# BAB 4 IMPLEMENTASI

## 4.1 Instalasi

### 4.1.1 Kerberos

#### 4.1.1.1 Instalasi Kerberos V5

Kerberos diperlukan pada satu platform dengan struktur *single directory tree* yang berisi file-file sumber maupun file-file objek sehingga menjadi lebih sederhana dibandingkan dengan pengimplementasian Kerberos pada bermacam-macam platform. Mekanisme yang kedua membutuhkan proses *build* tersendiri untuk masing-masing platform dan tidak akan dijelaskan dalam penelitian ini.

Prosedur *build* yang harus dilakukan yaitu :

1. *cd /krb5-1.6.3/src*
2. *./configure*
3. *make*

Sesudah proses *build* Kerberos, selanjutnya adalah instalasi binary. Dapat dilakukan dengan perintah *make install.*

Pengujian proses *build* tersebut apakah berhasil atau tidak dengan perintah *make check* serta harus dilakukan dari level paling atas dari direktori *build.*

#### 4.1.1.2 Instalasi KDC

KDC (*Key distribution center*) menerbitkan tiket Kerberos. Masing-masing KDC berisi salinan database Kerberos. Master KDC berisi salinan utama dari

database yang disebarkan ke *slave-slave* KDC pada waktu-waktu tertentu. Semua perubahan database termasuk perubahan *password* dilakukan pada master KDC. *Slave* KDC menyediakan *ticket granting service,* tetapi tidak menyediakan administrasi database. Oleh sebab itulah *client* masih bisa memperoleh tiket meskipun master KDC tidak tersedia. Namun dalam penelitian ini hanya akan diadakan sebuah master KDC tanpa mengadakan replikasinya.

Modifikasi file-file konfigurasi yaitu *krb5.conf* dank dc.*conf* ditujukan untuk menggambarkan informasi-informasi yang sesuai (seperti *hostname* dan nama *realm*).

File *krb.conf* berisikan informasi konfigurasi Kerberos, termasuk lokasi KDC dan server admin pada *realm* yang dibangun, *realm* default dan aplikasi Kerberos, dan pemetaan *hostname* ke dalam *realm* Kerberos. Normalnya, *krb5.conf* terinstal didirektori */etc.*

Table 4.1 berisi nama bagian-bagian dalam file *krb5.conf* beserta deskripsinya. File *krb5.conf* dapat menyertakan seluruh bagian ataupun hanya beberapa bagian saja.

**Tabel 4.1 Komponen File *krb5.conf***

| **Nama Bagian** | **Deskripsi** |
|---|---|
| *Libdefaults* | Berisi nilai-nilai default yang digunakan oleh library Kerberos V5 |
| *Login* | Berisi nilai-nilai default yang digunakan oleh program login Kerberos V5 |
| *Appdefaults* | Berisi nilai-nilai default yang bisa digunakan oleh aplikasi Kerberos |
| *Realms* | Berisi subbagian-subbagian yang berisi nama-nam *realm.* Masing-masing subbagian menggambarkan informasi mengenai |

| | |
|---|---|
| | *realm* tertentu, termasuk dimana ditemukan server Kerberos untuk suatu *realm* |
| *Domain_realm* | Berisi relasi-relasi yang memetakan nama domain dan subdomain ke nama *realm* Kerberos. Ini digunakan oleh program-program untuk menentukan *realm* yang mana yang menjadi domain *realm* dari *host* tersebut |
| *Logging* | Berisi relasi-relasi yang menentukan proses log program Kerberos |
| *Capaths* | Berisi jalur-jalur otentikasi yang digunakan dengan otentikasi *cross-realm* langsung. Entri dalam bagian ini digunakan oleh *client* untuk menentukan *realm* lanjutan pada *cross-realm authentication.* |

Gambar 4.1 memperlihatkan contoh file *krb5.conf.* Pada gambar tersebut tidak seluruh bagian *krb5.conf* yang disebutkan dalam table 4.1 diikutsertakan. Hanya beberapa bagian yang diperlukan untuk pengimplementasian serta pengujian saja.

```
  GNU nano 2.0.6        File: /etc/krb5.conf

[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dcfault_realm = AUTH.LANGHUA
 dns_lookup_realm = false
 dns_lookup_kdc = false
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 AUTH.LANGHUA = {
  kdc = auth.langhua:88
 # kdc = dc-02.auth.langhua:88
  admin_server = auth.langhua:749
  default_domain = auth.langhua
 }

[domain_realm]
  .auth.langhua = AUTH.LANGHUA
  auth.langhua = AUTH.LANGHUA

[kdc]
  profile = /var/kerberos/krb5kdc/kdc.conf

[appdefaults]
  pam = {
    debug = false
    ticket_lifetime = 36000
    renew_lifetime = 36000
    forwardable = yes
    krb4_convert = false
```

**Gambar 4.1 File krb.conf**

Dari gambar tersebut dapat diketahui bahwa secara default *realm* yang ada adalah AUTH.LANGHUA, sedangkan mesin KDC terdapat pada *host* kerberos.auth.langhua. Server admin juga terdapat pada *host* yang sama yaitu kerberos.auth.langhua.

File *kdc.conf* berisi konfigurasi KDC, termasuk default-default yang digunakan untuk menerbitkan tiket Kerberos. Biasanya *kdc.conf* terinstal di direktori */usr/local/var/krb5kdc. Kdc.conf* memiliki format yang sama dengan *krb5.conf.*

Bagian-bagian yang menyusun file *kdc.conf* diperlihatkan pada table 4.2 berikut ini.

**Tabel 4.2 Komponen File *kdc.conf***

| Nama Bagian | Deskripsi |
|---|---|
| *Kdcdefaults* | Berisi nilai-nilai default untuk proses KDC secara keseluruhan |
| *Realms* | Barisi subbagian-subbagian yang berisi nama-nama *realm*. Masing-masing subbagian menggambarkan informasi mengenai *realm* tertentu, termasuk dimana ditemukan server Kerberos untuk suatu *realm* |
| *Logging* | Berisi relasi-relasi yang menentukan cara proses log program-program Kerberos |

#### 4.1.1.3 Pembuatan Database Kerberos

Selanjutnya perintah *kdb5_util* akan digunakan pada master KDC untuk membuat database Kerberos dan *stash* file apabila diinginkan. *Stash* file merupakan salinan local *master key*. *Stash* file ini digunakan untuk mengotentikasi KDC itu sendiri sebelum menjalankan daemon *kadmind dan krb5kdc* (seperti bagian dari rangkaian booting mesin). Jika ditentukan untuk menginstal *stash* file ini, KDC akan

meminta master key setiap kali dijalankan. Ini berarti KDC tidak akan dijalankan secara otomatis, misalnya setelah sistem boot.

