Authentication Kerberos

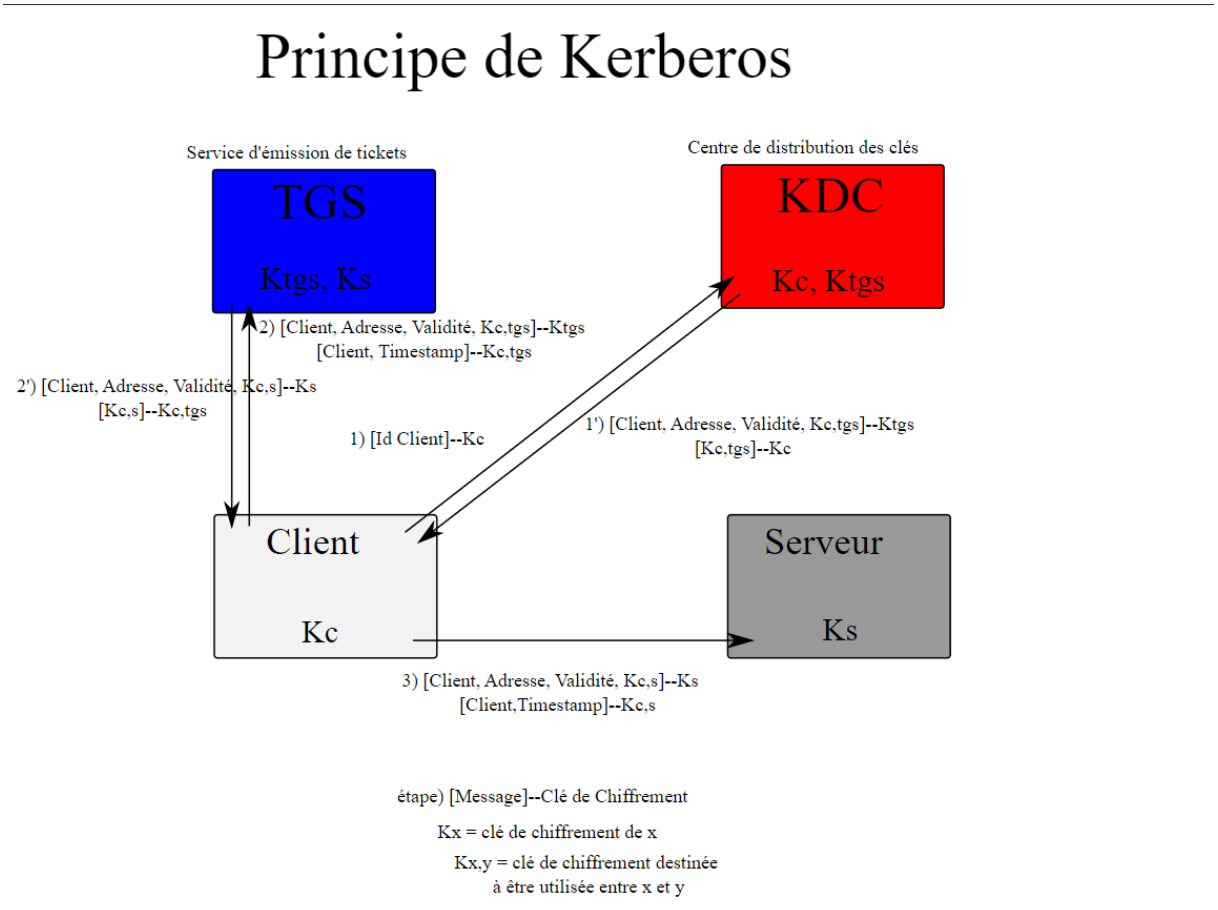
PostgreSQL Authentication with Kerberos

Enabling GSSAPI / Kerberos authentication in PostgreSQL will allow ***single-sign-on*** – i.e. authentication without using the standard username and password for PostgreSQL clients.

Steps To Setup Kerberos On UBUNTU

Kerberos is a network *authentication protocol* used to verify the identity of two or more *trusted hosts* across an *untrusted network*. It uses *secret-key cryptography* and a *trusted third party* (Kerberos Key Distribution Center) for authenticating client-server applications. Key Distribution Cente (KDC) gives clients tickets representing their network credentials. The Kerberos ticket is presented to the servers after the connection has been established

