ECOLE CENTRALE DES LOGICIELS LIBRES ET DE TELECOMMUNICATIONS

INSTITUT SUPÉRIEUR DE TECHNOLOGIE DE l'UNIVERSITÉ DE BANGUI (IST/UB)

SERVICE D'ANNUAIRE ET D'AUTHENTIFICATION

➢ **Présenté par :**
Ismaïla FALL
Benam BERENGER
Germaine Daba DIOUF

Sous la Direction du
Pr Samuel OUYA

Table des matières

Introduction 1
Séquence 1 : Kerberos Principe et fonctionnement 2
I Définition des composants et des termes 4
II Installation de Kerberos 4
III Test client/serveur Kerberos 10
Conclusion 12
Séquence 2 : LDAP ET SSSD 13
I LDAP 13
II SSSD 14
III Scenario 14
IV Installation et configuration 15
VI Test 21
Conclusion 25
Séquence 3 : Kerberos et openLDAP comme backend 26
I Kerberos et LDAP 26
II Installation et configuration 26
Conclusion 33
Séquence 4 : SSSD, LDAP et Kerberos 34
I SSSD Configuration 35
II Test 36
Conclusion 40
Annexes : Quelques commande utiles 41

Objectifs :

- ✓ Décrire les différentes entités du service Kerberos 5
- ✓ Comprendre les termes du service Kerberos 5
- ✓ Savoir utiliser LDAP comme base de données de Kerberos
- ✓ Savoir gérer une infrastructure à clé (Pki) (entités TLS, CA, certificats clients, certificats serveurs, …)
- ✓ Savoir utiliser SSSD pour gestion des identités centralisées
- ✓ Pouvoir mettre en œuvre SSSD et LDAP et le coupler à Kerberos pour l'authentification au lieu de mots de passe simples

Introduction

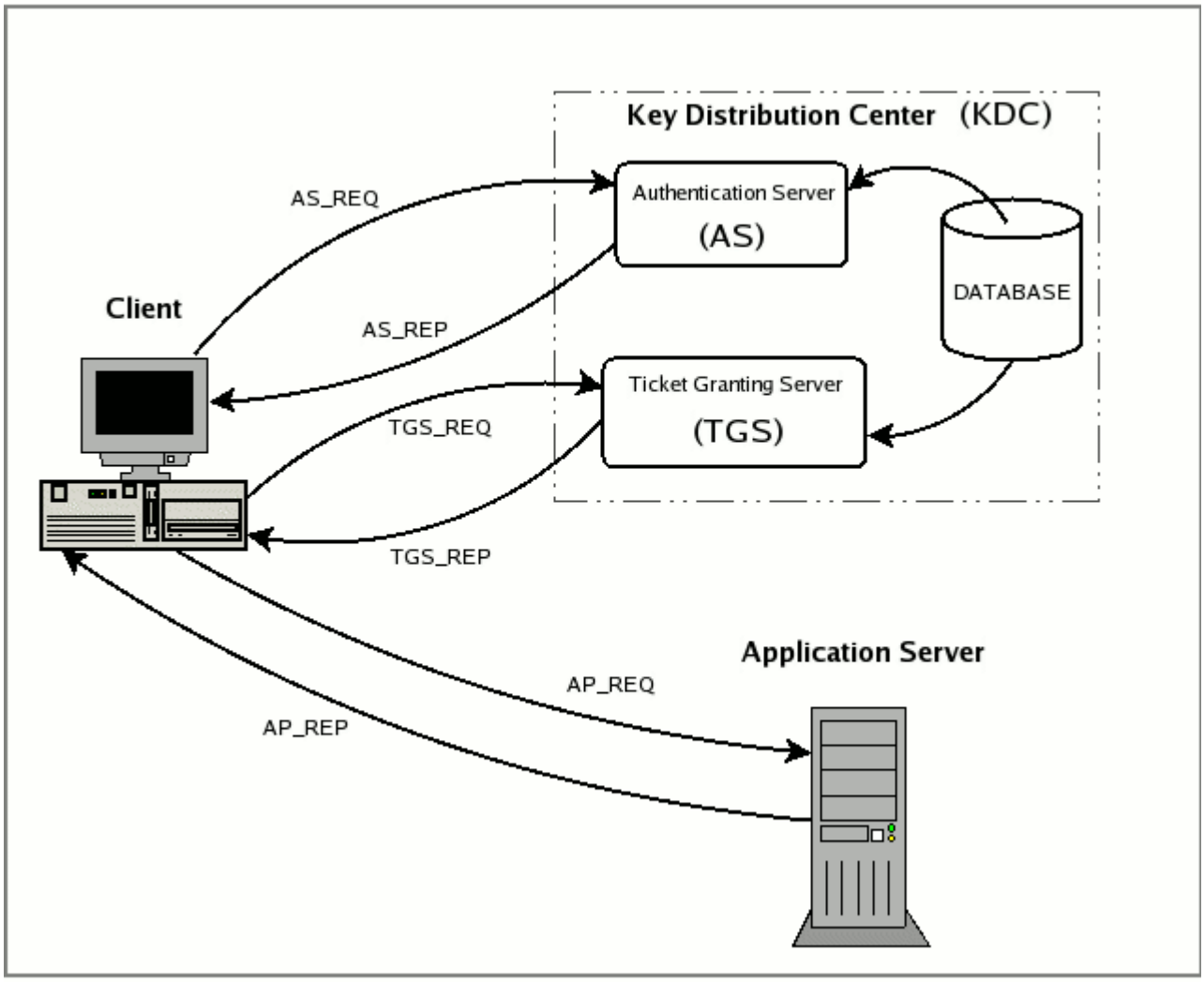
L'authentification est le processus de confirmation d'une identité. Pour les interactions réseau, l'authentification implique l'identification d'une partie par une autre partie. Il existe de nombreuses façons d'utiliser l'authentification sur les réseaux : mots de passe simples, certificats, jetons de mot de passe à usage unique (OTP), scans biométriques.
L'autorisation, d'autre part, définit ce que la partie authentifiée est autorisée à faire ou à accéder.

L'authentification nécessite qu'un utilisateur présente une sorte d'informations d'identification pour vérifier son identité. Le type d'informations d'identification requis est défini par le mécanisme d'authentification utilisé. Il existe plusieurs types d'authentification pour les utilisateurs locaux sur un système :

- Authentification par mot de passe. Presque tous les logiciels permettent à l'utilisateur de s'authentifier en fournissant un nom et un mot de passe reconnus. C'est ce qu'on appelle aussi l'authentification simple.
- Authentification basée sur les certificats. L'authentification client basée sur des certificats fait partie du protocole SSL. Le client signe numériquement un élément de données généré de manière aléatoire et envoie à la fois le certificat et les données signées sur le réseau. Le serveur valide la signature et confirme la validité du certificat.
- Authentification Kerberos. Kerberos établit un système d'informations d'identification de courte durée, appelées tickets d'octroi de billets (TGT). L'utilisateur présente des informations d'identification, c'est-à-dire un nom d'utilisateur et un mot de passe, qui identifient l'utilisateur et indiquent au système que l'utilisateur peut recevoir un ticket. TGT peut ensuite être utilisé à plusieurs reprises pour demander des billets d'accès à d'autres services, tels que des sites Web et des e-mails. L'authentification à l'aide de TGT permet à l'utilisateur de ne subir qu'un seul processus d'authentification de cette manière.

Séquence 1 : Kerberos Principe et fonctionnement

Kerberos est un protocole d'authentification réseau. Il est conçu pour Fournir une authentification forte pour les applications client/serveur à l'aide de Cryptographie à clé secrète.
Son nom «Kerberos» vient de la mythologie grecque et correspond au nom du chien à trois têtes gardien des portes d'Hadès. Il porte relativement bien son nom puisqu'il est chargé d'authentifier, d'autoriser et de surveiller les utilisateurs voulant accéder aux ressources et services de votre réseau. Il agit en chien de garde contre les intrus sur vos services réseau. Le protocole Kerberos a été normalisé dans sa version 5 par l'IETF dans les RFC 1510 (en septembre 1993) et RFC 1964 (juin 1996).
C'est un standard qui résout de nombreux problèmes de sécurité, d'administration, et de productivité dans l'authentification des clients et des services au sein d'un réseau. Le protocole Kerberos est conçu pour fournir une authentification fiable sur des réseaux ouverts et non sécurisés où les communications entre les hôtes lui appartenant peuvent être interceptées.
Cependant, il faut savoir que Kerberos ne fournit aucune garantie si les ordinateurs utilisés sont vulnérables : les serveurs d'authentification, les serveurs d'applications (imap, pop, smtp, telnet, ftp, ssh, AFS, lpr, ...) et les clients doivent être constamment mis à jour afin que l'authenticité des utilisateurs et des fournisseurs de services demandeurs puisse être garantie.



Fonctionnement de de Kerberos

- **AS_REQ** est la demande d'authentification initiale de l'utilisateur (c'est-à-dire effectuée avec kinit) Ce message est dirigé vers le composant KDC appelé serveur d'authentification (AS) ;
- **AS_REP** est la réponse du serveur d'authentification à la demande précédente. Fondamentalement, il contient le TGT (crypté à l'aide de la clé secrète TGS) et la clé de session (cryptée à l'aide de la clé secrète de l'utilisateur demandeur) ;
- **TGS_REQ** s'agit de la demande du client au serveur TGS (Ticket Grant Server) pour un ticket de service. Ce paquet comprend le TGT obtenu à partir du message précédent et un authentificateur généré par le client et chiffré avec la clé de session ;
- **TGS_REP** est la réponse du serveur d'octroi de tickets à la demande précédente. À l'intérieur se trouvent le ticket de service demandé (crypté avec la clé secrète du service) et une clé de session de service générée par TGS et cryptée à l'aide de la clé de session précédente générée par l'AS ;
- **AP_REQ** s'agit de la demande que le client envoie à un serveur d'applications pour accéder à un service. Les composants sont le ticket de service obtenu à partir de TGS avec la

réponse précédente et un authentificateur à nouveau généré par le client, mais cette fois crypté à l'aide de la clé de session de service (générée par TGS) ;

- **AP_REP'est** la réponse que le serveur d'applications donne au client pour prouver qu'il s'agit bien du serveur que le client attend. Ce paquet n'est pas toujours demandé. Le client demande le serveur uniquement lorsque l'authentification mutuelle est nécessaire.

I Définition des composants et des termes

Si vous débutez avec Kerberos, il est bon de comprendre quelques termes avant de configurer un serveur Kerberos. La plupart des termes se rapportent à des choses que vous connaissez peut-être dans d'autres environnements :

- ***Principal :*** tous les utilisateurs, ordinateurs et services fournis par les serveurs doivent être définis comme principaux Kerberos.
- ***Instances :*** sont une variante pour les principaux de service.
- **Realms** : domaine de contrôle unique fourni par l'installation Kerberos. Considérez-le comme le domaine ou le groupe auquel appartiennent vos hôtes et utilisateurs. La convention dicte que le royaume doit être en majuscules. Par défaut, Ubuntu utilisera le domaine DNS converti en majuscules () comme royaume RTN.SN
- ***Le Centre de distribution de clés*** : (KDC) se compose de trois parties : une base de données de tous les mandataires, le serveur d'authentification et le serveur d'octroi de tickets. Pour chaque domaine, il doit y avoir au moins un KDC.
- **Ticket d'octroi de ticket** : émis par le serveur d'authentification (AS), le ticket *d'octroi de ticket* (TGT) est crypté dans le mot de passe de l'utilisateur qui n'est connu que de l'utilisateur et du KDC. C'est le point de départ pour un utilisateur d'acquérir des tickets supplémentaires pour les services auxquels il accède.
- ***Serveur d'octroi* de tickets :** (TGS) émet des tickets de service aux clients sur demande.
- ***Billets :*** confirmez l'identité des deux donneurs d'ordre. Un principal étant un utilisateur et l'autre un service demandé par l'utilisateur. Les tickets établissent une clé de chiffrement utilisée pour une communication sécurisée pendant la session authentifiée.
- ***Fichiers keytab*** *:* contiennent des clés de chiffrement pour un service ou un hôte extrait de la base de données principale KDC.

Généralement, Kerberos est utilisé dans l'authentification POSIX, ainsi que dans Active Directory, NFS et Samba.

II Installation de Kerberos

Dans cette séquence, nous allons créer un domaine Kerberos avec les fonctionnalités suivantes :

- *Royaume :* **RTN.SN** (de préférence en MAJUSCULE)
- *KDC principal :* **server.rnt.sn**
- *Principal de l'utilisateur :* **berenger**
- *Principal de l'administrateur :* **ismaila/admin**

#apt update

```
root@server:/home/berenger# apt update
Atteint :1 http://ppa.launchpad.net/open5gs/latest/ubuntu focal InRelease
Atteint :2 https://deb.nodesource.com/node_14.x focal InRelease
```

- ***Configurer la résolution du nom d'hôte***

Tout d'abord, on doit configurer un nom d'hôte complet sur le serveur et sur la machine cliente.
Sur le serveur, définissons le nom d'hôte complet avec la commande suivante :

#hostnamectl set-hostname server.rtn.sn

```
root@server:/home/berenger# hostnamectl set-hostname server.rtn.sn
root@server:/home/berenger#
```

Sur la machine cliente, définissez le nom d'hôte complet avec la commande suivante :
#hostnamectl set-hostname client1.rtn.sn

```
root@berenger:~# hostnamectl set-hostname client1.rtn.sn
root@berenger:~#
```

Ensuite, on modifie les fichiers **/etc/hosts** sur les machines (serveur et client1) pour que les entités puissent communiquer à travers des noms de domaines
On ajoute les lignes suivantes sur les deux machines :

```
  GNU nano 4.8                                    /etc/hosts
127.0.0.1       localhost
127.0.1.1       labo-HP-Z620
192.168.2.80    server.rtn.sn
192.168.2.77    client1.rtn.sn
```

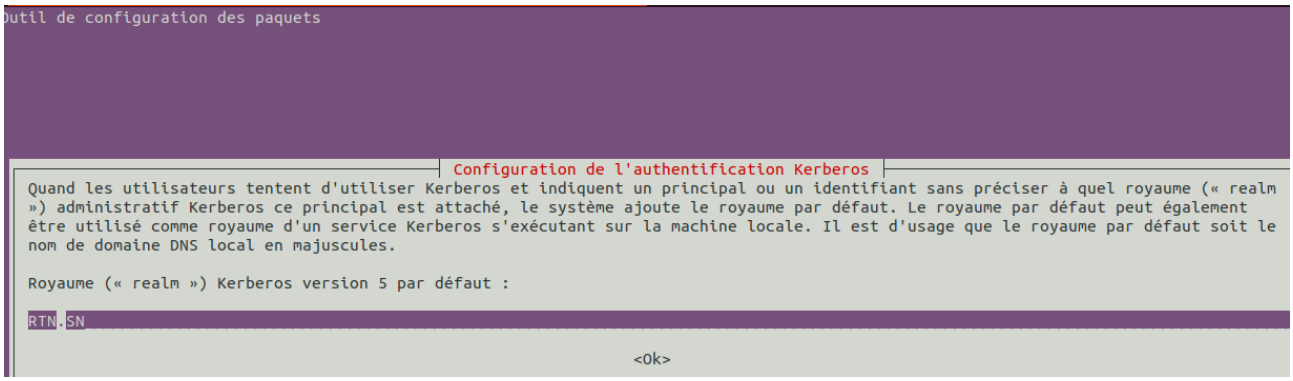
- ***Installer le serveur Kerberos***

Ensuite, on installe les paquets du serveur Kerberos sur la machine serveur.