Hal penting lainnya yang perlu diperhatikan adalah bahwa *kdb5_util* akan meminta *master key* untuk database Kerberos. Kunci ini dapat berupa string. Kunci yang baik adalah yang dapat diingat tetapi tak seorangpun mampu menebaknya.

Gambar 4.2 berikut ini memperlihatkan mekanisme pembuatan database Kerberos dan *stash* file pada master KDC menggunakan perintah *kdb_util.*

```
[root@kemas laptop -]# /usr/local/sbin/kdb5_util create -r AUTH.LANGHUA -s
Loading random data
Initializing database '/usr/local/var/krb5kdc/principal' for realm 'AUTH.LANGHUA',
master key name 'K/M@AUTH.LANGHUA'
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:
```

**Gambar 4.2 Mekanisme Pembuatan Database Kerberos**

Langkah-langkah diatas akan membentuk lima file didirektori yang dispesifikasikan di *kdc.conf.* Kelima file tersebut yaitu file database Kerberos, *principal.db* dan *principal.ok;* kemudian file administrasi database Kerberos, *principal.kadm5;* file penguncian administrasi database, *principal.kadm5.lock*; dan stash file, k5stash. Direktori default adalah */usrlocal/var/krb5kdc.* Jika tidak menginginkan *stash file,* maka perintah diatas dijalankan tanpa –s.

### 4.1.1.4 Access Control List User Kerberos

Berikut adalah pembuatan file *Access Control List,* dan meletakkan sedikitnya satu *principal* yang bertindak sebagai administrator. File ini digunakan oleh daemon *kadmind* untuk mengatur *principal* yang dapat melihat dan memodifikasi hak akses pada file database Kerberos. Nama file tersebut seharusnya

sesuai dengan nilai yang telah diatur pada *kdc.conf* untuk *acl_file*. Nama default yang digunakan adalah */usr/local/var/krb5kdc/kadm5.acl.*

Format file tersebut sebagai berikut :

*Kerberos_principal permission [target_principal][restriction]*

*Principal* Kerberos dapat mengikutsertakan "*" (wildcard) sehingga jika menginginkan semua *principal* dangan *instance* "admin" memiliki hak akses keseluruhan pada database, dapat dituliskan "*/*admin@REALM*" dimana "*REALM*" disini adalah nama *realm* Kerberos yang telah didefinisikan.

Berikut ini isi dari file *kadm5.acl* :

**/admin@AUTH.LANGHUA* *

*krbadm@AUTH.LANGHUA* *

*ldapadm@AUTH.LANGHUA* *

*ldap/auth.langhua@AUTH.LANGHUA* *

Pada file diatas, seluruh *principal* pada *realm* AUTH.LANGHUA dengan *instance* admin memiliki keseluruhan hak akses administrative. *ldap/auth.langhua@AUTH.LANGHUA* merupakan *service principal* untuk layanan LDAP.

### 4.1.1.5 Penambahan Administrator pada Database Kerberos

Langkah selanjutnya adalah menambahkan *principal* administrative pada database Kerberos. Untuk melakukan ini digunakan *kadmin.local* pada master KDC. *Principal* administrative yang dibuat harus ditambahkan pada file acl.

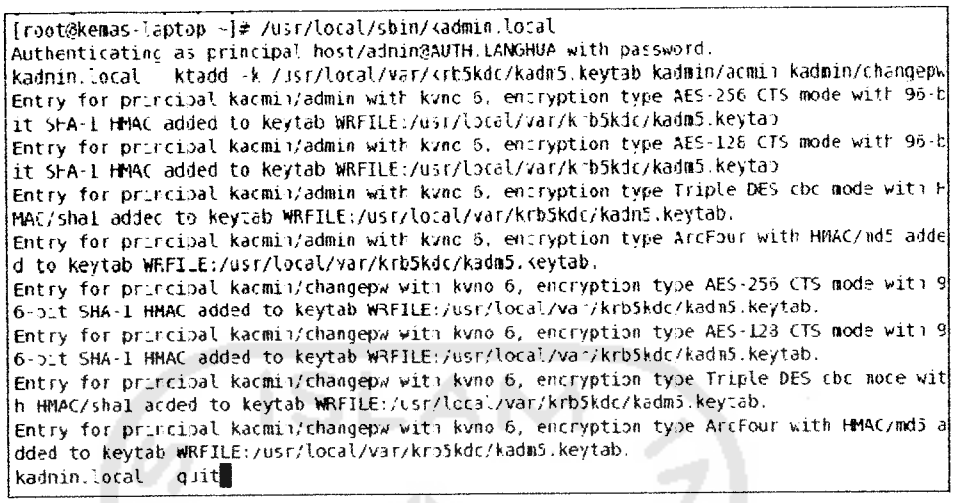
```
[root@kemas-laptop ~]# /usr/local/sbin/kdb5_util create -r AUTH.LANGHUA -s
Loading random data
Initializing database '/usr/local/var/krb5kdc/principal' for realm 'AUTH.LANGHUA',
master key name 'K/M@AUTH.LANGHUA'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
kdb5_util: File exists while creating database '/usr/local/var/krb5kdc/principal'
[root@kemas-laptop ~]# /usr/local/sbin/kadmin.local
Authenticating as principal host/admin@AUTH.LANGHUA with password.
kadmin.local:  addprinc admin/admin@AUTH.LANGHUA
WARNING: no policy specified for admin/admin@AUTH.LANGHUA; defaulting to no policy
Enter password for principal "admin/admin@AUTH.LANGHUA":
Re-enter password for principal "admin/admin@AUTH.LANGHUA":
```

**Gambar 4.3 Mekanisme Penambahan Administrator**

Mekanisme penambahan *principal* admin ditunjukkan melalui gambar 4.3 diatas. Operasi *addprinc* akan menambahkan *principal* pada database Kerberos. Perintah yang digunakan menurut gambar 4.3 tersebut adalah *addprinc* admin/admin@AUTH.LANGHUA yang akan menambahkan *user principal* admin dengan *instance* admin pada *realm AUTH.LANGHUA.*

### 4.1.1.6 Pembuatan Keytab Kadmind

*Keytab kadmin* adalah kunci sukses yang akan mengaktifkan daemon *kadmind4* dan *v5passwdd* yang digunakan untuk mendekripsikan tiket administrator maupun *client* Kerberos. Sebelumnya perlu membuat *keytab kadmin* dengan entri-entri *principal kadmin/admin* dan *kadmin/changepw. Principal-principal* ini diletakkan pada database Kerberos secara otomatis ketika dibuat. Gambar 4.4 memperlihatkan tahapan-tahapan proses tersebut.