---

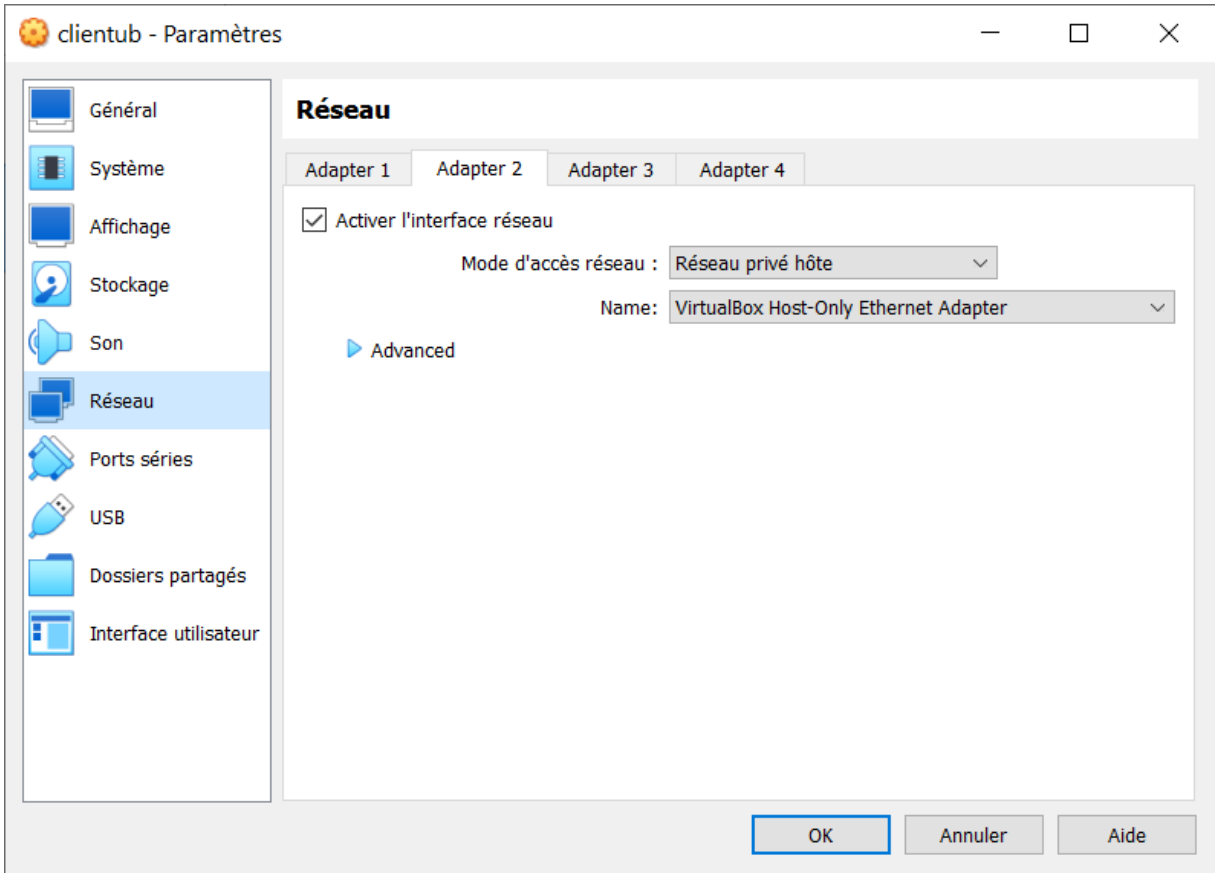


Hostname and IP Addresses

We will need three machines. In my case I'm using three *ubuntu* machines : my physical machine and two virtual machines inside of VirtualBox. My physical machine will be the client and the two other machines will be the Service Server and the KDC.

Virtual machines have a NAT adapter by default but in order to assign IP addresses to these machines we will need to add a host-only adapter manually.

Let's start by creating a new virtual machine. Under *File* go to *Host Network Manager ...* and then click *Create*.



Three machines :

Machine Name	Machine IP	Sub-domain name
KDC	192.168.56.109	Kdc.central.tn
Pg server	192.168.56.106	Pg.central.tn

Client	192.168.56.101	Client.central.tn

KDC Machine

```
clientclie@client:~/Bureau$ hostnamectl --static set-hostname kdc.central.tn
clientclie@client:~/Bureau$ hostname
kdc.central.tn
clientclie@client:~/Bureau$
```

Client machine

```
cli@client:~$ hostnamectl --static set-hostname client.central.tn
cli@client:~$ hostname
client.central.tn
cli@client:~$
```

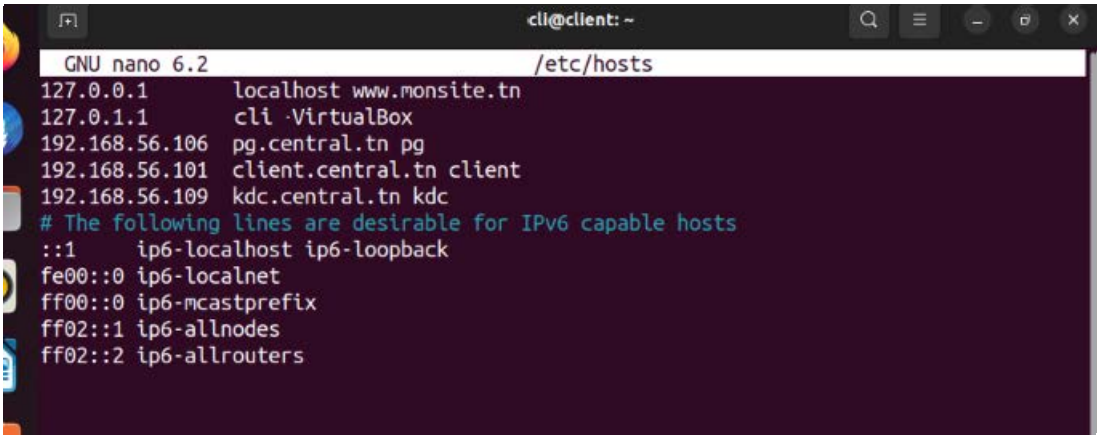
Service Server machine

```
ubuntu@kdc:~$ hostnamectl --static set-hostname pg.central.tn
ubuntu@kdc:~$ hostname
pg.central.tn
ubuntu@kdc:~$
```