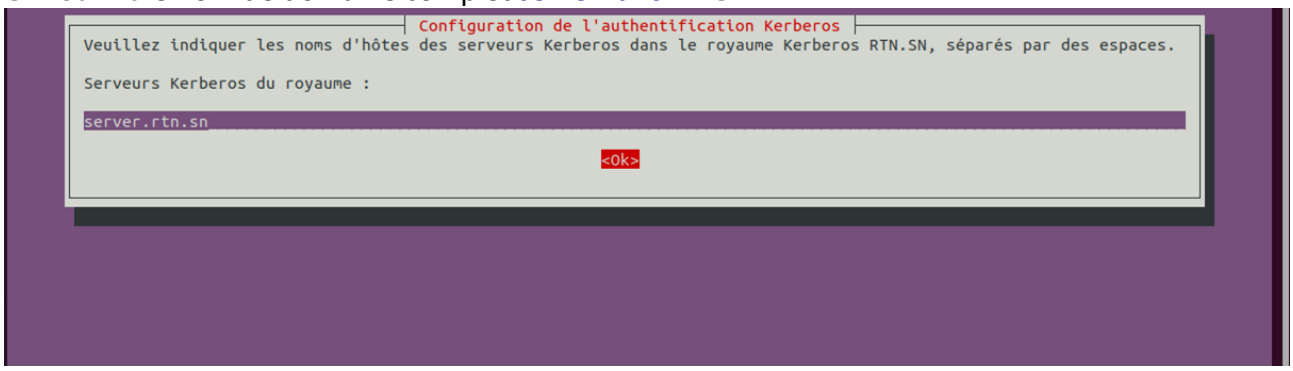
#apt-get install krb5-kdc krb5-admin-server krb5-config -y

```
root@labo-HP-Z620:/home/berenger# apt-get install krb5-kdc krb5-admin-server krb5-config
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  gir1.2-goa-1.0
Veuillez utiliser « apt autoremove » pour le supprimer.
Les paquets supplémentaires suivants seront installés :
  krb5-user libverto-libevent1 libverto1
```

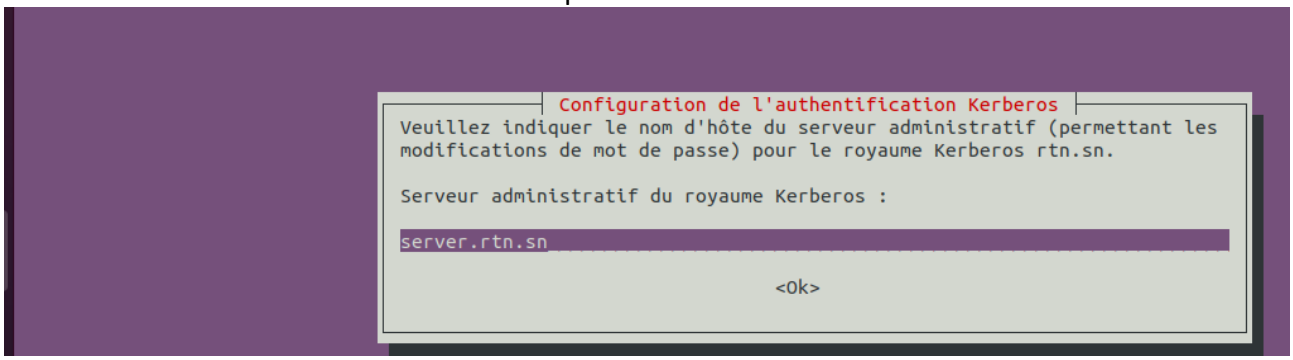
Lors de l'installation, il nous sera demandé de fournir Kerberos **Realm**, comme indiqué ci-dessous :
On renseigne **RTN.SN** et on clique sur le bouton **OK.**



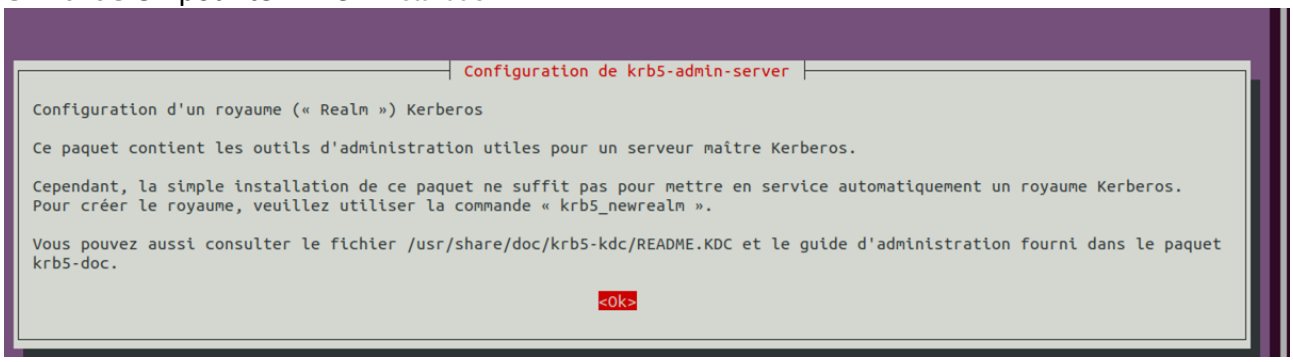
On fournit le nom de domaine complet server.rtn.sn + OK .



Ici aussi on fournit le nom de domaine complet server.rtn.sn + OK .

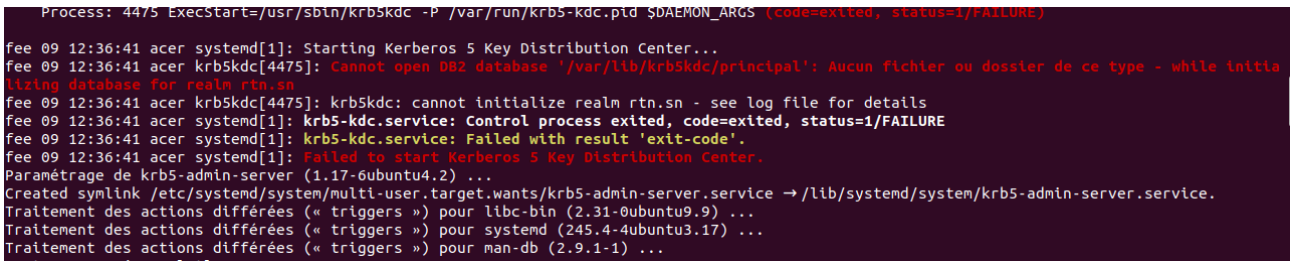


On valide OK pour terminer l'installation



- ***Configurer le serveur Kerberos***

Apres l'installation le serveur ne démarre parce qu'il nous faut d'abord créer un royaume (realm) :



Pour créer un realm on utilise la commande suivante : **krb5_newrealm**

Il nous sera demandé de fournir un mot de passe comme indiqué ci-dessous :

```
root@labo-HP-Z620:/home/berenger# 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 'RTN.SN',
master key name 'K/M@RTN.SN'
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.
```

Ensuite, vous devrez ajouter le principal admin à la base de données Kerberos. Vous pouvez le faire avec la commande suivante : **kadmin.local**

Vous devriez voir la sortie suivante :

#addprinc root/admin « ajout d'un principal user admin »

#addprinc -randkey host/server.rtn.sn « ajout d'un principal service »

#ktadd host/server.rtn.sn « extraire la clé du KDC et la stocker dans le serveur »

```
root@labo-HP-Z620:/home/berenger# kadmin.local
Authenticating as principal root/admin@RTN.SN with password.
kadmin.local:  addprinc root/admin
WARNING: no policy specified for root/admin@RTN.SN; defaulting to no policy
Enter password for principal "root/admin@RTN.SN":
Re-enter password for principal "root/admin@RTN.SN":
Principal "root/admin@RTN.SN" created.
kadmin.local:  passer
kadmin.local: Unknown request "passer".  Type "?" for a request list.
kadmin.local:  addprinc -randkey host/server.rtn.sn
WARNING: no policy specified for host/server.rtn.sn@RTN.SN; defaulting to no policy
Principal "host/server.rtn.sn@RTN.SN" created.
kadmin.local:  ktadd host/server.rtn.sn
Entry for principal host/server.rtn.sn with kvno 2, encryption type aes256-cts-hmac-sha1-96 added to keytab FILE:/etc/k
rb5.keytab.
Entry for principal host/server.rtn.sn with kvno 2, encryption type aes128-cts-hmac-sha1-96 added to keytab FILE:/etc/k
rb5.keytab.
```

On ajoute le principe de l'utilisateur administrateur au contrôle d'accès.

nano /etc/krb5kdc/kadm5.acl

Ajout de la ligne suivante à la fin :

root/admin *

```
  GNU nano 4.8                          /etc/krb5kdc/kadm5.acl
# 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 principal
# ending in /admin  is given full administrative rights.
# To enable this, uncomment the following line:
root/admin *
```

On redémarre le serveur par la commande : **systemctl restart krb5-admin-server**

```
root@labo-HP-Z620:/home/berenger# systemctl restart krb5-admin-server
```

On vérifie le statut de krb : **systemctl status krb5-admin-server**

```
root@labo-HP-Z620:/home/berenger# systemctl status krb5-admin-server
● krb5-admin-server.service - Kerberos 5 Admin Server
     Loaded: loaded (/lib/systemd/system/krb5-admin-server.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2023-02-10 10:59:33 GMT; 26s ago
   Main PID: 13176 (kadmind)
      Tasks: 1 (limit: 77145)
     Memory: 820.0K
     CGroup: /system.slice/krb5-admin-server.service
             └─13176 /usr/sbin/kadmind -nofork

fee 10 10:59:33 server.rtn.sn kadmind[13176]: Setting up TCP socket for address 0.0.0.0.464
fee 10 10:59:33 server.rtn.sn kadmind[13176]: Setting up TCP socket for address ::.464
fee 10 10:59:33 server.rtn.sn kadmind[13176]: setsockopt(12,IPV6_V6ONLY,1) worked
fee 10 10:59:33 server.rtn.sn kadmind[13176]: Setting up RPC socket for address 0.0.0.0.749
fee 10 10:59:33 server.rtn.sn kadmind[13176]: Setting up RPC socket for address ::.749
fee 10 10:59:33 server.rtn.sn kadmind[13176]: setsockopt(14,IPV6_V6ONLY,1) worked
fee 10 10:59:33 server.rtn.sn kadmind[13176]: set up 6 sockets
fee 10 10:59:33 server.rtn.sn kadmind[13176]: Seeding random number generator
fee 10 10:59:33 server.rtn.sn kadmind[13176]: kadmind: starting...
fee 10 10:59:33 server.rtn.sn kadmind[13176]: starting
```

- ***Configuration du client Kerberos***
- ***Configuration de FQDN du client***

#hostnamectl set-hostname client1.rtn.sn

```
root@berenger:~# hostnamectl set-hostname client1.rtn.sn
root@berenger:~# nano /etc/hosts
root@berenger:~#
```

Editons le fichier ***/etc/*hosts** et renseigner l'adresse IP du serveur et la machine cliente elle-même.

```
  GNU nano 2.9.3                          /etc/hosts                          Modifié

127.0.0.1       localhost
192.168.2.80    server.rtn.sn
192.168.2.77    client1.rtn.sn

# 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
```

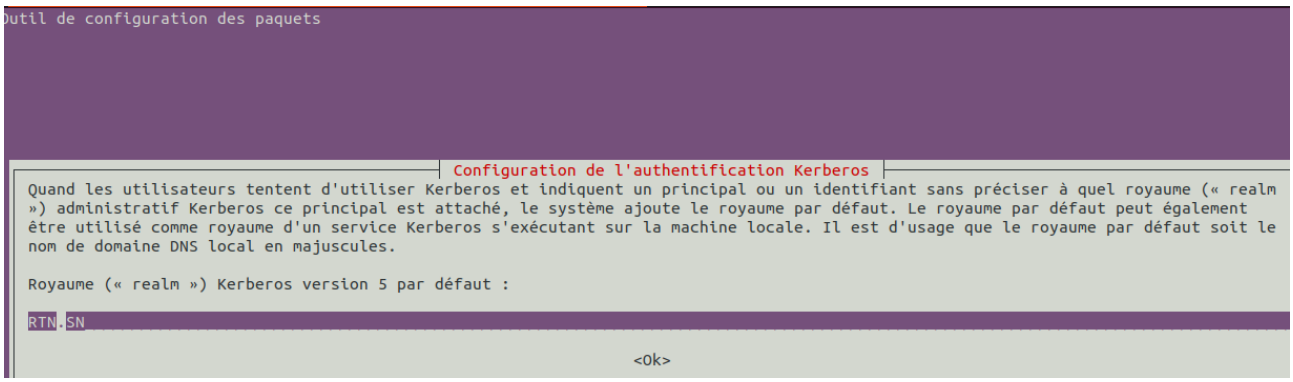
- ***Installation du client kerberos***

Les paquets à installer :