**Gambar 4.4 Mekanisme Penambahan *keytab***

Sesuai dengan argument –k, *ktadd* akan menyimpan *keytab* terekstraksi pada */usr/local/sbin/kadmind.*

### 4.1.2 SASL

Computing Services Department di Carnigie Mellon University membuat library-librari SASL yang bisa diperoleh dari [illegible]. Library-librari SASL versi 2 mendukung beberapa mekanisme SASL, diantaranya:

1. ANONYMOUS
2. CRAM-MD5
3. DIGEST-MD5
4. GSSAPI (MIT Kerberos 5 atau Heimdal Kerberos 5)
5. KERBEROS_V4
6. PLAIN

Alasan utama integrasi SASL dengan LDAP adalah otentikasi Kerberos akan dijalankan saat mengakses server LDAP. Demi kepentingan fleksibilitas dalam

otentikasi, akan dikembangkan server menggunakan dukungan SASL. Instalasi library SASL diperlukan dalam penelitian ini ditujukan untuk otentikasi pihak ketiga, dalam hal ini adalah Kerberos. Lebih khusus lagi menjembatani integrasi LDAP dengan Kerberos untuk keperluan inilah diperlukan adanya library SASL.

Proses pembangunan SASL tidak jauh berbeda dengan paket-paket yang lain. Dalam kesempatan kali ini digunakan distribusi Cyrus SASL versi 2. Gambar 4.5 menggambarkan proses pembangunan yang dilakukan pada distribusi Cyrus SASL.

```
Cd cyrus-sasl-2.1.22
./configure
Make
./bin/su -c "make install ; ln -s /usr/local/lib/sasl2 /usr/lib/sasl2"
```

**Gambar 4.5 Proses Pembangunan CyrusSASL**

### 4.1.3 LDAP

#### 4.1.3.1 Instalasi OpenLDAPv3

Dalam penelitian ini digunakan distribusi OpenLDAP-2.4.15 sebagai *software* penyedia protokol LDAP. Setelah file sumber diekstrak, perintah berikut ini akan menjalankan konfigurasi yang telah ditentukan :

1. *./configure*
2. *Make depend*
3. *Make*
4. *Make test*
5. *Make install*

**Tabel 4.3 Paket Instalasi OpenLDAP**

| Nama | Deskripsi |
|---|---|
| *Libexec/slapd* | Server LDAP |
| *Libexec/slurpd* | Replikasi LDAP |
| *Bin/ldapadd*<br>*Bin/ldapmodify*<br>*Bin/ldapdelete*<br>*Bin/ldapmodrdn* | Perintah untuk menambah, memodifikasi, dan menghapus entri dari server LDAP. |
| *Bin/ldapsearch*<br>*Bin/ldapcompare* | Perintah untuk mencari sebuah direktori LDAP dan membandingkan atribut tertentu pada sebuah entri |
| *Sbin/slappasswd* | Perintah yang akan membuat hash *password* yang cocok untuk digunakan dalam *slapd.conf* |
| *Lib/libladp**<br>*Lib/liblber**<br>*Include/ldap*.h*<br>*Include/lber*.h* | SDK dari *client* OpenLDAP |
| *Bin/ldappasswd* | Digunakan untuk mengubah atribut *password* pada entri LDAP. Tool ini seperti */bin/passwd* |
| *Sbin/salpdadd*<br>*Sbin/slapcat*<br>*Sbin/slapindex* | Digunakan untuk memanipulasi data local yang tersimpan di *backend* yang digunakan oleh daemon *slapd* |

Paket OpenLDAP berisi library-librari *client, server,* dan *development.* Tabel 4.3 menggambarkan komponen-komponen yang ada dalam paket instalasi. Seluruh *pathname* adalah relative tergantung proses instalasi yang dilakukan, tetapi secara default ada pada */usr/local.*

### 4.1.3.2 Konfigurasi OpenLDAP

File *slapd.conf* adalah file sumber yang utama dari server OpenLDAP *standalone (slapd), replication helper daemon (slurpd),* dan perintah-perintah yang berhubungan seperti *slpadd* dan *slapcat.* Sebagai aturan utama, tools *client*

OpenLDAP seperti *ldapmodify* dan *ldapsearch* menggunakan *ldap.conf* untuk pengaturan default.

Dalam file konfigurasi unix tradisional, *slapd.conf* adalah sebuah file ASCII dengan aturan :

1. Baris kosong dan baris yang diawali dengan tanda # akan diabaikan.
2. Parameter dan nilai yang diasosiasikan dipisahkan oleh karakter spasi atau tabulasi.
3. Sebuah baris dengan spasi kosong dalam kolom pertama dianggap sebagai kelanjutan dari sebelumnya.

Untuk sebuah kebutuhan umum, file *slapd.conf* yang digunakan oleh OpenLDAPv3 dapat dibagi menjadi dua bagian. Bagian pertama berisi parameter-parameter yang akan berpengaruh pada tingkah laku keseluruhan dari server OpenLDAP. Bagian kedua berupa parameter-parameter yang berhubungan dengan *backend* database yang digunakan oleh daemon *slapd*. Konfigurasi *slapd.conf* yang menunjukkan bagian tersebut ditunjukkan oleh gambar 4.6.

```
# See slapd.conf(5) for details on configuration options.
# This file should NOT be world readable.
#

include         /etc/openldap/schema/corba.schema
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/duaconf.schema
include         /etc/openldap/schema/dyngroup.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/java.schema
include         /etc/openldap/schema/misc.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/openldap.schema
include         /etc/openldap/schema/ppolicy.schema
include         /etc/openldap/schema/collective.schema
include         /etc/openldap/schema/krb5-kdc.schema

# Allow LDAPv2 client connections.  This is NOT the default.
allow bind_v2

# Do not enable referrals until AFTER you have a working directory
# service AND an understanding of referrals.
#referral        ldap://root.openldap.org

pidfile         /var/run/openldap/slapd.pid
argsfile        /var/run/openldap/slapd.args

# Load dynamic backend modules:
# modulepath    /usr/lib/openldap # or /usr/lib64/openldap
```

**Gambar 4.6 Konfigurasi slapd.conf**

Bagian umum mulai dari awal file sampai dengan direktif database yang pertama. Mulai dari bagian database ditandai dengan parameter database, sampai dengan akhir file. Dimungkinkan juga untuk mendefinisikan banyak database yang disediakan oleh instalasi tunggal *slapd.* Masing-masing independent secara logika, dan file database yang diasosiasikan akan disimpan perbagian.