<KDC_IP_ADDRESS> kdc.central.tn kdc

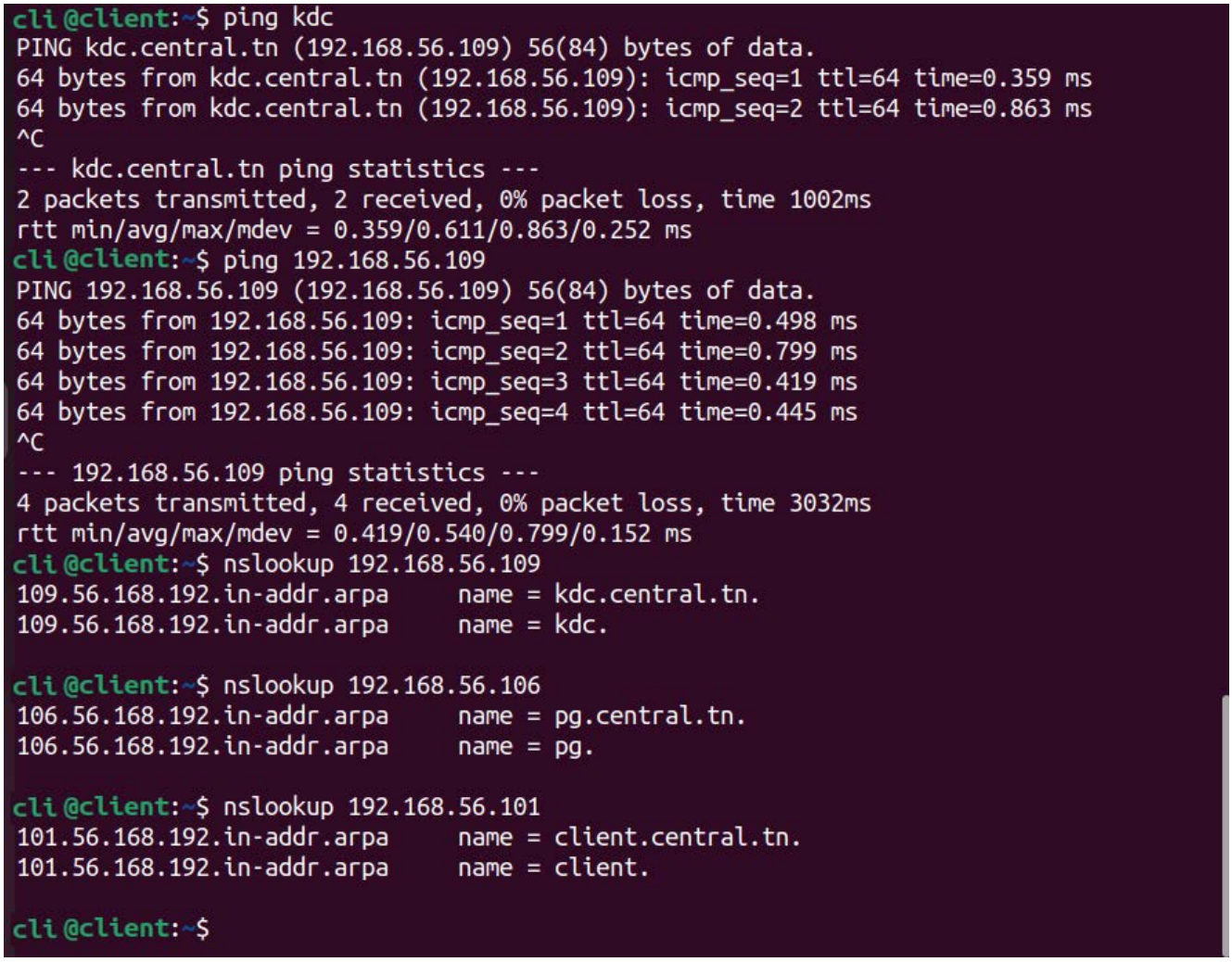
<PG_SERVER_ADDRESS> pg.central.tn pg

<CLIENT_ADDRESS> Client.central.tn client



Once the setup is done, we can check if everything is working fine by using the nslookup command to **query the DNS** to obtain the mapping we just did and the ping command to ensure that all three machines are reachable.

This an example in the client machine :



Key Distribution Center Machine Configuration

Following are the packages that need to installed on the KDC machine :