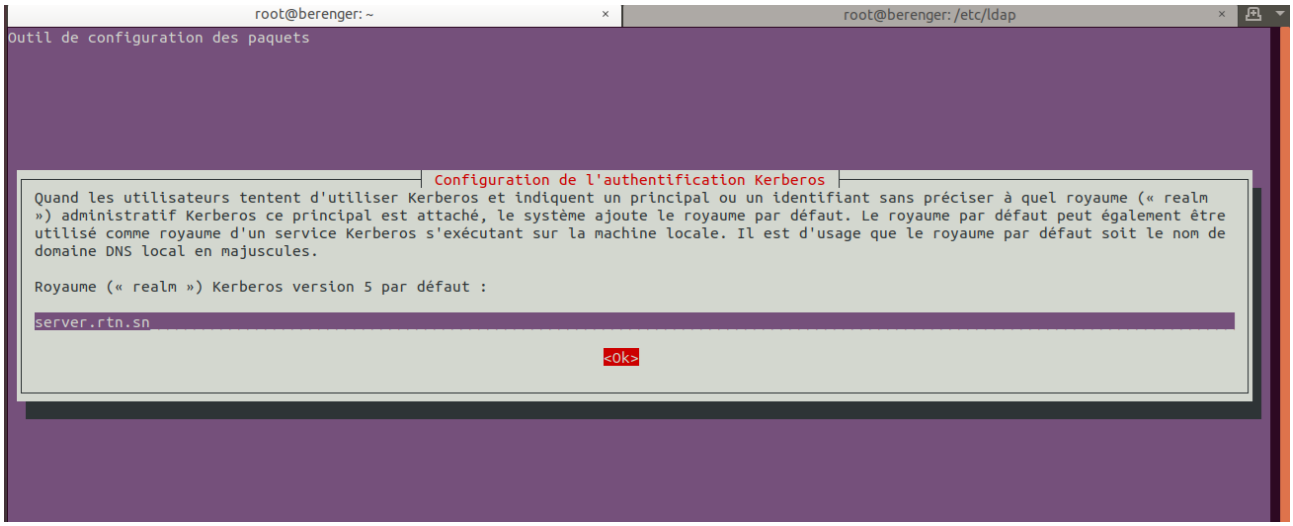
#apt install krb5-user libpam-krb5 libpam-ccreds auth-client-config

```
root@berenger:~# apt install krb5-user libpam-krb5 libpam-ccreds auth-client-config
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  krb5-config libgssrpc4 libkadm5clnt-mit11 libkadm5srv-mit11 libkdb5-9
Paquets suggérés :
  libpam-cracklib krb5-doc nss-updatedb
```

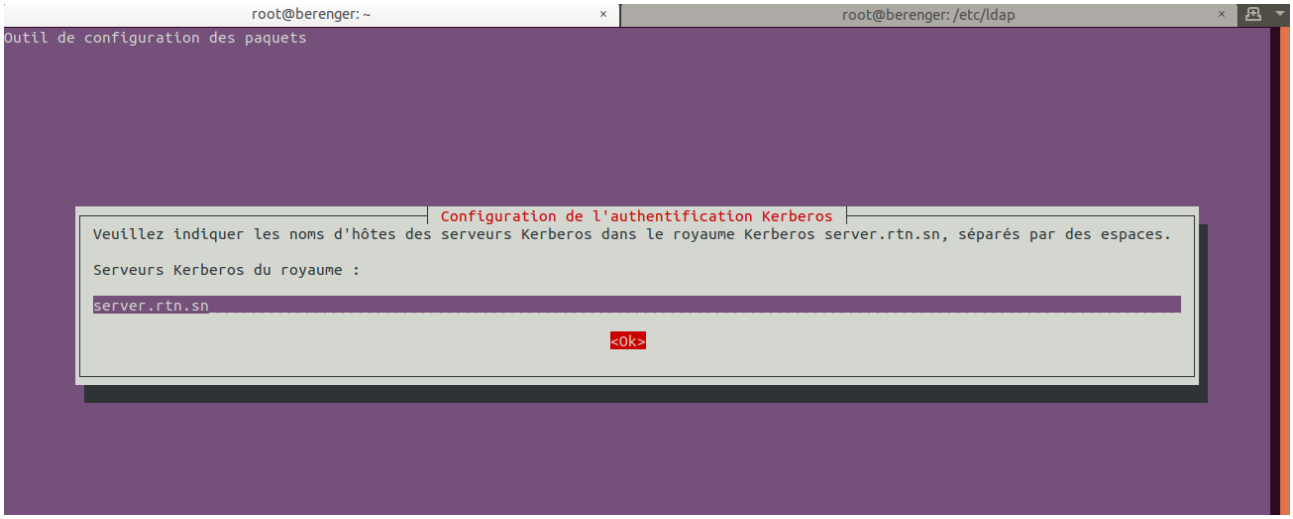
On renseigne le realm (RTN.SN) et le nom de domaine du serveur.



On fournit le nom de domaine du serveur



On renseigne encore le nom de domaine définit sur la machine serveur



À partir de la machine cliente on test la connectivité par la commande : **kadmin** pour ajouter le principal service de la machine cliente

#addprinc -randkey host/client1.rtn.sn

```
root@berenger:~# kadmin
Authenticating as principal root/admin@RTN.SN with password.
Password for root/admin@RTN.SN:
kadmin:  addprinc -randkey host/client1.rtn.sn
WARNING: no policy specified for host/client1.rtn.sn@RTN.SN; defaulting to no policy
Principal "host/client1.rtn.sn@RTN.SN" created.
kadmin:  ktadd host/client1.rtn.sn
Entry for principal host/client1.rtn.sn with kvno 2, encryption type aes256-cts-hmac-sha1-96 added to keyta
b FILE:/etc/krb5.keytab.
Entry for principal host/client1.rtn.sn with kvno 2, encryption type aes128-cts-hmac-sha1-96 added to keyta
b FILE:/etc/krb5.keytab.
kadmin:  quit
```

III Test client/serveur Kerberos

- **Configurer le serveur 'server.rtn.sn' en ajoutant un compte**

L'ajout du compte **mbaye** sur le serveur par la commande **:**

#useradd -m -s /bin/bash mbaye

```
root@labo-HP-Z620:/home/berenger# useradd -m -s /bin/bash mbaye
```

On teste par la commande : **kadmin.local**

#addprinc mbaye

```
root@labo-HP-Z620:/home/berenger# kadmin.local
Authenticating as principal root/admin@RTN.SN with password.
kadmin.local:  addprinc mbaye
WARNING: no policy specified for mbaye@RTN.SN; defaulting to no policy
Enter password for principal "mbaye@RTN.SN":
Re-enter password for principal "mbaye@RTN.SN":
add_principal: Empty passwords are not allowed while creating "mbaye@RTN.SN".
kadmin.local:
kadmin.local:
kadmin.local:
kadmin.local:  addprinc mbaye
WARNING: no policy specified for mbaye@RTN.SN; defaulting to no policy
Enter password for principal "mbaye@RTN.SN":
Re-enter password for principal "mbaye@RTN.SN":
Principal "mbaye@RTN.SN" created.
kadmin.local:
kadmin.local:  quit
```

On édite le fichier /etc/ssh/sshd_config pour décommenter les deux lignes suivantes :

#GSSAPIAuthentication yes

#GSSAPICleanupCredentials yes

```
  GNU nano 4.8                          /etc/ssh/sshd_config

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no

# GSSAPI options
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
```

Puis on redémarre sshd : **systemctl restart sshd**

```
root@server:/home/berenger# systemctl restart sshd
root@server:/home/berenger#
```

- **Configurer la machine 'client1.rtn.sn'**

Ajout de l'utilisateur **mbaye** sur la machine cliente

#useradd -m -s /bin/bash edgard

```
root@berenger:~# useradd -m -s /bin/bash mbaye
```

On vient d'ajouter le compte **mbaye** sur la machine cliente et ensuite on se connecte au compte **mbaye** par la commande **su - mbaye**

#su – mbaye

```
root@berenger:~# su - mbaye
```

On fait **kinit mbaye**

```
mbaye@client1:~$ kinit mbaye
Password for mbaye@RTN.SN:
mbaye@client1:~$ klist
Ticket cache: FILE:/tmp/krb5cc_1004
Default principal: mbaye@RTN.SN

Valid starting       Expires              Service principal
10/02/2023 13:38:07  10/02/2023 23:38:07  krbtgt/RTN.SN@RTN.SN
        renew until 11/02/2023 13:37:54
```

Sur cette capture le serveur **kerberos** délivre un ticket d'authentification y compris la durée de validité de celui-ci.

Test de connexion client vers le serveur par ssh

```
mbaye@client1:~$ ssh server.rtn.sn
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-43-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

625 mises à jour peuvent être installées immédiatement.
426 de ces mises à jour sont des mises à jour de sécurité.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Fri Feb 10 12:55:33 2023 from 192.168.2.77
mbaye@server:~$ pwd
/home/mbaye
mbaye@server:~$
```

On remarque que l'utilisateur arrive à se connecter au serveur kerberos sans mot de passe la grâce au ticket obtenu avec **kinit**.

```
root@server:/home/berenger# who
labo     :1           2023-02-10 10:18 (:1)
mbaye    pts/2        2023-02-10 16:31 (192.168.2.77)
root@server:/home/berenger# w
 16:36:45 up  5:20,  2 users,  load average: 0,42, 0,52, 0,54
UTIL.    TTY      DE               LOGIN@   IDLE   JCPU   PCPU QUOI
labo     :1       :1               10:18    ?xdm?   2:46m  0.02s /usr/lib/gdm3/gdm-x-session --run-
mbaye    pts/2    192.168.2.77     16:31   21.00s  0.06s  0.06s -bash
root@server:/home/berenger#
```

Sur la machine serveur **kerberos** on peut voir l'utilisateur **mbaye** s'est connecté via l'adresse IP **192.168.2.77.**

Conclusion

Nous venons d'installer, configurer et tester correctement le serveur et le client Kerberos sur Ubuntu 20.04. Nous pouvons désormais utiliser Kerberos sur votre réseau pour l'authentification des utilisateurs LDAP et LDAP+SSSD.

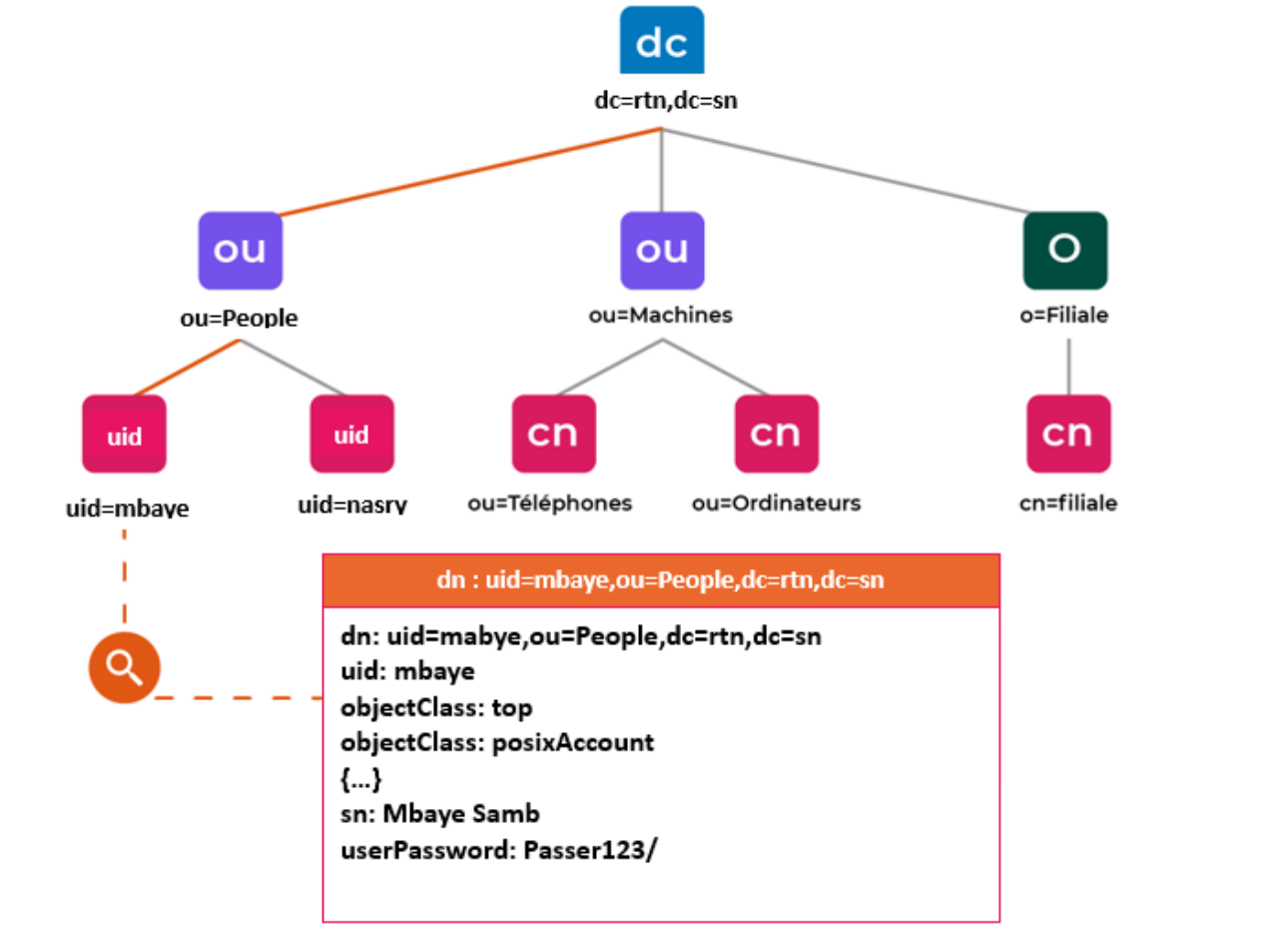
Séquence 2 : LDAP ET SSSD

I LDAP

Le protocole LDAP (Lightweight Directory Access Protocol) est un protocole permettant d'interroger et de modifier un service d'annuaire X.500 exécuté sur TCP/IP. La version actuelle de LDAP est LDAPv3, telle que définie dans RFC4510, et l'implémentation utilisée dans Ubuntu est OpenLDAP.