#### 4.1.3.3 Schema

OpenLDAPv3 secara default menyertakan beberapa skema yang popular untuk digunakan oleh administrator. Kebutuhan aplikasi yang akan digunakan dalam direktori akan menentukan skema yang dipakai. Secara default, file ini berlokasi didalam direktori */usr/local/etc/openldap/schema* setelah proses instalasi. Dalam file konfigurasi, parameter *include* digunakan untuk menentukan skema yang akan diikutkan kedalam server. Pendefinisian skema dalam file konfigurasi ditunjukkan oleh gambar 4.7.

```
include         /etc/openldap/schema/corba.schema
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/duaconf.schema
include         /etc/openldap/schema/dyngroup.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/java.schema
include         /etc/openldap/schema/misc.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/openldap.schema
include         /etc/openldap/schema/ppolicy.schema
include         /etc/openldap/schema/collective.schema
include         /etc/openldap/schema/krb5-kdc.schema
```

**Gambar 4.7 Pendefinisian schema**

Beberapa file schema yang biasanya disertakan dalam setiap instalasi OpenLDAP adalah :

1. *Corba.schema*

   Skema untuk menyimpan objek dari corba.

2. *Core.schema*

   Skema utama dari OpenLDAP. Skema ini mendefinisikan atribut dan objek dasar dari LDAPv3.

3. *Cosine.schema*

   Skema untuk mendukung direktori cosine dan X.500.

4. *Inetorgperson.schema*

   Skema yang mendefinisikan objectclass InetOrgPerson dan atribut yang diasosialisasikan. Biasanya untuk menyimpan informasi kontak dari orang lain.

5. *Java.schema*

   Skema untuk menyimpan objek serial dari java, objek java marshaled, objek remote java, atau referensi JDNI.

6. *Misc.schema*

   Skema yang mendefinisikan sebuah grup kecil dari objek dan atribut lainnya. Sekarang ini, file ini berisi skema yang dibutuhkan untuk mengimplementasikan routing email dalam sendmail yng berbasis LDAP.

7. *Nis.schema*

   Skema yang mendefinisikan atribut dan objek yang dibutuhkan untuk menggunakan LDAP dangen Network Information Service (NIS).

8. *Openldap.schema*

   Objek-objek lain yang digunakan oleh proyek OpenLDAP. Digunakan untuk informasi.

#### 4.1.3.4 Penyedia Data

Setelah bagian utama dari *slapd.conf,* dilanjutkan satu atau beberapa bagian dari database, masing-masing mendefinisikan sebuah partisi direktori. Sebuah bagian

database dimulai dengan direktif database dan seterusnya sampai ditemukan database selanjutnya. Parameter untuk database yang mungkin adalah :

1. Bdb

   Menggunakan database dari Berkeley DB 4. *Backend* ini sering digunakan untuk *indexing* dan *caching* yang akan meningkatkan performa, *backend* ini paling direkomendasikan untuk digunakan pada server OpenLDAP.
2. Ldbm

   Sebuah database yang diimplementasikan oleh GNU database manager atau paket *software* Sleepycat Berkeley DB. *Backend* ini implementasi yang lebih tua dari *backend bdb.*
3. Passwd

   *Backend* ini menggunakan sistem file passwd untuk memberikan sebuah antarmuka direktori.
4. Sql

   *Backend* ini menggunakan databse yang berbasis SQL. Databse yang didukung antara lain MySQL, Oracle, MSSQL, IBMDB2, PosgreSQL, dan timesten.
5. Shell

   *Backend shell* memperbolehkan untuk menggunakan database alternative (eksternal). Direktif ini mengijinkan kita untuk menentukan program eksternal yang dipanggil untuk masing-masing operasi dasar LDAPv3.

Gambar 4.8 menunjukkan contoh pendefinisian *backend.* Dari gambar tersebut dapat diketahui bahwa *backend* yang digunakan adalah bdb yaitu Berkeley DB 4 sesuai keterangan sebelumnya. Pada gambar 4.8 juga terdapat keterangan

mengnai *userid* yang bertindak sebagai root pada server LDAP. Dari gambar 4.8 tersebut didefinisikan bahwa yang bertindak sebagai rootdn adalah ldapadm pada *realm auth.langhua* menggunakan mekanisme otentikasi GSSAPI.

```
#######################################################################
# ldbm and/or bdb database definitions
#######################################################################

database        bdb
suffix          "dc=auth,dc=langhua"
rootdn          "uid=root,cn=auth.langhua,cn=gssapi,cn=auth"
# suffix                "dc=my-domain,dc=com"
# checkpoint    1024 15
# rootdn                "cn=Manager,dc=my-domain,dc=com"
# Cleartext passwords, especially for the rootdn, should
# be avoided.  See slappasswd(8) and slapd.conf(5) for details.
# Use of strong authentication encouraged.
# rootpw                secret
# rootpw                {SSHA}o/zDZrYFi9Zoz0gNO9QlWpNwz+WlHYuf
```

**Gambar 4.8 Pendefinisian backend**

Parameter index menyatakan atribut apa pada *slapd* yang seharusnya dipelihara indexnya. Index digunakan untuk mengoptimalkan pencarian, hampir sama seperti index yang digunakan oleh database relasional.

#### 4.1.3.5 Access Control List (ACL)

ACL direktori yang diberikan oleh OpenLDAP mempunyai sintaks yang sederhana, tetapi sangat fleksibel dan kuat dalam implementasinya.

1. *Wildcard (*)*

   Apabila cocok dengan setiap pengguna yang tersambung, termasuk koneksi anonym.

2. *Self*

   Cocok dengan DN dari pengguna yang sekarang tersambung, diasumsikan jika pengguna tersebut telah sukses diotentikasi pada permintaan bind sebelumnya.

3. *Anonymous*

Cocok dengan bukan pengguna yang terotentikasi.

4. *User*

   Koneksi pengguna yang telah terotentikasi.