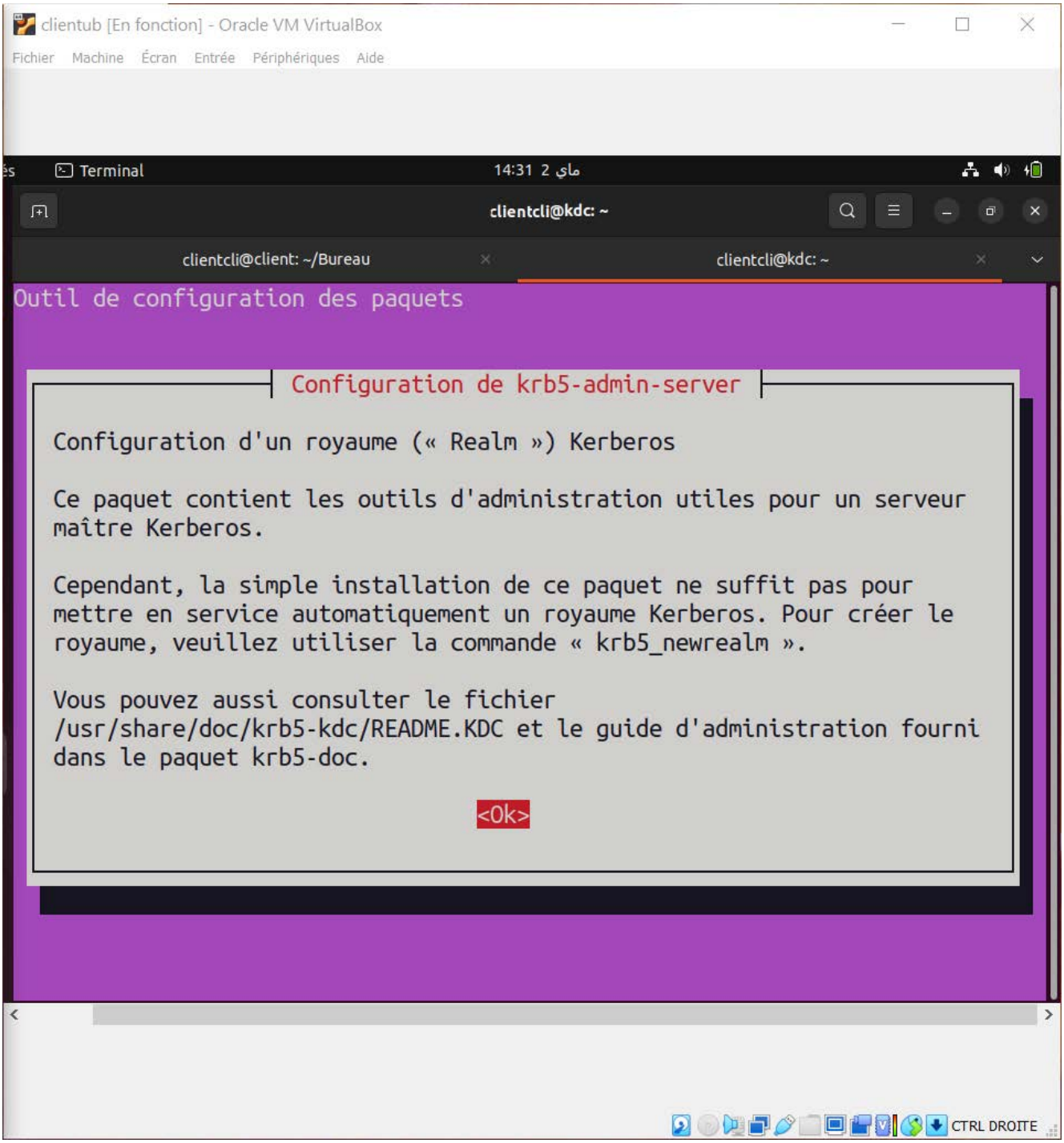
$ sudo apt-get update

$ sudo apt-get install krb5-kdc krb5-admin-server krb5-config

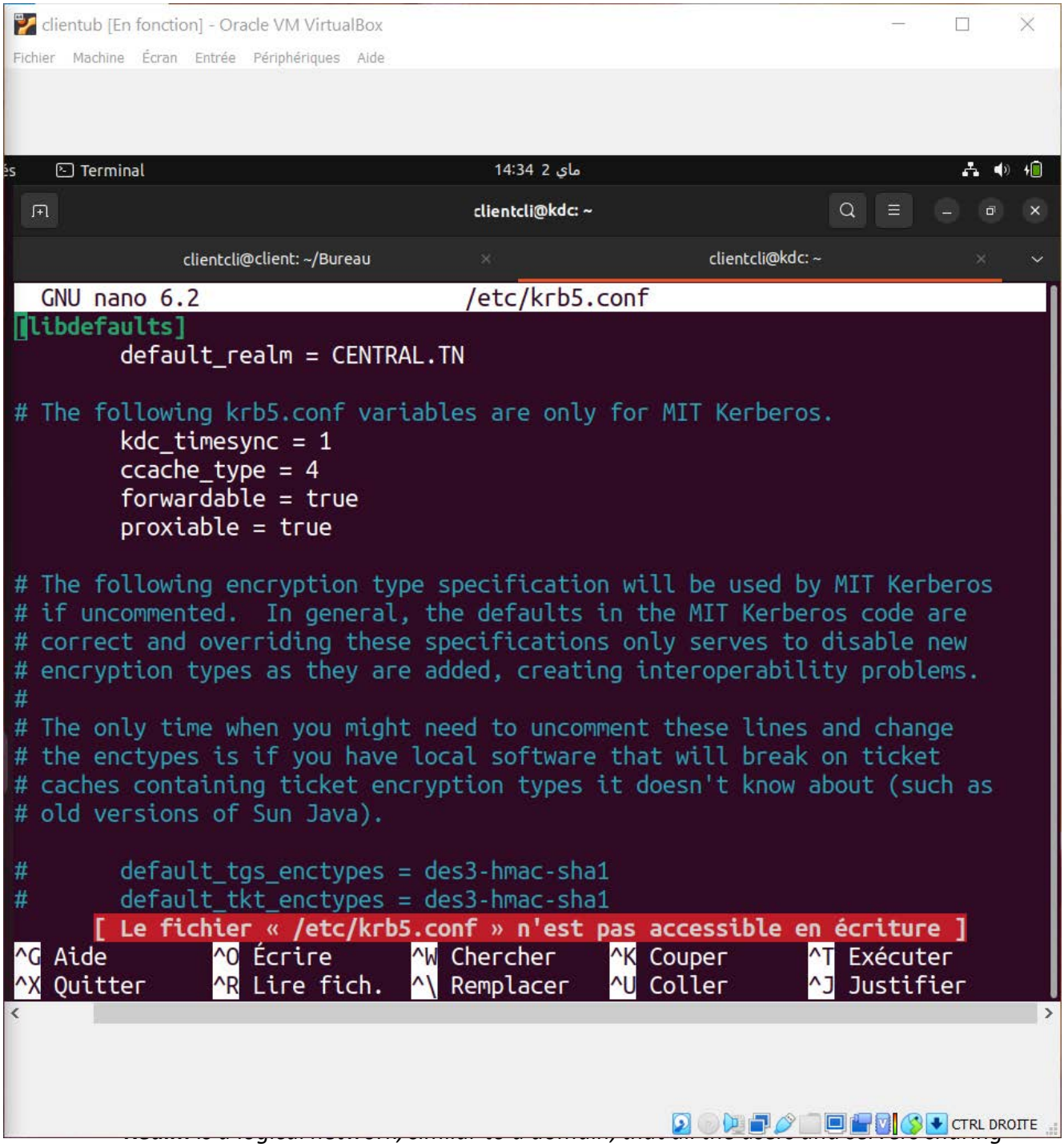
During the installation, we will be asked for configuration of :

the realm : 'CENTRAL.TN' (must be *all uppercase*)

Prompt	**value**
Realm	CENTRAL.TN
Kerberos servers	Kdc.central.tn
Administrative Service	Kdc.central.tn