Le protocole LDAP accède aux annuaires. Une erreur courante est d'appeler un annuaire un *annuaire LDAP*, ou *base de données LDAP*, mais c'est vraiment si courant, et nous savons tous de quoi nous parlons, que c'est ok. Voici quelques concepts et termes clés :

- Un répertoire est une arborescence *d'entrées* de données de nature hiérarchique appelée DIT (Directory Information Tree).
- Une entrée se compose d'un ensemble *d'attributs*.
- Un attribut possède une *clé* (un nom/une description) et une ou plusieurs *valeurs*.
- Chaque attribut doit être défini dans au moins un *objectClass*.
- Les attributs et les classes d'objets sont définis dans *des schémas* (une classe d'objet est en fait considérée comme un type spécial d'attribut).
- Chaque entrée a un identifiant unique : son *nom distinctif* (DN ou dn). Ceci, à son tour, consiste en un *nom distinctif relatif* (RDN) suivi du DN de l'entrée parente.
- Le DN de l'entrée n'est pas un attribut. Il n'est pas considéré comme faisant partie de l'entrée elle-même.



Arborescence d'informations du répertoire, ou DIT

II SSSD

Le démon SSSD (System Security Services Daemon) est un service système qui fournit un ensemble de démons (plugins) permettant de gérer l'accès aux services d'annuaire distants et aux mécanismes d'authentification. Il connecte un système local (un client SSSD) à un système back-end externe (un fournisseur). Cela permet au client SSSD d'accéder aux services distants d'identité et d'authentification à l'aide d'un fournisseur SSSD. Par exemple, ces services distants incluent : un annuaire LDAP, un domaine Identity Management (IdM) ou Active Directory (AD) ou un domaine Kerberos.

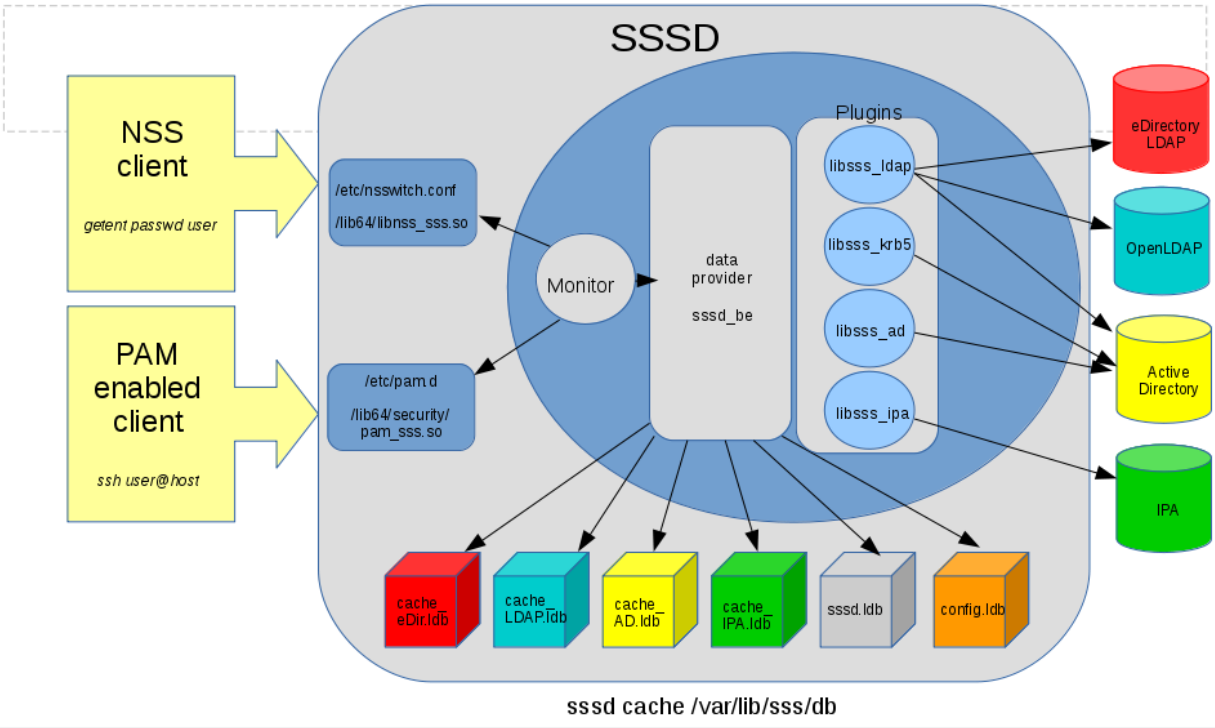
À cette fin, SSSD :

Connecte le client à un magasin d'identités pour récupérer les informations d'authentification.

Utilise les informations d'authentification obtenues pour créer un cache local d'utilisateurs et d'informations d'identification sur le client.

Les utilisateurs du système local peuvent ensuite s'authentifier à l'aide des comptes d'utilisateurs stockés dans le système principal externe.

SSSD ne crée pas de comptes d'utilisateurs sur le système local. Au lieu de cela, il utilise les identités du magasin de données externe et permet aux utilisateurs d'accéder au système local.



sssd cache /var/lib/sss/db

III Scenario

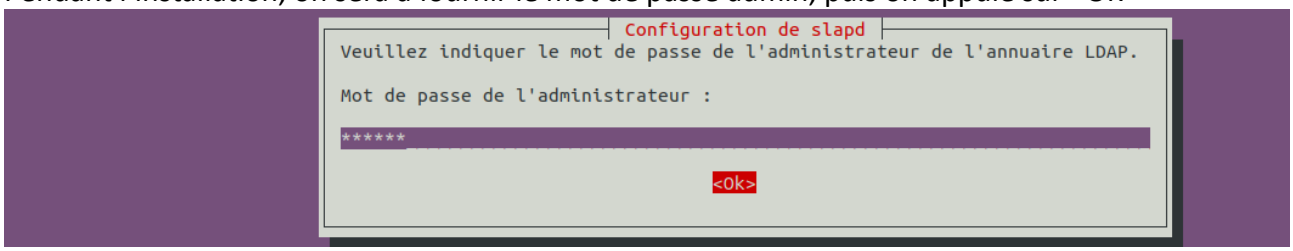
Pourquoi SSSD ?

Il n'est pas nécessaire lorsque qu'on a une poignée de machines à gérer, mais lorsqu'on a des dizaines, des centaines ou des milliers de machines, SSSD devient important un composant directory-client. On ne devrait certainement pas forcer quelqu'un à aller créer un utilisateur, mettre à jour son mot de passe, supprimer un utilisateur sur chaque machine. C'est une perte de temps quand il y a une façon beaucoup plus facile de faire les choses.

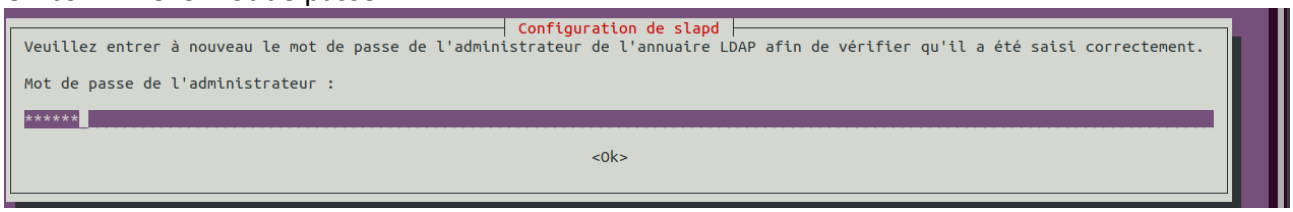
SSSD peut utiliser LDAP pour l'authentification, l'autorisation et les informations sur les utilisateurs/groupes. Dans cette sequence, nous allons configurer un hôte pour authentifier les utilisateurs à partir d'un annuaire OpenLDAP.
Une machine Ubuntu20(client) authentifie les utilisateurs stockés dans une base de données de comptes utilisateur LDAP d'une autre machine (serveur).
La machine cliente utilise le service SSSD (System Security Services Daemon) pour récupérer les données utilisateur.
La machine cliente doit communiquer avec le serveur LDAP via une connexion chiffrée TLS.

IV Installation et configuration

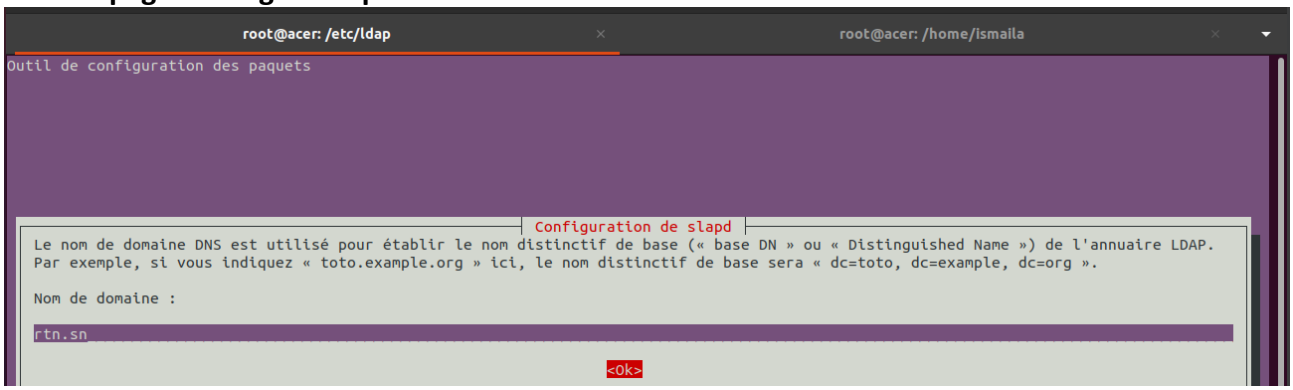
Il faut noter qu'on installera LDAP sur le server et sssd sur le client.
Les paquets à installer :
#apt-get install slapd ldap-utils sssd-ldap
Pendant l'installation, on sera à fournir le mot de passe admin, puis on appuie sur <OK>



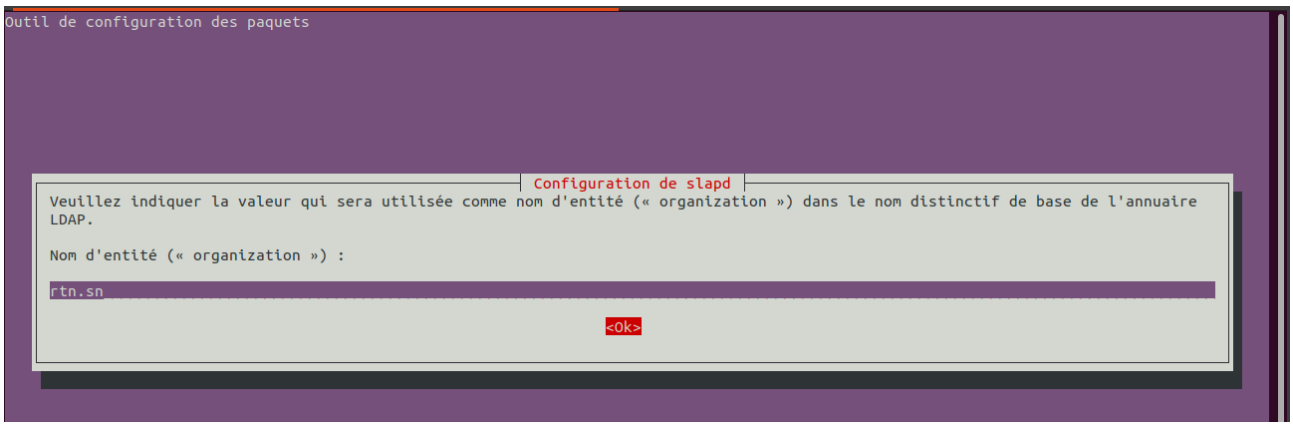
On confirme le mot de passe :



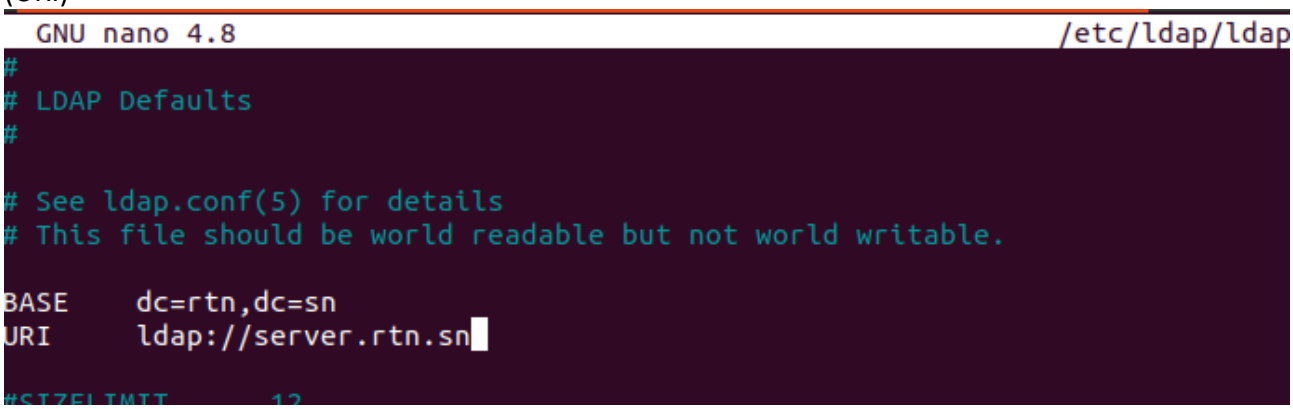
On va modifier votre suffixe DIT, ce serait le bon moment, car le changer supprime votre suffixe existant. Pour modifier le suffixe, on exécute la commande suivante :
#sudo dpkg-reconfigure slapd



On fournit le domaine :



Une fois l'installation terminée, on édite le fichier « **/etc/ldap/ldap.conf** ».
On renseigne les informations de la base tout en indiquant les composants du nom de domaine (URI)



L'empaquetage de slapd est conçu pour être configuré dans le service lui-même en dédiant un DIT séparé à cet effet. Cela permet de configurer dynamiquement slapd sans avoir besoin de redémarrer le service ou de modifier les fichiers de configuration. Cette base de données de configuration se compose d'une collection de fichiers LDIF textuels, mais ceux-ci ne doivent jamais être modifiés directement. Cette façon de travailler est connue sous plusieurs noms : la méthode slapd-config, la méthode RTC (Real Time Configuration), ou la méthode cn=config. On peut toujours utiliser la méthode traditionnelle du fichier (slapd.conf).