5. *Regular Expression (regex)*

   Cocok dengan sebuah DN atau sebuah identitas SASL. Hal yang perlu diingat bahwa nama login yang digunakan untuk menentukan sebuah pengguna diambil dari DN (misalnya: *dn="cnuser1,ou=people,dc=kemas,dc=com"*) atau sebuah identitas SASL (misalnya: *dn="uid=user1,cn=gssapi,cn=auth"*). Table 4.4 merangkum bermacam-macam hak akses. Level yang lebih tinggi memiliki semua kemampuan dari level dibawahnya.

**Tabel 4.4 Level Akses**

| Level Access | Hak Akses |
|---|---|
| Write | Dapat mengubah nilai atribut. |
| Read | Dapat membaca hasil pencarian. |
| Search | Dapat melakukan pemfilteran suatu pencarian. |
| Compare | Dapat membandingkan atribut. |
| Auth | Dapat melakukan bind (otentikasi). |
| None | Tidak mendapatkan akses. |

Gambar 4.9 menunjukkan konfigurasi ACL pada server LDAP. Pada gambar dapat dilihat bahwa *uid=ldapadm* yang merupakan *user principal* Kerberos diberi hak *write* keseluruh entri direktori LDAP.

```
sasl-realm      AUTH.LANGHUA
sasl-host       auth.langhua

access to attrs=userPassword
        by self write
        by dn="uid=test,cn=auth.langhua,cn=gssapi,cn=auth" write
        by anonymous auth
        by * none

access to *
        by dn="uid=test,cn=auth.langhua,cn=gssapi,cn=auth" write
        by self write
        by * read
```

**Gambar 4.9 Konfigurasi ACL**

### *4.1.3.6* Konfigurasi SASL pada *slapd.conf*

File *slapd.conf* memiliki tiga bagian opsi berkenaan dengan SASL.

| | |
|---|---|
| *Sasl-host* | *hostname* |
| *Sasl-realm* | *string* |
| *Sasl-secprops* | *properties* |

*Sasl-host* merupakan *fully qualified domain name* dari host yang digunakan untuk otentikasi SASL yatiu *auth.langhua. Sasl-realm* merupakan domain SASL yang digunakan untuk otentikasi yaitu AUTH.LANGHUA. Parameter ketiga yaitu, *sasl-secprops* mengijinkan penentuan kondisi berbeda terhadap property SASL. Nilai yang mungkin untuk parameter ini digambarkan pada tabel 4.5

**Tabel 4.5 Deskripsi Parameter sasl-secprops**

| Flag | Deskripsi |
|---|---|
| *None* | Pengaturan properti keamanan secara default (*noplain, noanonymous).* |
| *Noplain* | Mematikan mekanisme-mekanisme serangan pasif. |
| *Noactive* | Mematikan mekanisme-mekanisme penyerangan aktif. |
| *Noanonymous* | Mematikan mekanisme-mekanisme yang mendukung login secara |

| | |
|---|---|
| | anonym. |
| *Frowardsec* | Mekanisme-mekanisme yang melewatkan credential dari *client.* |
| *Passcred* | Mekanisme-mekanisme yang melewatkan *credential* dari *client.* |
| *Minssf=factor* | Menentukan kekuatan keamanan minimum. Nilai yang mungkin yaitu; 0 (tidak ada penjagaan), 1 (penjagaan integritas saja), 56 (membolehkan enkripsi DES), 112 (mengijinkan 3DES atau metode enkripsi string lainnya), dan 128 (mengijinkan RC4, Bowlfish, atau algoritma enkripsi dari kelas ini). |
| *Maxssf=factor* | Menentukan pengaturan keamanan maximum. Nilai yang mungkin identik seperti pada minssf. |
| *Maxbufsize=size* | Menentukan ukuran maksimum dari *layer* keamanan untuk buffer. Nilai 0 mematikan *layer* keamanan. Nilai default yaitu nilai maksimum dari INT_MAX (sebagai contoh 65536). |

Pada parameter *sasl-secprop* ini, penentuan nilainya boleh menggunakan banyak kombinasi nilai. Properti default adalah *noanonymous* dan *noplain.*

Pemahaman menyeluruh mengenai parameter *sasl-secprop* juga diperlukan untuk hasil optimal untuk *plugin-plugin cyrus-sasl.* Tabel 4.6 berisi ringkasan mekanisme-mekanisme dan *flag* properti yang tersedia.

**Tabel 4.6 Properti Mekanisme Otentikasi SASL**

| SASL Mechanism | Security Property Flags | Maxssf |
|---|---|---|
| ANONYMOUS | NOPLAIN | 0 |
| CRAM-MD5 | NOPLAIN<br>NOANONYMOUS | 0 |

| | | |
|---|---|---|
| DIGEST-MD5 | NOPLAIN<br>NOANONYMOUS | 128 jika dikompilasi dengan RC; 112 jika dikompilasi bersama DES; 0 jika dikompilasi dengan RC4 nor DES lainnya |
| GSSAPI | NOPLAIN<br>NOACTIVE<br>NOANONYMOUS | 56 |
| KERBEROS-V4 | NOPLAIN<br>NOACTIVE<br>NOANONYMOUS | 56 |
| LOGIN | NOANONYMOUS | 0 |
| PLAIN | NOANONYMOUS | 0 |
| SCRAM-MD5 | NONE | 0 |
| SRP | NOPLAIN | 0 |

Penambahan baris-baris dibawah ini kebagian global dari file *slapd.conf* :

*Sasl-secprops noplain, noanonymous, minssf=56*

Apabila dibandingkan dengan nilai *sasl-secprops* dengan mekanisme pada tabel 4.6 menunjukkan bahwa server akan mengijinkan mekanisme-mekanisme dibawah ini untuk otentikasinya :

1. DIGEST-MD5
2. GSSAPI
3. KERBEROS_4

## 4.2 Hasil Penelitian dan Pembahasan

### 4.2.1 Pengujian Kerberos

Konfigurasi pada Kerberos perlu dilakukan apakah berhasil atau tidak. Pengujian dilakukan dengan perintah *kinit* yang akan memperoleh dan menyimpan sementara suatu inisial *Ticket Granting Ticket*. Untuk selanjutnya memasukkan *password* yang sesuai dengan *username*.

```
[root@kemas-laptop ~]# kinit -k host/auth.larghua
```

**Gambar 4.10 kinit AUTH.LANGHUA**

Dari gambar tersebut tampak bahwa *kinit* meminta TGT bagi ldapadm. Sebelumnya *user principal* ldapadm telah ditambahkan dalam database Kerberos. Apabila *password* yang dimasukkan sesuai maka *Ticket Granting Ticket* dapat diperoleh.