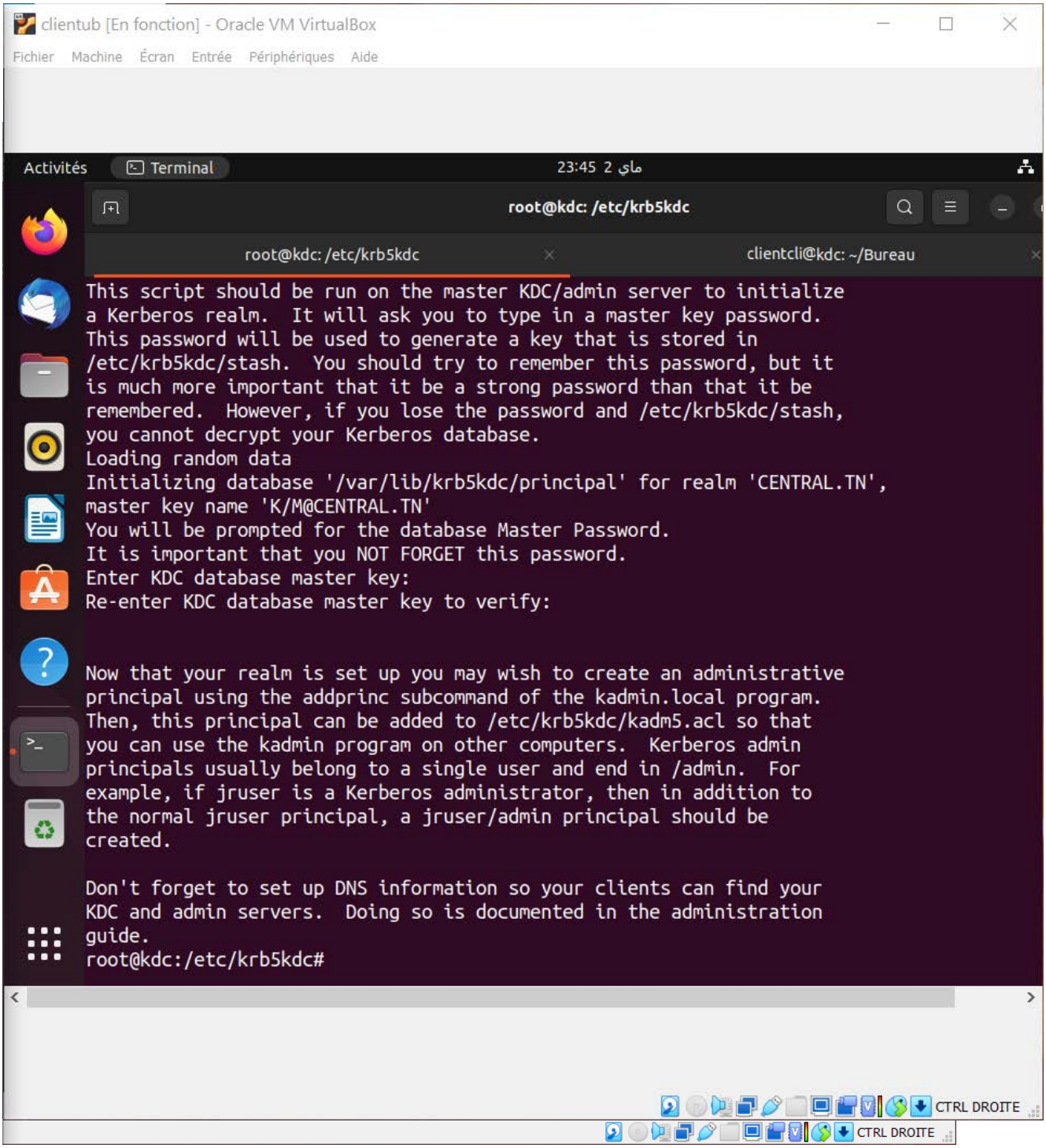
the realm : 'CENTRAL.TN' (must be *all uppercase*)



the same Kerberos database belong to.

The master key for this KDC database needs to be set once the installation is complete :

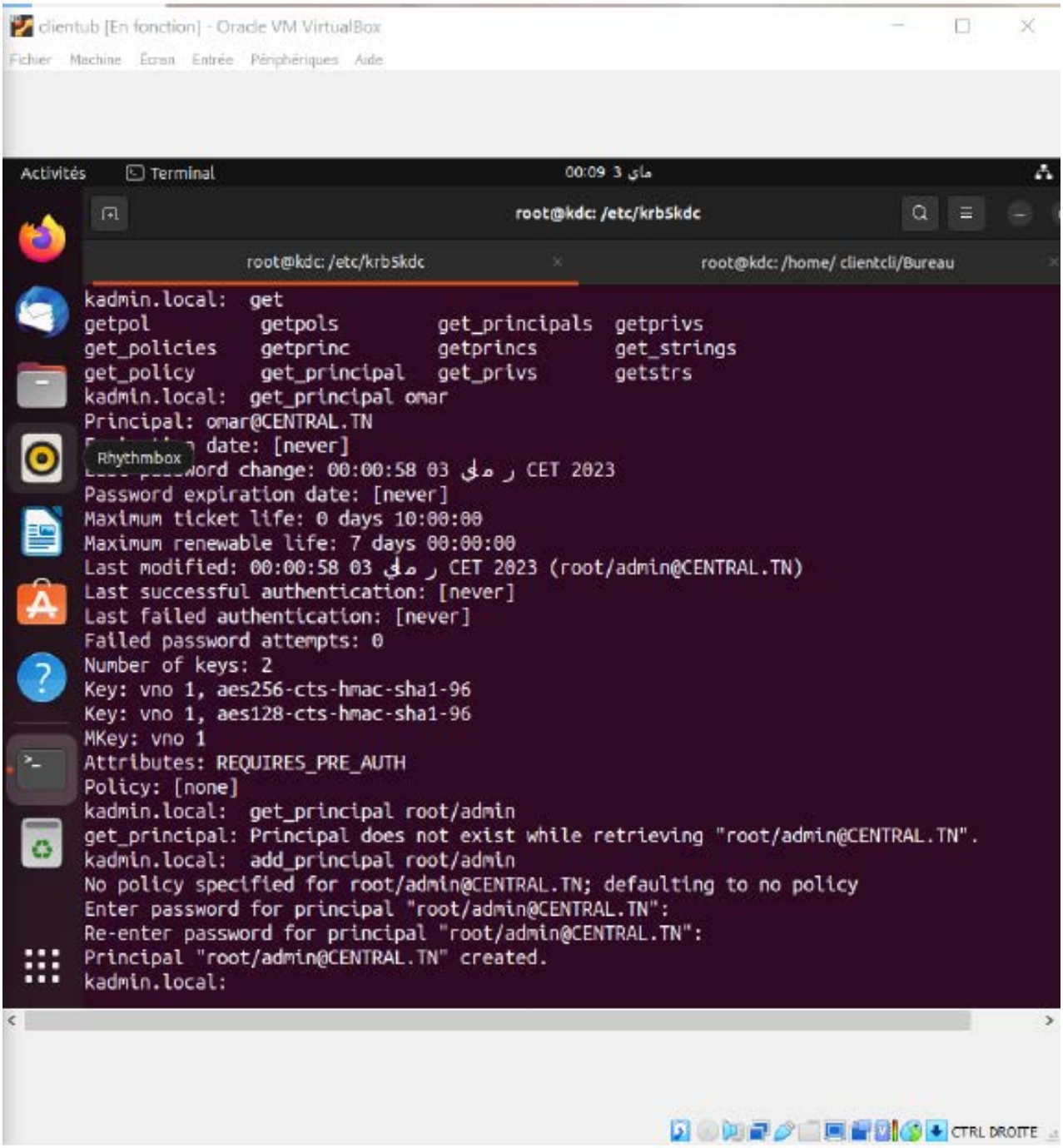
sudo krb5_newrealm



The users and services in a realm are defined as a ***principal*** *in Kerberos.* These principals are managed by an *admin user* that we need to create manually:

$ sudo kadmin.local

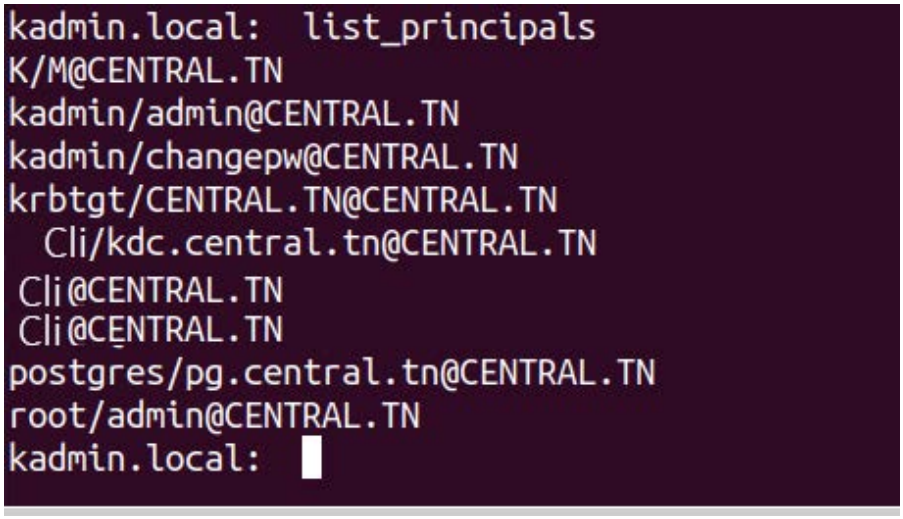
kadmin.local: add_principal root/admin



kadmin.local is a KDC database administration program. We used this tool to create a new principal in the INSAT.TN realm (add_principal).

We can check if the user *root/admin* was successfully created by running the command : kadmin.local: list_principals. We should see the 'root/admin@INSAT.TN' principal listed along with other default principals.

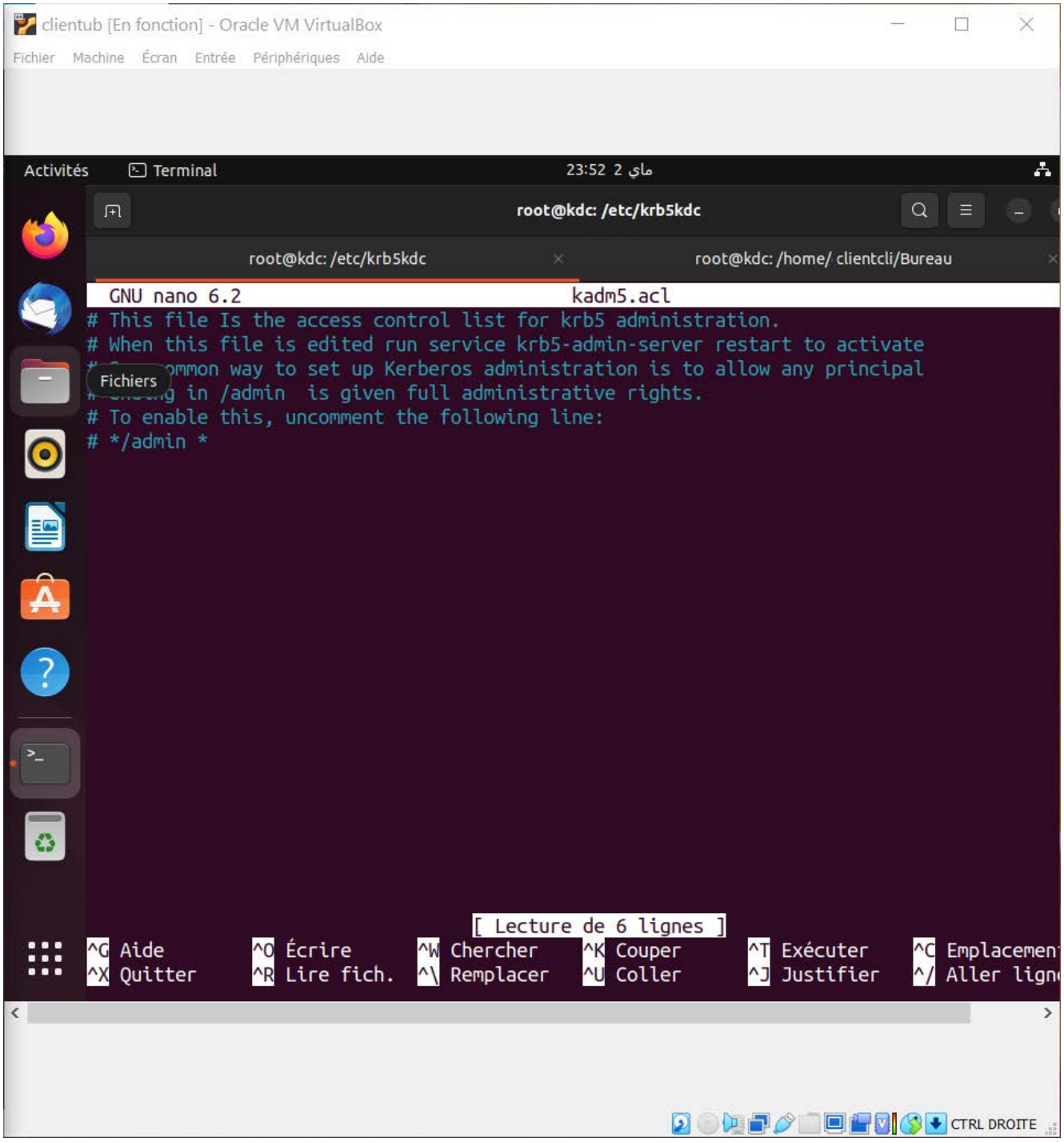
.



