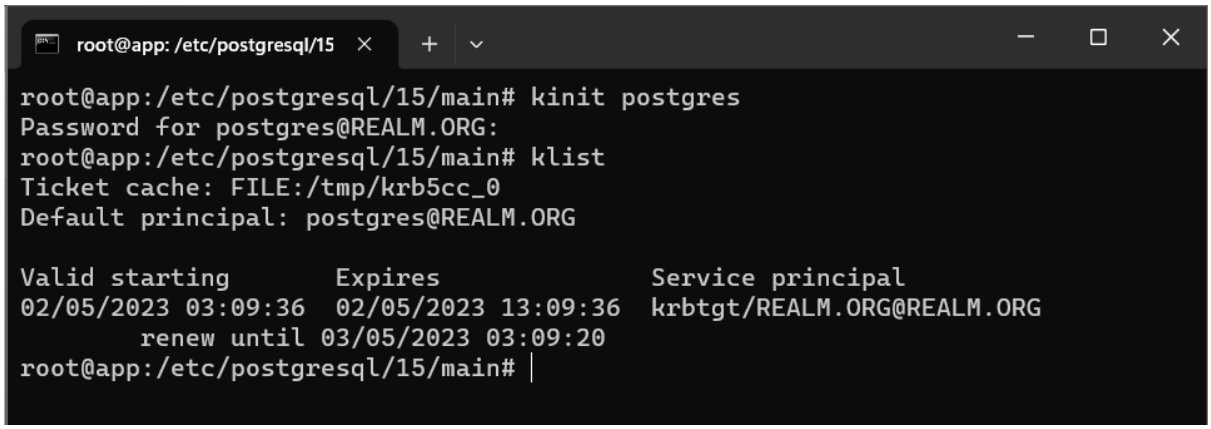
Now we give access to principals “postgress” and “utilisateur” host machines to access to postgress service by using Kerberos protocol

```
root@app:/etc/postgresql/15/main# nano pg_hba.conf
```



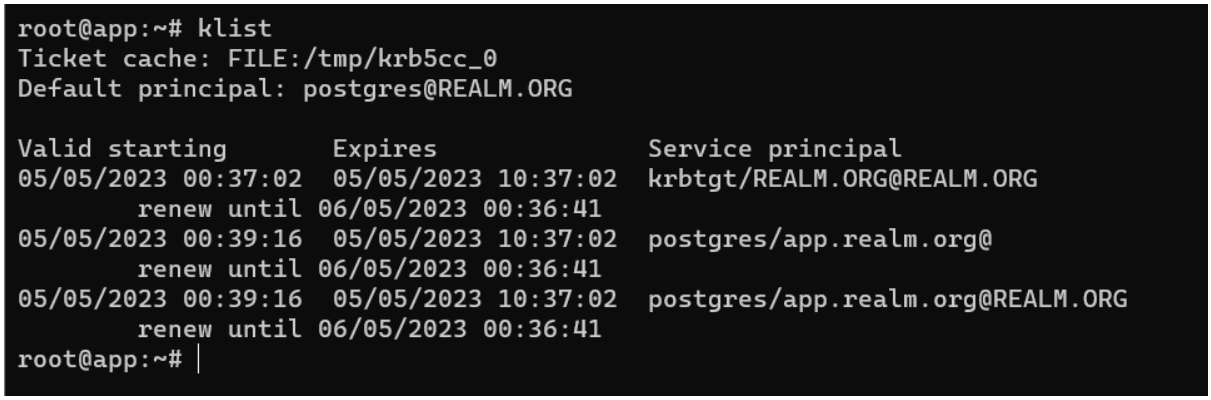
```
local   all             all                                     peer
# IPv4 local connections:
#host    all             all             127.0.0.1/32            scram-sha-256
host    all             all             127.0.0.1/32            trust
hostgssenc postgres     utilisateur           192.168.0.0/24          gss include_realm=0
hostgssenc postgres     postgres        192.168.0.0/24          gss include_realm=0
```

Try to connect to the kdc from postgres machine



Test of postgress access :

Check the cached credentials :



Now from the client machine :

Initialisation ;

```
root@client:~# kinit postgres
Password for postgres@REALM.ORG:
root@client:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: postgres@REALM.ORG

Valid starting       Expires              Service principal
05/05/2023 01:52:49  05/05/2023 11:52:49  krbtgt/REALM.ORG@REALM.ORG
        renew until 06/05/2023 01:52:32
root@client:~#
```

And now we access to postgres :

```
root@client:~# psql -d postgres -U postgres -h app.realm.org
psql (15.2 (Ubuntu 15.2-1.pgdg20.04+1))
GSSAPI-encrypted connection
Type "help" for help.

postgres=#
```

Finally we access successfully to the postgres service in the app machine using Kerberos .