Perintah *kinit* akan menunjukkan daftar tiket-tiket Kerberos dan *principal* dalam *credential cache,* maupun key yang terapat dalam *keytab.*

```
[root@kemas-laptop ~]# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: host/auth.langhua@AUTH.LANGHUA

Valid starting     Expires            Service principal
08/14/10 12:47:05  08/15/10 12:47:05  krbtgt/AUTH.LANGHUA@AUTH.LANGHUA


Kerberos 4 ticket cache: /tmp/tkt0
klist: You have no tickets cached
[root@kemas-laptop ~]#
```

**Gambar 4.11 klist**

Dari proses pengujian tersebut dapat diketahui bahwa TGT telah diperoleh melaui *kinit.* Dengan demikian berarti Kerberos telah dapat dijalankan.

### 4.2.2 Pengujian SASL

#### 4.2.2.1 Penambahan LDAP Service

Penambahan *ldap service* diperluan agar server LDAP dapat menerima mekanisme GSSAPI yang ditawarkan. LDAP akan menjadi klien dari server SASL dimana mekanisme yang akan diberikan adalah GSSAPI.

*Addprinc –randkey ldap/auth.langhua@AUTH.LANGHUA*

Perintah di atas akan menambahka *service principal* yaitu LDAP. *Service principal* LDAP yang telah terbentuk harus ditambahkan dalam *keytab* milik SASL yaitu krb5.keytab yang terbentuk pada /etc/krb5.keytab.

*Ktadd –k /etc/krb5.keytab ldap/auth.langhua@AUTH.LANGHUA*

#### *4.2.2.2* Penambahan *User Principal test*

*User principal* test akan digunakan sebagai user dalam pengujian SASL GSSAPL. Penambahan *user principal* test akan dilakukan dengan perintah addprinc test@AUTH.LANGHUA. Langkah selanjutnya adalah menambahkannya ke dalam *keytab* dengan perintah ktadd –k /etc/krb5.keytab test.

Gambar berikut ini memperlihatkan proses penambahan *user principal test.*

```
[root@kemas-laptop ~]# kadmin.local
Authenticating as principal host/admin@AUTH.LANGHUA with password.
kadmin.local:  addprinc test@AUTH.LANGHUA
WARNING: no policy specified for test@AUTH.LANGHUA; defaulting to no policy
Enter password for principal "test@AUTH.LANGHUA":
Re-enter password for principal "test@AUTH.LANGHUA":
add_principal: Principal or policy already exists while creating "test@AUTH.LANGHUA".
kadmin.local:  addprinc test3@AUTH.LANGHUA
WARNING: no policy specified for test3@AUTH.LANGHUA; defaulting to no policy
Enter password for principal "test3@AUTH.LANGHUA":
Re-enter password for principal "test3@AUTH.LANGHUA"
Principal "test3@AUTH.LANGHUA" created.
kadmin.local:  ktadd -k /etc/krb5.keytab test3
Entry for principal test3 with kvno 2, encryption type Triple DES cbc mode with HMAC/sha
1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal test3 with kvno 2, encryption type DES cbc mode with CRC-32 added to
 keytab WRFILE:/etc/krb5.keytab.
kadmin.local:
```

**Gambar 4.12 Penambahan *user principal* test**

Langkah berikutnya adalah memperoleh tiket untuk *principal test* dengan *kinit-k test.*

### 4.2.2.3 Pengujian GSSAPI

Pengujian GSSAPI diperlukan untuk mengetahui apakah mekanisme SASL GSSAPI telah berhasil dikonfigurasikan. Pengujian menggunakan program *sample – server* dan *sample-client.*

Dari salah satu terminal perintah *sasl2-sample-server –s ldap –m GSSAPI.* Secara garis besar –s pada perintah tersebut menunjukkan *service* atau layanan yaitu LDAP. Kemudian mengidentifikasi mekanisme otentikasi yang akan digunakan yaitu GSSAPI, sehingga maksud dari perintah di atas adalah pengujian server SASL terhadap klien dalam hal ini adalah layanan LDAP apakah mekanisme otentikasi SASL GSSAPI yang ditawarkan berhasil dikonfigurasikan dengan baik atau tidak.

Gambar berikut ini menggambarkan tentang pengujian *sample-server* dari SASL GSSAPI yang dilakukan.

```
[root@kemas-laptop ~]# sasl2-sample-server -s ldap -m GSSAPI
trying 2, 1, 6
trying 10, 1, 6
bind: Address already in use
```

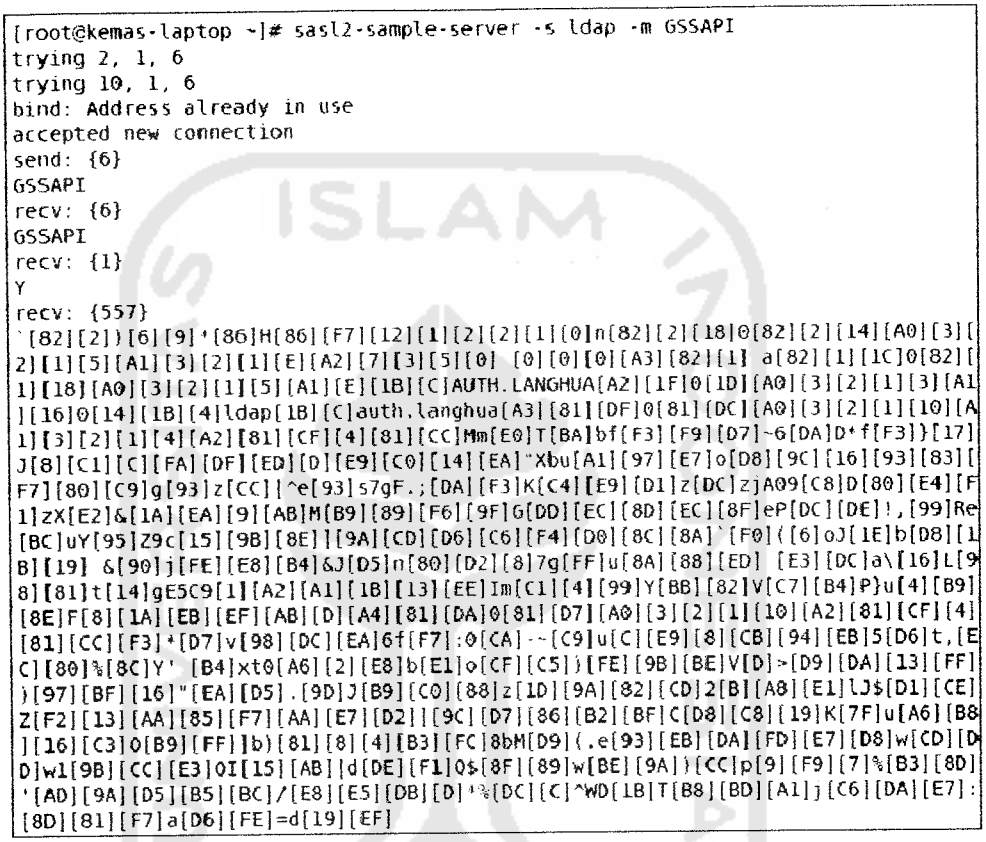
**Gambar 4.13 Pengujian *sample-server* SASL GSSAPI**