Juste après l'installation, on obtient deux bases de données, ou suffixes : une pour nos données, basé sur le domaine (dc=rtn,dc=sn), et une autre pour notre configuration, avec sa racine à cn=config. Pour modifier les données sur chacun, nous avons besoin de différentes informations d'identification et méthodes d'accès :

- **dc=rtn,dc=sn :** l'utilisateur administratif pour ce suffixe est cn=admin,dc=rtn,dc=sn et son mot de passe est celui sélectionné lors de l'installation du package slapd
- **cn=config** : la configuration elle-même est stockée sous ce suffixe. Les modifications peuvent être apportées par le DN spécial gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth. C'est ainsi que l'utilisateur root du système local (uid=0/gid=0) est vu par l'annuaire lors de l'utilisation de l'authentification SASL EXTERNAL via le transport ldapi:/// via le socket unix /run/slapd/ldapi. Essentiellement, cela signifie que seul l'utilisateur root local peut mettre à jour la base de données cn=config.

- **Activation de TLS (serveur Ldap)**

Lors de l'authentification sur un serveur OpenLDAP, il est préférable de le faire en utilisant une session cryptée. Cela peut être accompli à l'aide de TLS (Transport Layer Security).
Avant d'aller plus loin on active TLS pour sur le serveur (OpenLDAP) pour les utilisateurs.
Ici, nous serons notre propre autorité de certification, puis nous créerons et signerons notre certificat de serveur LDAP en tant qu'autorité de certification avec l'outil certool.
On installe les paquets gnutls-bin et ssl-cert :
#sudo apt install gnutls-bin ssl-cert

On crée une clé privée pour l'autorité de certification :

```
sudo certtool --generate-privkey --bits 4096 --outfile /etc/ssl/private/mycakey.pem
```

Créons le modèle/fichier pour définir l'autorité de certification :/etc/ssl/ca.info

On crée le modèle fichier pour définir l'autorité de certification :/etc/ssl/ca.info

```
 GNU nano 4.8                                    /etc/ssl/ca.info
cn = rtn Company
ca
cert_signing_key
expiration_days = 3650
```

On crée le certificat d'autorité de certification auto-signé :

```
sudo certtool --generate-self-signed \
--load-privkey /etc/ssl/private/mycakey.pem \
--template /etc/ssl/ca.info \
--outfile /usr/local/share/ca-certificates/mycacert.crt
```

On éxecute cette commande pour ajouter le nouveau certificat d'autorité de certification à la liste des autorités de certification approuvées :
#update-ca-certificates

```
Signing certificate...
root@acer:/home/ismaila# update-ca-certificates
Updating certificates in /etc/ssl/certs...
rehash: warning: skipping ca-certificates.crt,it does not contain exactly one certificate or CRL
1 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
root@acer:/home/ismaila#
```

Cela crée également un lien symbolique du certificat dans le dossier **/etc/ssl/certs** pointant vers le fichier réel **/usr/local/share/ca-certificates/mycacert.crt**

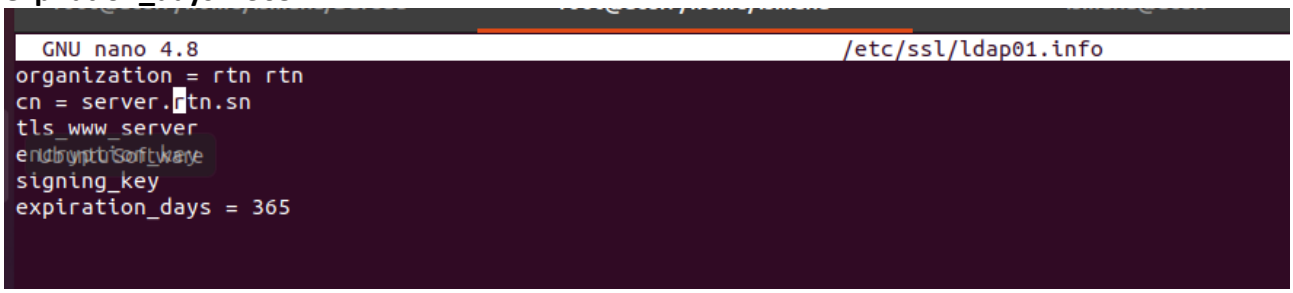
```
root@acer:/home/ismaila# ls -l /etc/ssl/certs/mycacert.pem
lrwxrwxrwx 1 root root 45 fee 14 01:56 /etc/ssl/certs/mycacert.pem -> /usr/local/share/ca-certificates/mycacert.crt
root@acer:/home/ismaila#
```

Créons une clé privée pour le serveur :

```
sudo certtool --generate-privkey \
--bits 2048 \
--outfile /etc/ldap/ldap01_slapd_key.pem
```

*Créons le fichier info **/etc/ssl/ldap01.info** contenant :*

organization = rtn rtn
cn = ldap01.example.com
tls_www_server
encryption_key
signing_key
expiration_days = 365



Le certificat ci-dessus est valable 1 an et n'est valable que pour le nom d'hôte **server.rtn.sn**.

Créons le certificat du serveur :

```
sudo certtool --generate-certificate \
--load-privkey /etc/ldap/ldap01_slapd_key.pem \
--load-ca-certificate /etc/ssl/certs/mycacert.pem \
--load-ca-privkey /etc/ssl/private/mycakey.pem \
--template /etc/ssl/ldap01.info \
--outfile /etc/ldap/ldap01_slapd_cert.pem
```

Ajustons les autorisations et la propriété :

```
sudo chgrp openldap /etc/ldap/ldap01_slapd_key.pem
sudo chmod 0640 /etc/ldap/ldap01_slapd_key.pem
```

Le serveur est maintenant prêt à accepter la nouvelle configuration TLS.
Créons le fichier avec le contenu suivant : ***certinfo.ldif***

```
dn: cn=config
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certs/mycacert.pem
-
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/ldap01_slapd_cert.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/ldap01_slapd_key.pem
```

```
 GNU nano 4.8                                                certinfo.ldif
dn: cn=config
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certs/mycacert.pem
-
add: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/ldap01_slapd_cert.pem
-
add: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/ldap01_slapd_key.pem
```

On utilise la commande **ldapmodify** pour informer slapd de notre travail TLS via la base de données slapd-config :

sudo ldapmodify -Y EXTERNAL -H ldapi:/// -f certinfo.ldif

```
root@acer:/home/ismaila# ldapmodify -Y EXTERNAL -H ldapi:/// -f certinfo.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"

root@acer:/home/ismaila#
```

Vérifications

```
root@acer:/home/ismaila# slapcat -b "cn=config" | grep -E "olcTLS"
olcTLSCACertificateFile: /etc/ssl/certs/mycacert.pem
olcTLSCertificateFile: /etc/ldap/ldap01_slapd_cert.pem
olcTLSCertificateKeyFile: /etc/ldap/ldap01_slapd_key.pem
root@acer:/home/ismaila#
```

Notons qu'on n'a pas besoin d'un redémarrage de slapd pour la prise en charge de StartTLS.

- **Activation de TLS (client Ldap)**

On installe les packages suivants :

#sudo apt install sssd-ldap ldap-utils

```
ismaila@VirtualBox:~$ sudo apt install ldap-utils sssd-ldap
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
```

Vérifier la configuration SSL sur le client
Le client doit pouvoir utiliser START_TLS lors de la connexion au serveur LDAP, avec vérification complète des certificats. Cela signifie :
Le client doit connaître et approuver l'autorité de certification qui a signé le certificat de serveur LDAP.
Le certificat de serveur (ldap01_slapd_cert.pem) a été émis pour l'hôte correct server.rtn.sn

Pour cela on a copié le certificat du serveur LDAP (non pas de l'autorité) qu'on a renommé **ldap.crt** dans le dossier **/usr/local/share/ca-certificates** de la machine cliente.
Ensuite on tape la commande **update-ca-certificates** qui récupérera les autorités de certification locales de confiance.
Cela crée également un lien symbolique du certificat dans le dossier **/etc/ssl/certs** pointant vers le fichier réel **/usr/local/share/ca-certificates/ldap.crt**

```
root@VirtualBox:~# ls -l /etc/ssl/certs/ldap.pem
lrwxrwxrwx 1 root root 41 fee 14 04:03 /etc/ssl/certs/ldap.pem -> /usr/local/share/ca-certificates/ldap.crt
root@VirtualBox:~#
```

On indique au client le chemin du certificat dans le fichier **/etc/ldap/ldap.conf**

```
GNU nano 4.8                                                    /etc/ldap/ldap.conf
#
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example,dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never

# TLS certificates (needed for GnuTLS)
TLS_CACERT      /etc/ssl/certs/ldap.pem
```

- **SSSD Configuration**

On crée le fichier de configuration **/etc/sssd/sssd.conf** et on y renseigne les parametres suivants :

```
GNU nano 4.8                                                            /etc/sss
[sssd]
services = nss, pam
config_file_version = 2
domains = default

[nss]

[pam]
offline_credentials_expiration = 60

[domain/default]
ldap_id_use_start_tls = True
cache_credentials = True
ldap_search_base = dc=rtn,dc=sn
id_provider = ldap
auth_provider = ldap
ldap_uri = ldap://server.rtn.sn
ldap_tls_reqcert = demand
ldap_tls_cacert = /etc/ssl/certs/ldap.pem
ldap_tls_cacertdir = /etc/ssl/certs
```

On donne ensuite les autorisations **600** puis on redemarre le service **sssd**

```
root@virtualbox:~# chmod 600 /etc/sssd/sssd.conf
root@virtualbox:~# systemctl restart sssd
root@virtualbox:~#
```

Note
sssd utilisera START_TLS par défaut pour les demandes d'authentification sur le serveur LDAP (le auth_provider), mais pas pour le id_provider. Si on souhaite également activer START_TLS pour le id_provider, on spécifie ***ldap_id_use_start_tls = true***

Création automatique d'un répertoire personnel
Pour activer la création automatique du répertoire de base, on exécute la commande suivante :
#sudo pam-auth-update --enable mkhomedir

```
pam-auth-update --enable mkhomedir
```

Une fois que tout cela est fait, vérifions qu'on peut se connecter au serveur LDAP en utilisant des connexions TLS vérifiées :

```
root@VirtualBox:/home/ismaila# ldapwhoami -x -ZZ -H ldap://server.rtn.sn
anonymous
```

NB :
Si on veut accéder à LDAPS (LDAP sur SSL), on modifie le fichier **/etc/default/slapd** du serveur comme suit :

```
  GNU nano 4.8                    /etc/default/slapd
# SLAPD_SERVICES="ldap://127.0.0.1:389/ ldaps:/// ldapi:///"
SLAPD_SERVICES="ldap:/// ldapi:/// ldaps:///"

# If SLAPD_NO_START is set, the init script will not start or
```

On rédemmare le service

```
root@acer:/home/ismaila# service slapd restart
```

Sur le client on rappelle la commande ldapwhoami sans l'option -ZZ

```
root@virtualbox:~# ldapwhoami -x -H ldaps://server.rtn.sn
anonymous
root@virtualbox:~#
```

VI Test

On peut maintenant passer aux tests.
Notre serveur LDAP possède l'entrée utilisateur suivante que nous allons utiliser pour le test :

```
  GNU nano 4.8
dn: uid=mbaye,ou=People,dc=rtn,dc=sn
objectClass: top
objectClass: posixAccount
objectClass: inetOrgPerson
objectClass: person
uid: mbaye
uidNumber: 2000
gidNumber: 2000
homeDirectory: /home/mbaye
loginShell: /bin/bash
mail:mbaye@rtn.sn
givenName: mbaye
cn: mbaye samb
userPassword: Passer123/
sn: Mbaye Samb
```

Vérifions si la machine cliente parvient à retrouver l'utilisateur « **mbaye** » du serveur LDAP.
Pour cela on tape les commandes **getent** et **id**

```
root@VirtualBox:/home/ismaila# getent passwd mbaye
mbaye:*:2000:2000:mbaye samb:/home/mbaye:/bin/bash
root@VirtualBox:/home/ismaila# id mbaye
uid=2000(mbaye) gid=2000 groupes=2000
root@VirtualBox:/home/ismaila#
```

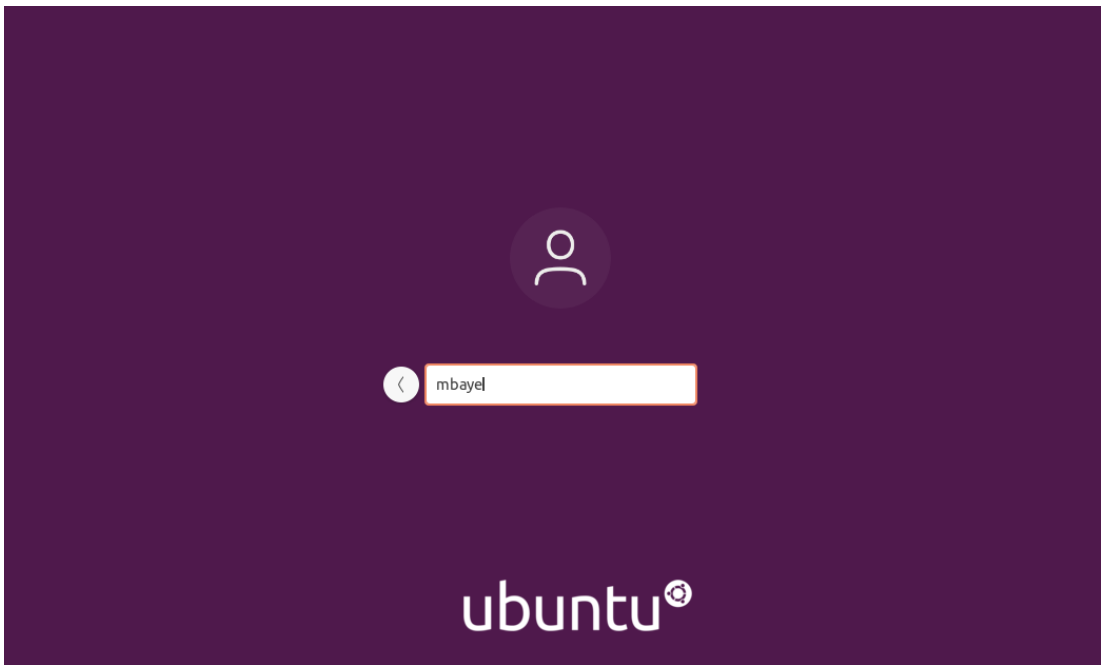
Maintenant on est mesure de s'authentifier en tant que « **mbaye** » avec la commande **login** :

```
root@VirtualBox:/home/ismaila# sudo login
VirtualBox login : mbaye
Mot de passe :
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

La maintenance de sécurité étendue pour Applications n'est pas activée.

3 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

13 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Dernière connexion : talaata 14 feebriy'e 2023 à 04:08:37 GMT sur pts/0
Création du répertoire « /home/mbaye ».
groups: impossible de trouver le nom pour le GID 2000
mbaye@VirtualBox:~$
```