Next, we need to grant all access rights to the Kerberos database to admin principal *root/admin* using the configuration file */etc/krb5kdc/kadm5.acl* .
sudo vi /etc/krb5kdc/kadm5.acl

In this file, we need to add the following line:

***/admin@CENTRAL.TN ***



For changes to take effect, we need to restart the following service : sudo service krb5-admin-server restart

Once the admin user who manages principals is created, we need to create the principals. We will to create principals for both the client machine and the service server machine.

Create a principal for the client

$ sudo kadmin.local

kadmin.local: add_principal cli

Create a principal for the service server

kadmin.local: add_principal postgres/pg.central.tn

```
kadmin.local:  add_principal postgres/pg.central.tn
No policy specified for postgres/pg.central.tn@CENTRAL.TN; defaulting to no policy
Enter password for principal "postgres/pg.central.tn@CENTRAL.TN":
Re-enter password for principal "postgres/pg.central.tn@CENTRAL.TN":
Principal "postgres/pg.central.tn@CENTRAL.TN" created.
```

We can check the list of principals by running the command ***:*** **kadmin.local: list_principals**

```
kadmin.local:  list_principals
K/M@CENTRAL.TN
kadmin/admin@CENTRAL.TN
kadmin/changepw@CENTRAL.TN
krbtgt/CENTRAL.TN@CENTRAL.TN
  Cli/kdc.central.tn@CENTRAL.TN
Cli@CENTRAL.TN
Cli@CENTRAL.TN
postgres/pg.central.tn@CENTRAL.TN
root/admin@CENTRAL.TN
kadmin.local:
```

Service server Machine Configuration

For an easier configuration of Postgres I changed the Operating System login name from 'omar' to 'postgres'.

Configuration of Kerberos

Installation of Packages

Following are the packages that need to be installed on the Service server machine :

$ sudo apt-get update

$ sudo apt-get install krb5-user libpam-krb5 libpam-ccreds

During the installation, we will be asked for the configuration of :

- the realm : 'INSAT.TN' (must be *all uppercase*)
- the Kerberos server : 'kdc.insat.tn'
- the administrative server : 'kdc.insat.tn'

PS : *We need to enter the same information used for KDC Server.*

Preparation of the *keytab file*

We need to extract the service principal from KDC principal database to a keytab file.

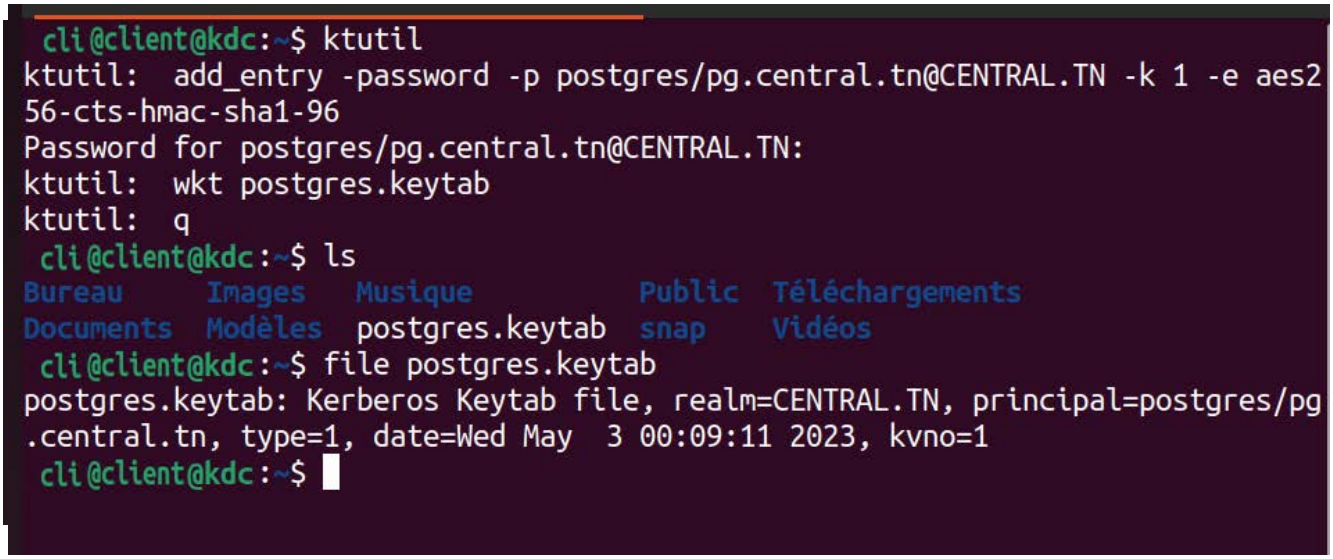
1. In the KDC machine run the following command to generate the keytab file in the current folder :

$ ktutil

ktutil: add_entry -password -p postgres/pg.insat.tn@INSAT.TN -k 1 -e aes256-cts-hmac-sha1-96

Password for postgres/pg.insat.tn@INSAT.TN:

ktutil: wkt postgres.keytab



2. Send the keytab file from the KDC machine to the Service server machine :

In the Postgres server machine make the following directories :

mkdir -p /home/postgres/pgsql/data

In the KDC machine send the keytab file to the Postgres server :

scp postgres.keytab
postgres@<PG_SERVER_IP_ADDRESS>:/home/postgres/pgsql/data

! We need to have **openssh-server** package installed on the service server **: sudo apt-get install openssh-server.**

```
cli@client@kdc:~$ scp postgres.keytab ubuntu@pg.central.tn:/home/ubuntu/Bureau
/postgres/pgsql/data
ubuntu@pg.central.tn's password:
postgres.keytab                               100%   94    66.5KB/s   00:00
cli@client@kdc:~$
```

3. **Verify that the service principal was succesfully extracted from the KDC database :**

   i. **List the current keylist**

ktutil: list

   ii. **Read a krb5 keytab into the current keylist**

ktutil: read_kt pgsql/data/postgres.keytab

   iii. **List the current keylist again**

ktutil: list

```
ktutil:  list
slot KVNO Principal
---- ---- ---------------------------------------------------------------------
   1    1        postgres/pg.central.tn@CENTRAL.TN
ktutil:
```