Dari terminal yang lain digunakan perintah *sasl2-sample-client –s ldap –m GSSAPI AUTH.LANGHUA* yang akan berkomunikasi dengan *sample-server* yang telah dijalankan sebelumnya. AUTH.LANGHUA dimasukkan dalam perintah diatas sebagai petunjuk *host* yang akan menjalankan *service*. *Userid* test dimasukkan ketika permintaan *userid* untuk otorisasi. Tentunya setelah user test ketika telah memperoleh TGT saat *kinit* berhasil dijalankan.

```
[root@kemas-laptop ~]# sasl2-sample-client -s ldap -m GSSAPI auth.langhua
receiving capability list... recv: {6}
GSSAPI
GSSAPI
please enter an authorization id: test
send: {6}
GSSAPI
send: {1}
Y
send: {557}
`[82][2])[6][9]'[86]H[86][F7][12][L][2][2][1][0]n[82][2][18]0[82][2][14][A0][3][
2][1][5][A1][3][2][1][E][A2][7][3][5][0] [0][0][0][A3][82][1] a[82][1][1C]0[82][
1][18][A0][3][2][1][5][A1][E][1B][C]AUTH.LANGHUA[A2][1F]0[1D][A0][3][2][1][3][A1
][16]0[14][1B][4]ldap[1B][C]auth.langhua[A3][81][DF]0[81][DC][A0][3][2][1][10][A
1][3][2][1][4][A2][81][CF][4][81][CC]Mm[E0]T[BA]bf[F3][F9][D7]~6[DA]D'f[F3])[17]
J[8][C1][C][FA][DF][ED][D][E9][C0][14][EA]"Xbu[A1][97][E7]o[D8][9C][16][93][83][
F7][80][C9]g[93]z[CC]|^e[93]s7gF.;[DA][F3]K[C4][E9][D1]z[DC]zjA09[C8]D[80][E4][F
1]zX[E2]&[1A][EA][9][AB]M[B9][89][F6][9F]G[DD][EC][8D][EC][8F]eP[DC][DE]!,[99]Re
[BC]uY[95]Z9c[15][9B][8E]][9A][CD][D6][C6][F4][D0][8C][8A]`[F0]([6]oJ[1E]b[D8][1
B][19] &[90]][FE][E8][B4]&J[D5]n[80][D2][8]7g[FF]u[8A][88][ED] [E3][DC]a\[16]L[9
8][81]t[14]gE5C9[1][A2][A1][1B][13][EE]Im[C1][4][99]Y[BB][82]V[C7][B4]P}u[4][B9]
[8E]F[8][1A][EB][EF][AB][D][A4][81][DA]0[81][D7][A0][3][2][1][10][A2][81][CF][4]
[81][CC][F3]'[D7]v[98][DC][EA]6f[F7]:0[CA]~[C9]u[C][E9][8][CB][94][EB]5[D6]t,[E
C][80]%[8C]Y' [B4]xt0[A6][2][E8]b[E1]o[CF][C5])[FE][9B][BE]V[D]>[D9][DA][13][FF]
)[97][BF][16]"[EA][D5].[9D]J[B9][C0][88]z[1D][9A][82][CD]2[B][A8][E1]lJ$[D1][CE]
Z[F2][13][AA][85][F7][AA][E7][D2]][9C][D7][86][B2][BF]C[D8][C8][19]K[7F]u[A6][B8
][16][C3]0[B9][FF]]b)[81][8][4][B3][FC]8bM[D9](.e[93][EB][DA][FD][E7][D8]w[CD][D
D]w1[9B][CC][E3]0I[15][AB]|d[DE][F1]Q$[8F][89]w[BE][9A])[CC]p[9][F9][7]%[B3][8D]
'[AD][9A][D5][B5][BC]/[E8][E5][DB][D]'%[DC][C]^WD[1B]T[B8][BD][A1])[C6][DA][E7]:
[8D][81][F7]a[D6][FE]=d[19][EF]
```

**Gambar 4.14 Pengujian SASL GSSAPI *sample-client***

Gambar 4.14 menggambarkan komunikasi antara *sample-server* dan *sample-client* yang ada disisi *client*. Negosiasi antara server dan *client* terjalin sehingga otentikasi berhasil ataukah sebaliknya.




**Gambar 4.15 Pengujian SASL GSSAPI *sample-server***

Gambar diatas menunjukkan komunikasi pengujian dari sisi server. Jika telah diperoleh keterangan *succesfull authentication* di akhir komunikasi antara server dan *client*, maka konfigurasi SASL GSSAPI telah berhasil dan didukung oleh LDAP.

### 4.2.3 Pengujian LDAP

Salah satu operasi *client* LDAP adalah *ldapsearch* yang digunakan dengan tujuan mencari entri-entri dalam direktori LDAP. Gambar dibawah ini menunjukkan operasi LDAP.