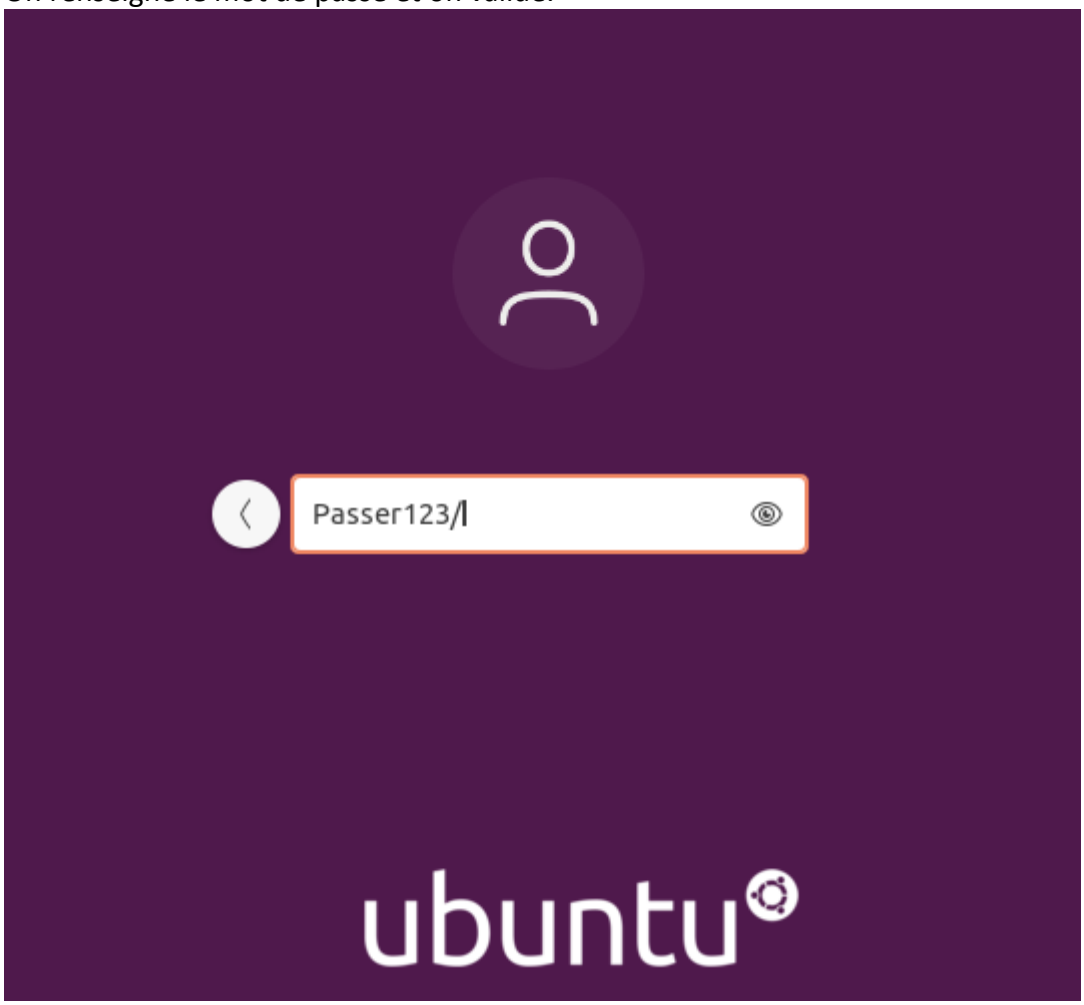
On voit bien qu'on s'est connecté avec le compte utilisateur « **mbaye** » se trouvant dans l'annuaire du serveur distant.
Quand on regarde les logs de slapd (à activer) de la machine distante, on s'aperçoit qu'une connexion TLS a été initialisée :

```
server slapd[3290]: conn=1007 op=6 SRCH attr=objectClass cn userPassword gidNumber memberuid modifyTimestamp modifyTimestamp
server slapd[3290]: conn=1007 op=6 SEARCH RESULT tag=101 err=0 nentries=0 text=
server slapd[3290]: conn=1008 fd=20 ACCEPT from IP=192.168.1.13:45404 (IP=0.0.0.0:389)
server slapd[3290]: conn=1008 op=0 EXT oid=1.3.6.1.4.1.1466.20037
server slapd[3290]: conn=1008 op=0 STARTTLS
server slapd[3290]: conn=1008 op=0 RESULT oid= err=0 text=
server slapd[3290]: conn=1008 fd=20 TLS established tls_ssf=256 ssf=256
server slapd[3290]: conn=1008 op=1 SRCH base="" scope=0 deref=0 filter="(objectClass=*)"
server slapd[3290]: conn=1008 op=1 SRCH attr=* altServer namingContexts supportedControl supportedExtension supportedFeature
rsion supportedSASLMechanisms domainControllerFunctionality defaultNamingContext lastUSN highestCommittedUSN
server slapd[3290]: conn=1008 op=1 SEARCH RESULT tag=101 err=0 nentries=1 text=
server slapd[3290]: conn=1008 op=2 BIND dn="uid=mbaye,ou=People,dc=rtn,dc=sn" method=128
server slapd[3290]: slap_global_control: unrecognized control: 1.3.6.1.4.1.42.2.27.8.5.1
server slapd[3290]: conn=1008 op=2 BIND dn="uid=mbaye,ou=People,dc=rtn,dc=sn" mech=SIMPLE ssf=0
```

Vérifier l'authentification GUI via OpenLDAP SSSD
Pour se connecter en mode graphique, on ferme la session en cours (ou redémarrer la machine cliente), sur l'interface de connexion GDM, on clique sur **Absent de la liste ?** pour avoir la possibilité de saisir un nom d'utilisateur et un mot de passe.



On renseigne le mot de passe et on valide.



Une fois la connexion réussie, on atterrit sur le bureau du compte mbaye : Ubuntu 20.04.

Conclusion

Nous venons d'installer, configurer et tester correctement le serveur (LDAP) et le client (SSSD+clientLDAP) sur Ubuntu 20.04. Nous allons désormais utiliser LDAP comme backend Kerberos dans la séquence suivante.

Séquence 3 : Kerberos et openLDAP comme backend

I Kerberos et LDAP

Kerberos prend en charge quelques backends de base de données. Celui par défaut est ce que nous avons utilisé jusque-là, est appelé db2. La documentation Types de bases de données présente toutes les options, dont l'une est LDAP.
Il existe plusieurs raisons pour lesquelles il faudrait que les principaux Kerberos soient stockés dans LDAP plutôt que dans une base de données locale sur disque. Il y a aussi des cas où ce n'est pas une bonne idée. Chaque site doit évaluer les avantages et les inconvénients. En voici quelques-unes :

- la réplication OpenLDAP est plus rapide et plus robuste que la réplication Kerberos native, basée sur une tâche cron
- La configuration des choses avec le backend LDAP n'est pas vraiment triviale et ne devrait pas être tentée par les administrateurs sans connaissance préalable d'OpenLDAP
- il peut y avoir une latence plus élevée dans la gestion des demandes lors de l'utilisation du backend OpenLDAPkrb5kdc
- si on a déjà configuré OpenLDAP pour d'autres choses, comme le stockage d'utilisateurs et de groupes, l'ajout des attributs Kerberos au même mélange peut être bénéfique et peut fournir une belle histoire intégrée

Cette séquence traite de la configuration d'un serveur Kerberos utilisant OpenLDAP comme base de données.

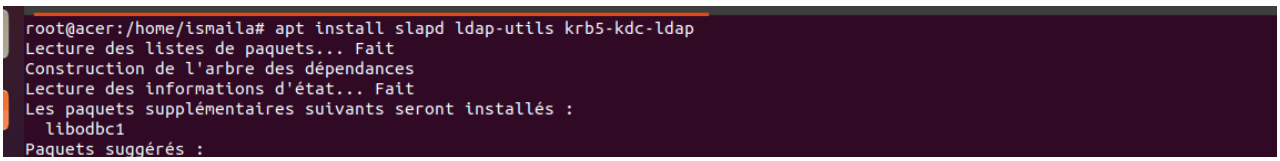
- ***Configuration d'OpenLDAP***

Nous devons installer le serveur OpenLDAP sur le même hôte que le KDC, pour simplifier la communication entre eux (déjà fait). Dans une telle configuration, nous pouvons utiliser le transport ldapi:///, qui se fait via un socket unix, et nous n'avons pas besoin de configurer des certificats SSL pour sécuriser la communication entre les services Kerberos et OpenLDAP.
Quand on souhaite utiliser un serveur OpenLDAP distant, ce qui est également possible, alors on faudra utiliser SSL pour la communication entre le KDC et ce serveur OpenLDAP.
Tout d'abord, le schéma nécessaire doit être chargé sur un serveur OpenLDAP disposant d'une connectivité réseau aux KDC.

II Installation et configuration

- Installez les paquets nécessaires (il est supposé qu'OpenLDAP est déjà installé) :

sudo apt install krb5-kdc-ldap krb5-admin-server (on déjà installer un seveur LDAP et kerberos voir plus haut)



- **Copier le schema dans /etc/ldap/schema**

#sudo cp /usr/share/doc/krb5-kdc-ldap/kerberos.schema.gz /etc/ldap/schema/

- **Ensuite, extraire le fichier : kerberos.schema.gz**

#sudo gunzip /etc/ldap/schema/kerberos.schema.gz

Le schéma kerberos doit être ajouté à l'arborescence cn=config. Ce fichier de schéma doit être converti au format LDIF avant de pouvoir être ajouté. Pour cela, nous utiliserons un outil d'aide : schema2ldif

#sudo apt install schema2ldif

```
oot@acer:/home/ismaila# apt install schema2ldif
ecture des listes de paquets... Fait
onstruction de l'arbre des dépendances
ecture des informations d'état... Fait
es NOUVEAUX paquets suivants seront installés :
 schema2ldif
 mis à jour, 1 nouvellement installés, 0 à enlever et 146 non mis à
l est nécessaire de prendre 14,9 ko dans les archives.
```

Pour importer le schéma, on exécute : **ldap-schema-manager -i kerberos.schema**

```
root@acer:/home/ismaila# ldap-schema-manager -i kerberos.schema

SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
executing 'ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/ldap/schema/kerberos.ldif'
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
adding new entry "cn=kerberos,cn=schema,cn=config"

root@acer:/home/ismaila#
```

Une fois le nouveau schéma chargé, indexons un attribut souvent utilisé dans les recherches :

```
root@acer:/home/ismaila# ldapmodify -Q -Y EXTERNAL -H ldapi:/// <<EOF
> dn: olcDatabase={1}mdb,cn=config
> add: olcDbIndex
> olcDbIndex: krbPrincipalName eq,pres,sub
> EOF
modifying entry "olcDatabase={1}mdb,cn=config"

root@acer:/home/ismaila#
```

On crée des entrées LDAP pour les entités administratives Kerberos qui contacteront le serveur openLDAP pour effectuer des opérations. Il y en a deux :

ldap_kdc_dn : doit disposer de droits de lecture sur le conteneur de domaine, le conteneur principal et les sous-arborescences de domaine.

ldap_kadmind_dn : doit disposer de droits de lecture et d'écriture sur le conteneur de domaine, le conteneur principal et les sous-arborescences de domaine

Voici la commande pour créer ces entités :

```
root@acer:/home/ismaila# ldapadd -x -D cn=admin,dc=rtn,dc=sn -W <<EOF
> dn: uid=kdc-service,dc=rtn,dc=sn
> uid: kdc-service
> objectClass: account
> objectClass: simpleSecurityObject
> userPassword: {CRYPT}x
> description: Account used for the Kerberos KDC
>
> dn: uid=kadmin-service,dc=rtn,dc=sn
> uid: kadmin-service
> objectClass: account
> objectClass: simpleSecurityObject
> userPassword: {CRYPT}x
> description: Account used for the Kerberos Admin server
> EOF
Enter LDAP Password:
adding new entry "uid=kdc-service,dc=rtn,dc=sn"

adding new entry "uid=kadmin-service,dc=rtn,dc=sn"

root@acer:/home/ismaila#
```

Maintenant, définissons un mot de passe pour eux.

```
root@acer:/home/ismaila# ldappasswd -x -D cn=admin,dc=rtn,dc=sn -W -S uid=kdc-service,dc=rtn,dc=sn
New password:
Re-enter new password:
Enter LDAP Password:
root@acer:/home/ismaila# ldappasswd -x -D cn=admin,dc=rtn,dc=sn -W -S uid=kadmin-service,dc=rtn,dc=sn
New password:
Re-enter new password:
Enter LDAP Password:
root@acer:/home/ismaila#
```

On peut les tester avec : ldapwhoami

```
root@acer:/home/ismaila# ldapwhoami -x -D uid=kdc-service,dc=rtn,dc=sn -W
Enter LDAP Password:
dn:uid=kdc-service,dc=rtn,dc=sn
root@acer:/home/ismaila#
```