Configuration of the service (PostgreSQL)

Installation of PostgreSQL

1. Update the package lists

sudo apt-get update

2. Install necessary packages for Postgres

sudo apt-get install postgresql postgresql-contrib

3. Ensure that the service is started

sudo systemctl start postgresql

Create a Postgres Role for the Client

We will need to :

- create a new role for the client

create user cli with encrypted password 'some_password';

- create a new database

create database omar;

- **grant all privileges on this database to the new role**

grant all privileges on database cli to cli;

```
ubuntu@pg:~$ sudo -u postgres psql
could not change directory to "/home/ubuntu": Permission non accordée
psql (14.7 (Ubuntu 14.7-0ubuntu0.22.04.1))
Type "help" for help.

postgres=# create user omar with encrypted password 'cli';
CREATE ROLE
postgres=# create database  cli;
CREATE DATABASE
postgres=# grant all privileges on database cli  to  cli;
GRANT
postgres=#
```

To ensure the role was successfully created run the following command :

postgres=# SELECT usename FROM pg_user WHERE usename LIKE 'cli';

```
postgres=# SELECT usename FROM pg_user WHERE usename LIKE 'cli';
 usename
---------
 cli
(1 row)

postgres=#
```

The client yosra has now a role in Postgres and can access its database 'yosra'.

Update Postgres Configuration files (*postgresql.conf* and *pg_hba.conf*)

- Updating *postgresql.conf*

To edit the file run the following command:

sudo vi /etc/postgresql/14/main/postgresql.conf

By default, Postgres Server only allows connections from localhost. Since the client will connect to the Postgres server remotely, we will need to modify *postgresql.conf* so that Postgres Server allows connection from the network:

listen_addresses = '*'

We will also need to specify the keytab file location:

krb_server_keyfile = '/home/postgres/pgsql/data/postgres.keytab'

```
Messagerie Thunderbird
t /etc/postgresql/14/main/postgresql.conf | grep -e listen_addr
esses -e krb_server_keyfile
#listen_addresses = '*'          # what IP address(es) to listen on;
#krb_server_keyfile = '/home/ubuntu/Bureau/postgres/pgsql/data/postgres.keytab
'
ubuntu@pg:~$
```

- **Updating *pg_hba.conf***

HBA stands for host-based authentication. *pg_hba.conf* is the file used to control clients authentication in PostgreSQL. It is basically a set of records. Each record specifies a **connection type**, a **client IP address range**, a **database name**, a **user name**, and the **authentication method** to be used for connections matching these parameters.

The first field of each record specifies the **type of the connection attempt** made. It can take the following values :

- **local :** Connection attempts using *Unix-domain sockets* will be matched.
- **host :** Connection attempts using *TCP/IP* will be matched (SSL or non-SSL as well as GSSAPI encrypted or non-GSSAPI encrypted connection attempts).
- **hostgssenc :** Connection attempts using *TCP/IP* will be matched, but only when the connection is made with GSSAPI encryption.

This field can take other values that we won't use in this setup. For futher information you can visit the official documentation.

Some of the possible choices for the **authentication method** field are the following :

- **trust :** Allow the connection unconditionally.
- **reject :** Reject the connection unconditionally.
- **md5 :** Perform SCRAM-SHA-256 or MD5 authentication to verify the user's password.
- **gss :** Use GSSAPI to authenticate the user.
- **peer :** Obtain the client's operating system user name from the operating system and check if it matches the requested database user name. This is only available for *local connections*.

So to allow the user 'omar' to connect remotely using Kerberos we will add the following line:

 # IPv4 local connections:

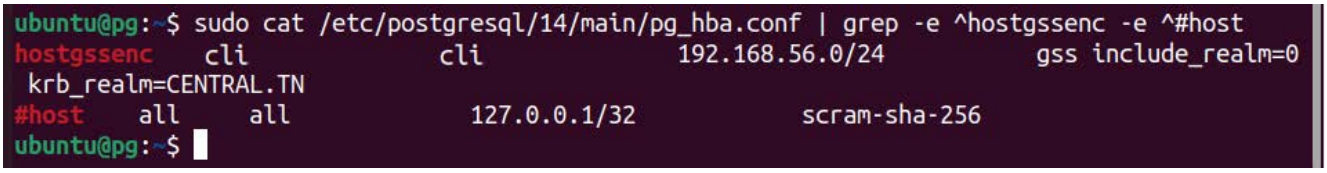
hostgssenc omar omar <IP_ADDRESS_RANGE> gss include_realm=0 krb_realm=INSAT.TN

And comment other connections over TCP/IP.

krb_realm= CENTRAL.TN: Only users of **CENTRAL.TN** realm will be accepted.

include_realm=0: If include_realm is set to 0, the realm name from the authenticated user principal is stripped off before being passed through the user name mapping. In a multi-realm environments this may not be secure unless krb_realm is also used.

For changes to take effect we need to restart the service: **sudo systemctl restart postgresql**.



Client Machine Configuration

Following are the packages that need to be installed on the Client machine :

$ sudo apt-get update

$ sudo apt-get install krb5-user libpam-krb5 libpam-ccreds

During the installation, we will be asked for configuration of :

- the realm : 'CENTRAL.TN' (must be *all uppercase*)
- the Kerberos server : 'kdc.central.tn'
- the administrative server : 'kdc.central.tn'

PS : *We need to enter the same information used for KDC Server.*

User Authentication

Once the setup is complete, it's time for the client to authenticate using Kerberos.

First, try to connect to PostgreSQL remotely:

$ psql -d omar -h postgres.insat.tn -U omar # No pg_hba.con entry found for host xxx

-d specifies the database, *-U* specifies the postgres role and *-h* specifies the ip address of the machine hosting postgres.

In the client machine, check the cached credentials:

$ Klist # No credentails found

Then initial the user authentication:

$ kinit omar

And check the ticket granting ticket (TGT) :

$ klist

Now try to connect once again and check the service ticket !

Acknowledgments

A list of resources, which are helpful and would like to give credit to:

- GSSAPI kerberos
- Kerberos docs
- Postgresql with KRB
- TECHWALL : Useful video tutorial