Enfin, mettons à jour les listes de contrôle d'accès (ACL).
Nous devons insérer les nouvelles règles avant la dernière, pour contrôler l'accès aux entrées et attributs liés à Kerberos

```
#ldapmodify -Q -Y EXTERNAL -H ldapi:/// <<EOF
dn: olcDatabase={1}mdb,cn=config
add: olcAccess
olcAccess: {2}to attrs=krbPrincipalKey
  by anonymous auth
  by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
  by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
  by self write
  by * none
-
add: olcAccess
olcAccess: {3}to dn.subtree="cn=krbContainer,dc=example,dc=com"
  by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
  by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
  by * none
EOF
```

```
root@acer:/home/ismaila# ldapmodify -Q -Y EXTERNAL -H ldapi:/// <<EOF
> dn: olcDatabase={1}mdb,cn=config
> add: olcAccess
> olcAccess: {2}to attrs=krbPrincipalKey
>   by anonymous auth
>   by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
>   by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
>   by self write
>   by * none
> -
> add: olcAccess
> olcAccess: {3}to dn.subtree="cn=krbContainer,dc=rtn,dc=sn"
>   by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
>   by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
>   by * none
> EOF
modifying entry "olcDatabase={1}mdb,cn=config"

root@acer:/home/ismaila#
```

Vérifions les ACLs avec la commande **sudo slapcat -b cn=config**

```
root@acer: /etc/ldap                    root@acer: /home/ismaila
objectClass: olcMdbConfig
olcDatabase: {1}mdb
olcDbDirectory: /var/lib/ldap
olcSuffix: dc=rtn,dc=sn
olcAccess: {0}to attrs=userPassword by self write by anonymous auth by * non
 e
olcAccess: {1}to attrs=shadowLastChange by self write by * read
olcAccess: {2}to attrs=krbPrincipalKey by anonymous auth by dn.exact="uid=kd
 c-service,dc=rtn,dc=sn" read by dn.exact="uid=kadmin-service,dc=rtn,dc=sn"
 write by self write by * none
olcAccess: {3}to dn.subtree="cn=krbContainer,dc=rtn,dc=sn" by dn.exact="uid=
 kdc-service,dc=rtn,dc=sn" read by dn.exact="uid=kadmin-service,dc=rtn,dc=sn
 " write by * none
olcAccess: {4}to * by * read
olcLastMod: TRUE
olcRootDN: cn=admin,dc=rtn,dc=sn
```

Voilà, notre annuaire LDAP est maintenant prêt à servir de base de données principale Kerberos.

- ***Configuration KDC principale (LDAP)***

Une fois OpenLDAP configuré, il est temps de configurer le KDC. Ici LDAP et Kerberos sont sur la même machine.

On édite le fichier **/etc/krb5.conf** en rajoutant les paramètres suivants dans la section **[realms]**

default_domain = example.com

database_module = openldap_ldapconf

```
 GNU nano 4.8                                                    /etc/krb5.conf

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

[realms]
        RTN.SN = {
                kdc = sever.rtn.sn
                admin_server = server.rtn.sn
                default_domain = RTN.SN
                database_module = openldap_ldapconf
        }
        ATHENA.MIT.EDU = {
                kdc = kerberos.mit.edu
                kdc = kerberos-1.mit.edu
```

Ensuite, on ajoute également ces nouvelles sections :

```
[dbdefaults]
    ldap_kerberos_container_dn = cn=krbContainer,dc=example,dc=com

[dbmodules]
    openldap_ldapconf = {
        db_library = kldap

                                    # if either of these is false, then the ldap_kdc_dn needs to
                                    # have write access
                                    disable_last_success = true
                                    disable_lockout  = true

        # this object needs to have read rights on
        # the realm container, principal container and realm sub-trees
        ldap_kdc_dn = "uid=kdc-service,dc=example,dc=com"

        # this object needs to have read and write rights on
        # the realm container, principal container and realm sub-trees
        ldap_kadmind_dn = "uid=kadmin-service,dc=example,dc=com"

        ldap_service_password_file = /etc/krb5kdc/service.keyfile
        ldap_servers = ldapi:///
        ldap_conns_per_server = 5
    }
```

```
[dbdefaults]
        ldap_kerberos_container_dn = cn=krbContainer,dc=rtn,dc=sn
[dbmodules]
        openldap_ldapconf = {
                db_library = kldap

                                # if either of these is false, then the ldap_kdc_dn needs to
                                # have write access
                                disable_last_success = true
                                disable_lockout  = true

                # this object needs to have read rights on
                # the realm container, principal container and realm sub-trees
                ldap_kdc_dn = "uid=kdc-service,dc=rtn,dc=sn"

                # this object needs to have read and write rights on
                # the realm container, principal container and realm sub-trees
                ldap_kadmind_dn = "uid=kadmin-service,dc=rtn,dc=sn"

                ldap_service_password_file = /etc/krb5kdc/service.keyfile
                ldap_servers = ldapi:///
                ldap_conns_per_server = 5
        }
```

On utilise l'utilitaire kdb5_ldap_util pour créer le domaine :

#kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn create -subtrees dc=rtn,dc=sn -r RTN.SN -s -H ldapi:///

```
root@acer:/home/ismaila# kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn create -subtrees dc=rtn,dc=sn -r RTN.SN -s -H ldapi:///
Password for "cn=admin,dc=rtn,dc=sn":
Initializing database for realm 'RTN.SN'
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:
root@acer:/home/ismaila#
```

On crée une réserve du mot de passe utilisé pour la liaison au serveur LDAP. On l'exécute une fois pour chaque **ldap_kdc_dn** et **ldap_kadmin_dn**.

#kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn stashsrvpw -f /etc/krb5kdc/service.keyfile uid=kdc-service,dc=rtn,dc=sn

#kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn stashsrvpw -f /etc/krb5kdc/service.keyfile uid=kadmin-service,dc=rtn,dc=sn

```
root@acer:/home/ismaila# kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn stashsrvpw -f /etc/krb5kdc/service.keyfile uid=kdc-service,dc=rtn,dc=sn
Password for "cn=admin,dc=rtn,dc=sn":
Password for "uid=kdc-service,dc=rtn,dc=sn":
Re-enter password for "uid=kdc-service,dc=rtn,dc=sn":
root@acer:/home/ismaila# kdb5_ldap_util -D cn=admin,dc=rtn,dc=sn stashsrvpw -f /etc/krb5kdc/service.keyfile uid=kadmin-service,dc=rtn,dc=sn
Password for "cn=admin,dc=rtn,dc=sn":
Password for "uid=kadmin-service,dc=rtn,dc=sn":
Re-enter password for "uid=kadmin-service,dc=rtn,dc=sn":
root@acer:/home/ismaila#
```

Le fichier contient maintenant des versions en texte clair des mots de passe utilisés par le KDC pour contacter le serveur LDAP **/etc/krb5kdc/service.keyfile**

On redémarre le KDC Kerberos et le serveur d'administration :

```
root@acer:/home/ismaila# sudo systemctl start krb5-kdc.service krb5-admin-server.service
root@acer:/home/ismaila#
```

On peut désormais ajouter des principaux Kerberos à la base de données LDAP.
Pour que le **ldap_kadmin_dn** puisse y écrire, nous devons d'abord mettre à jour les ACLs :

```
#ldapmodify -Q -Y EXTERNAL -H ldapi:/// <<EOF
dn: olcDatabase={1}mdb,cn=config
add: olcAccess
olcAccess: {4}to dn.subtree="ou=People,dc=rtn,dc=sn"
   by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
   by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
   by * break
EOF
```

```
root@acer:/home/ismaila# ldapmodify -Q -Y EXTERNAL -H ldapi:/// <<EOF
> dn: olcDatabase={1}mdb,cn=config
> add: olcAccess
> olcAccess: {4}to dn.subtree="ou=People,dc=rtn,dc=sn"
>   by dn.exact="uid=kdc-service,dc=rtn,dc=sn" read
>   by dn.exact="uid=kadmin-service,dc=rtn,dc=sn" write
>   by * break
> EOF
modifying entry "olcDatabase={1}mdb,cn=config"

root@acer:/home/ismaila#
```

Et maintenant, nous pouvons spécifier le principal :
Avant cela on crée un utilisateur « **nasry** » dans LDAP.

```
  GNU nano 4.8                                                    nasry.ldif
dn: uid=nasry,ou=People,dc=rtn,dc=sn
uid: nasry
objectClass: inetOrgPerson
objectClass: posixAccount
cn: Nasry Nasry
sn: Nasry
givenName: Nasry
mail: nasry@rtn.sn
uidNumber: 10001
gidNumber: 10001
loginShell: /bin/bash
homeDirectory: /home/nasry
```

ldapadd pour l'ajouter

```
root@acer:/home/ismaila/Bureau# ldapadd -xW -D "cn=admin,dc=rtn,dc=sn" -f nasry.ldif
Enter LDAP Password:
adding new entry "uid=nasry,ou=People,dc=rtn,dc=sn"

root@acer:/home/ismaila/Bureau#
```

Pour ajouter des principaux Kerberos à la base de données LDAP on utilise l'option **-x**

```
root@acer:/home/ismaila# kadmin.local
Authenticating as principal ismaila/admin@RTN.SN with password.
kadmin.local:  addprinc -x dn=uid=nasry,ou=People,dc=rtn,dc=sn nasry
WARNING: no policy specified for nasry@RTN.SN; defaulting to no policy
Enter password for principal "nasry@RTN.SN":
Re-enter password for principal "nasry@RTN.SN":
Principal "nasry@RTN.SN" created.
kadmin.local:
```

Les attributs **krbPrincipalName**, **krbPrincipalKey**, **krbLastPwdChange** et **krbExtraData** doivent maintenant être ajoutés à l'objet utilisateur *uid=nasry,ou=People,dc=rtn,dc=sn*.

```
cn: Nasry Nasry
sn: Nasry
givenName: Nasry
mail: nasry@rtn.sn
uidNumber: 10001
gidNumber: 10001
loginShell: /bin/bash
homeDirectory: /home/nasry
krbLoginFailedCount: 0
krbPrincipalName: nasry@RTN.SN
krbLastPwdChange: 20230214214300Z
krbExtraData:: AAJkAOxjaXNtYWlsYS9hZG1pbkBSVE4uU04A
krbExtraData:: AAgBAA==
```

Conclusion

Nous venons d'installer, configurer et tester correctement le serveur Kerberos avec openLDAP comme backend sur le serveur Ubuntu 20.04. Nous allons maintenant pour coupler Kerberos/LDAP à SSSD dans la séquence suivante.

Séquence 4 : SSSD, LDAP et Kerberos

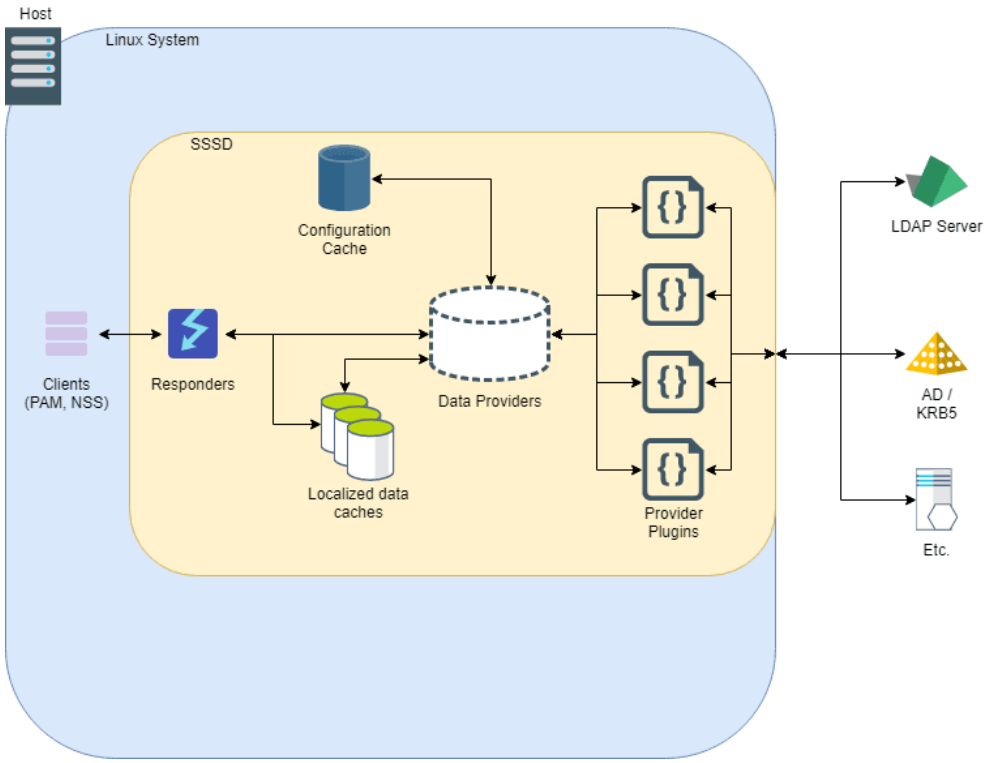
Enfin, nous pouvons coupler le tout dans une configuration très similaire à Active Directory en termes de technologies utilisées : utiliser LDAP pour les utilisateurs et les groupes, et Kerberos pour l'authentification.

Conditions préalables, hypothèses et exigences

Pour cette configuration, nous aurons besoin de :

- un serveur OpenLDAP existant utilisant le schéma RFC2307 pour les utilisateurs et les groupes. La prise en charge SSL est recommandée, mais pas strictement nécessaire car l'authentification dans cette configuration est effectuée via Kerberos et non LDAP.
- un serveur Kerberos. Il n'est pas nécessaire d'utiliser le backend OpenLDAP
- un hôte client où nous installerons et configurerons SSSD

Rappelons que nous avons déjà installé Kerberos et LDAP (voir séquences précédentes) sur le serveur et SSSD sur le client.



Architecture de SSSD

Sur l'hôte client, on installe les paquets suivants :

#sudo apt install sssd-ldap sssd-krb5 ldap-utils krb5-user

```
root@VirtualBox:/home/ismaila# sudo apt install sssd-ldap sssd-krb5 ldap-utils krb5-user
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
ldap-utils est déjà la version la plus récente (2.4.49+dfsg-2ubuntu1.9).
sssd-krb5 est déjà la version la plus récente (2.2.3-3ubuntu0.9).
sssd-krb5 passé en « installé manuellement ».
sssd-ldap est déjà la version la plus récente (2.2.3-3ubuntu0.9).
krb5-user est déjà la version la plus récente (1.17-6ubuntu4.2).
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  chromium-codecs-ffmpeg-extra gir1.2-goa-1.0 gstreamer1.0-vaapi libgstreamer-plugins-bad1.0-0 libva-wayland2
Veuillez utiliser « sudo apt autoremove » pour les supprimer.
0 mis à jour, 0 nouvellement installés, 0 à enlever et 3 non mis à jour.
root@VirtualBox:/home/ismaila#
```

I SSSD Configuration

On crée le fichier de configuration /etc/sssd/sssd.conf, avec les autorisations 600 ce qu'on a déjà fait dans la séquence 2. Donc on va juste le modifié en rajoutant quelques paramètres :

```
root@VirtualBox: /etc/sssd
  GNU nano 4.8                                    sssd.conf
[sssd]
config_file_version = 2
services = nss,pam
domains = rtn.sn

[nss]

[pam]

[domain/rtn.sn]
debug_level = 1
auth_provider = krb5
krb5_server = server.rtn.sn
krb5_realm = rtn.sn
cache_credentials = true

access_provider = simple
chpass_provider = krb5

id_provider = ldap
ldap_id_use_start_tls = true
ldap_uri = ldap://server.rtn.sn
ldap_search_base = dc=rtn,dc=sn
ldap_tls_reqcert = demand
ldap_tls_cacert = /etc/ssl/certs/ldap.pem
ldap_tls_cacertdir = /etc/ssl/certs
```

On redémarre le service SSSD

#sudo systemctl start sssd.service

```
coumba@VirtualBox:~$ déconnexion
root@VirtualBox:~# nano /etc/sssd/sssd.conf
root@VirtualBox:~# getent passwd nasry
nasry:*:10001:10001:Nasry Nasry:/home/nasry:/bin/bash
root@VirtualBox:~# id nasry
uid=10001(nasry) gid=10001 groupes=10001
root@VirtualBox:~#
```

Pour activer la création automatique du répertoire de base, on exécute la commande suivante :
#sudo pam-auth-update --enable mkhomedir

II Test

Notre serveur LDAP possède déjà l'entrée utilisateur « nasry » que nous allons utiliser pour dans ce test :

```
  GNU nano 4.8                                                  nasry.ldif
dn: uid=nasry,ou=People,dc=rtn,dc=sn
uid: nasry
objectClass: inetOrgPerson
objectClass: posixAccount
cn: Nasry Nasry
sn: Nasry
givenName: Nasry
mail: nasry@rtn.sn
uidNumber: 10001
gidNumber: 10001
loginShell: /bin/bash
homeDirectory: /home/nasry
```

Rappelons que l'utilisateur *nasry* n'a pas *d'attribut userPassword*.

Vérifions sur la machine client si l'utilisateur nasry est connu du système.
Pour cela on tape les commandes suivantes : getent et id

```
root@VirtualBox:~# getent passwd nasry
nasry:*:10001:10001:Nasry Nasry:/home/nasry:/bin/bash
root@VirtualBox:~# id nasry
uid=10001(nasry) gid=10001 groupes=10001
root@VirtualBox:~#
```

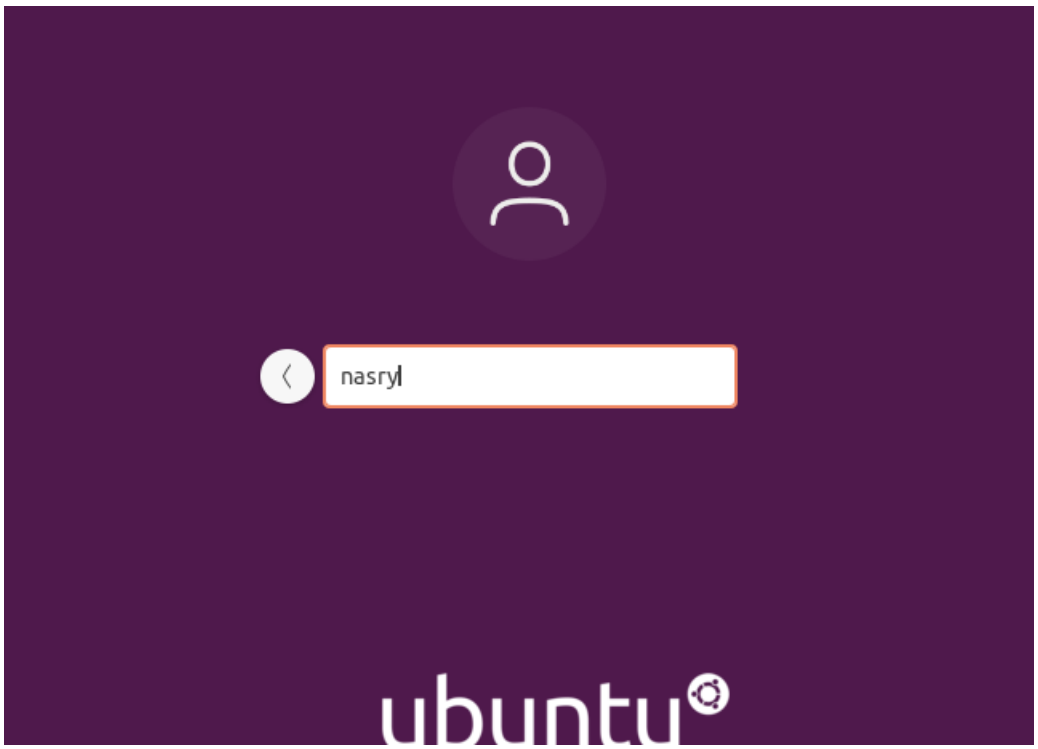
Essayons de se connecter en tant que nasry.
Nous allons nous connecter à l'aide du mot de passe kerberos (mot de passe définit lors de la création du principal nasry dans la séquence précédente) et des informations utilisateur du serveur LDAP.

```
root@virtualbox:~# login
virtualbox login : nasry
Mot de passe :
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-58-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

La maintenance de sécurité étendue pour Applications n'est pas activée.

4 mises à jour peuvent être appliquées immédiatement.
Pour afficher ces mises à jour supplémentaires, exécuter : apt list --upgradable

13 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Dernière connexion : talaata 14 feebriy'e 2023 à 22:26:22 GMT sur pts/0
Création du répertoire « /home/nasry ».
groups: impossible de trouver le nom pour le GID 10001
nasry@virtualbox:~$
```

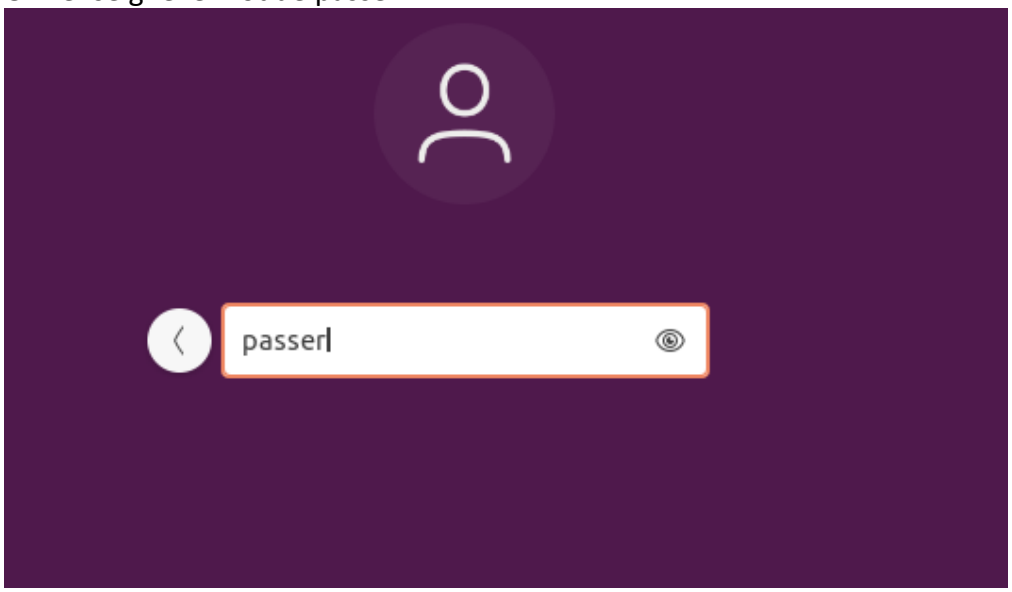
Nasry et bien connecté on peut utiliser l'utilitaire klist pour afficher des informations sur le ticket d'octroi de ticket (TGT)

```
groups: impossible de trouver le nom pour le GID 10001
nasry@VirtualBox:~$ klist
Ticket cache: FILE:/tmp/krb5cc_10001_SR7Oby
Default principal: nasry@rtn.sn

Valid starting       Expires              Service principal
14.02.2023 13:51:36  14.02.2023 23:51:36  krbtgt/rtn.sn@rtn.sn
        renew until 15.02.2023 13:51:36
nasry@VirtualBox:~$
```

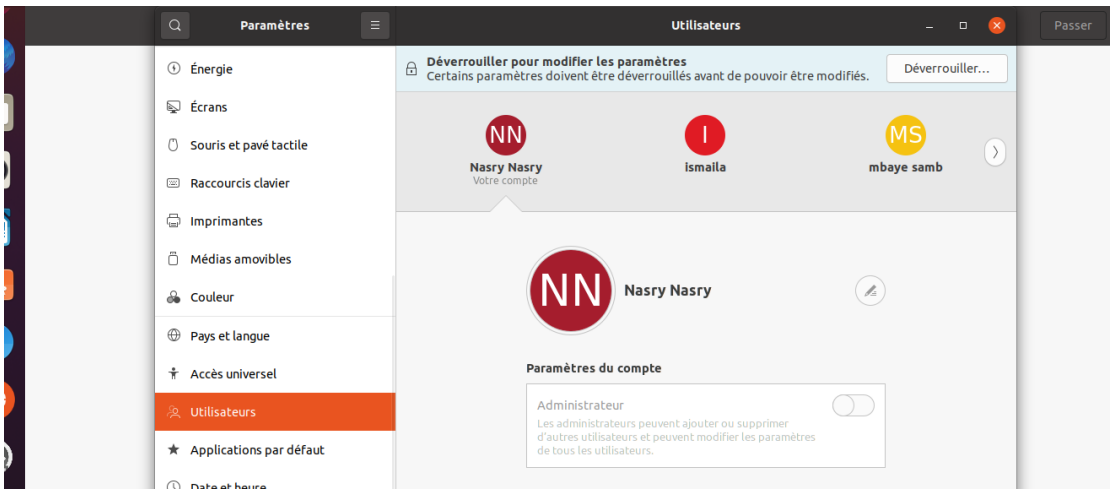
Notons que nasry peut aussi se connecter en mode graphique. Pour cela on **ferme la session** puis **Absent de la liste ?** pour saisir le nom :



On renseigne le mot de passe :



Si tout se passe, **nasry** accéder à son Bureau en mode graphique :



La prochaine fois que vous vous connecterez, l'utilisateur AD sera répertorié comme s'il s'agissait d'un utilisateur local :

Conclusion

Dans ce rapport, nous avons exposé les étapes de déploiement d'un service d'annuaire et d'authentification qui consiste à utiliser SSSD, Kerberos avec LDAP pour fournir un moyen simple pour un client d'utiliser des comptes utilisateurs et groupes LDAP existants pour se connecter à une machine Linux. C'est un exemple concret pour faciliter le rôle des administrateurs de domaine. Notre travail s'est déroulé en quatre étapes.

Nous avons commencé par mettre en place un système d'authentification avec Kerberos (serveur et client). Dans la deuxième phase, nous avons montré comment le couple SSSD et LDAP permet à une machine cliente d'utiliser un compte UNIX distant. La troisième étape concernait l'utilisation d'un annuaire LDAP comme backend de kerberos. Dans la dernière phase, nous avons combiné le tout (SSSD, Keberos+LDAP) dans une configuration similaire à Active Directory : nous avons utilisé LDAP pour les utilisateurs, et Kerberos pour l'authentification.

Ce rapport fut très instructif et pédagogique, il nous a permis de simplifier l'administration système de l'accès utilisateur authentifié et autorisé impliquant des hôtes distincts au prix de quelques efforts.

Il conviendra, cependant, d'intégrer des utilisateurs Active Directory pour mettre en place un environnement hybride Windows/Linux.

Annexes : Quelques commande utiles

Commande Kerberos

1. Ajouter un utilisateur :

```
kadmin: addprinc user
```

Le nom de domaine par défaut est ajouté au nom du principal par défaut

2. Supprimer un utilisateur :

```
kadmin: delprinc user
```

3. Liste des principaux :

```
kadmin: listprincs
```

4. Ajouter un service :

```
kadmin: addprinc service/Nomd'hôte
```

Le nom de domaine par défaut est ajouté au nom du principal par défaut

5. Supprimer un utilisateur :

```
kadmin: delprinc service/Nomd'hôte
```

6. Sur le KDC Kerberos, créez un principal de service et générez un fichier keytab. Afin de sélectionner un secret aléatoire, nous passons le paramètre **randkey**. Sinon, on nous demanderait de taper un mot de passe.

```
$ sudo kadmin.local
kadmin> addprinc -randkey service/Nomd'hôte
kadmin> ktadd service/Nomd'hôte
```

Commande Ldap

1. Ajouter une entrée :

```
ldapadd -xW -D "cn=admin,dc=ec2lt,dc=sn" -f nomFichier.ldif
```

1. Lister les entrées :

```
ldapsearch -xLLLWD "cn=admin,dc=ec2lt,dc=sn" -b "dc=ec2lt,dc=sn"
```

1. Modifier une entrée :

```
ldapmodify -x -D "cn=admin,dc=ec2lt,dc=sn" -w passer -f nomFichier.ldif
```

Le nom de domaine par défaut est ajouté au nom du principal par défaut

1. Supprimer une entrée :

```
ldapdelete -v -D "cn=admin,dc=rtn,dc=sn" -W "uid=berenger,ou=People,dc=rtn,dc=sn"
```