Dari gambar tersebut dapat diketahui bahwa mekanisme otentikasi yang digunakan adalah SASL GSSAPI dengan *username* . *Username* merupakan *user principal* Kerberos yang telah memperoleh TGTsebelumnya. Hasil operasi *ldapsearch* diatas dapat dilihat lebih lengka dalam lampiran. Apabila server LDAP tidak mendukung mekanisme SASL GSSAPI, maka otentikasi akan gagal.

```
[kemas@kemas-laptop ~]$ ldapsearch -x -b 'dc=auth,dc=langgua'
# extended LDIF
#
# LDAPv3
# base <dc=auth,dc=langgua> with scope subtree
# filter: (objectClass=*)
# requesting: ALL
#

# auth.langgua
dn: dc=auth,dc=langgua
objectClass: dcObject
objectClass: organization
o: Home LDAP Server
dc: auth

# Manager, auth.langgua
dn: cn=Manager,dc=auth,dc=langgua
objectClass: organizationalRole
cn: Manager

# users, auth.langgua
dn: ou=users,dc=auth,dc=langgua
ou: users
objectClass: top
objectClass: organizationalUnit

# addressbook, auth.langgua
dn: ou=addressbook,dc=auth,dc=langgua
ou: addressbook
objectClass: top
objectClass: organizationalUnit

# search result
search: 2
result: 0 Success
```

Gambar 4.16 Operasi *ldapsearch*

### 4.2.4 Keamanan Pengaksesan Direktori

Status akses ke direktori dapat diketahui melalui perintah *ldapwboami.* Otentikasi yang dilakukan dengan mekanisme simple bind menggunakan entri-entri yang ada dalam server LDAP sebagai usernya.

Operasi *ldapwboami* akan dijalankan oleh *user1* yang merupakan salah satu *entry* dalam direktori LDAP. Sebelumnya *user1* harus mengotentikasi dirinya menggunakan mekanisme *simple bind –W* akan meminta *user1* memasukkan *password.*

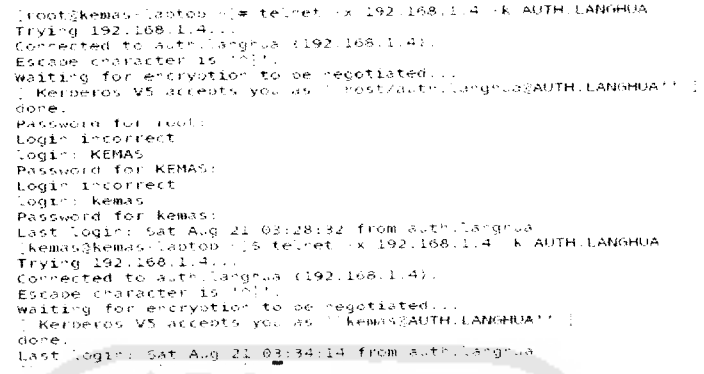
### 4.2.5 Perbandingan Sistem Otentikasi LDAP Menggunakan Kerberos dan Tanpa Kerberos

Pengujian dan perbandingan dilakukan pada sistem otentikasi server LDAP. Sistem yang pertama kali diuji adalah sistem otentikasi untuk mengakses server LDAP tanpa menggunakan Kerberos yaitu menggunakan *simple bind*, sedangkan sistem yang selanjutnya diuji adalah sistem otentikasi LDAP menggunakan otentikasi yang ditawarkan Kerberos dengan modul SASL GSSAPI. Kedua sistem diuji pada lingkungan pengujian yang sama. Tabel 4.7 memperlihatkan hasil pengujian dan perbandingan diantara kedua sistem.

**Tabel 4.7 Perbandingan Sistem Otentikasi LDAP dengan Kerberos dan tanpa Kerberos**

| Keterangan | Otentikasi LDAP Tanpa Kerberos | Otentikasi LDAP Menggunakan Kerbros |
|---|---|---|
| Mekanisme otentikasi ke direktori LDAP | *Simple bind* | SASL, GSSAPI sebagai modul otentikasi tambahan. |
| Keamanan akses ke direktori LDAP | Kurang aman karena *password* masih dilewatkan ke dalam jaringan. | Lebih aman karena *password* tidak ditransmisikan dalam jaringan |

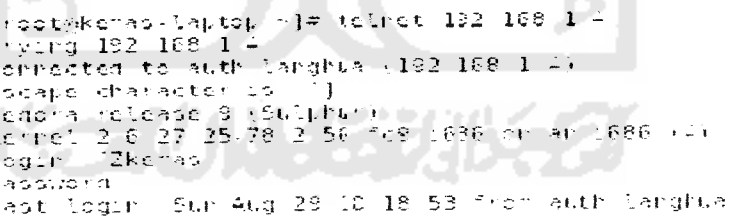
Berikut merupakan hasil pengujian keamanan akses ke direktori LDAP, yaitu dengan menggunakan perintah telnet. Bagian pertama dilakukan operasi telnet pada direktori LDAP yang menggunakan Kerberos.




**Gambar 4.17 Operasi telnet menggunakan Kerberos**

Gambar 4.18 menerangkan pengujian server LDAP yang menggunakan Kerberos. Operasi yang digunakan untuk pengujian adalah *telnet –x 192.168.1.4 –k AUTH.LANGHUA*, yaitu operasi untuk masuk ke computer lain dalam suatu jaringan. Lebih rinci operasi diatas adalah telnet meminta untuk masuk ke dalam jaringan IP 192.168.1.4 sebagai server untuk mengakses direktori dari auth.langhua. Hasil pengujian diatas memperlihatkan bahwa telnet tidak mendapatkan informasi apapun dari direktori auth.langhua yang memakai *protocol Kerberos* sebagai pengamannya.

Berikut merupakan hasil operasi telnet pada direktori LDAP yang tidak menggunakan Kerberos.




**Gambar 4.18 Operasi telnet tanpa menggunakan Kerberos**

Gambar 4.19 menerangkan pengujian server LDAP yang tidak menggunakan Kerberos. Operasi yang digunakan untuk pengujian adalah *telnet 192.168.1.4*. Telnet meminta untuk masuk ke jaringan IP 192.168.1.4 sebagai server LDAP. Hasil

pengujian diatas memperlihatkan bahwa telnet mendapatkan informasi system yang digunakan server LDAP.

Dari hasil pengujian diatas dapat disimpulkan bahwa operasi telnet pada direktori LDAP yang menggunakan Kerberos lebih aman karena enkripsi data berjalan dalam sistem server dibandingkan dengan LDAP yang tidak menggunkan Kerberos.

Tabel 4.7 juga menunjukkan bahwa otentikasi yang ditawarkan Kerberos meningkatkan keamanan pengaksesan direktori LDAP walaupun disisi lain kecepatan akses ke direktori LDAP yang menggunakan Kerberos relative lebih lama dibandingkan dengan sistem yang tanpa Kerberos.

### 4.2.6 Perbandingan Kerberos dengan Protokol Lain

Tabel dibawah ini merupakan perbandingan antara protokol Kerberos dengan protokol Radius, beberapa data dibawah ini didapat dari referensi yang dirujuk dari beberapa sumber diantaranya www.ilkom.unsri.ac.id.

**Tabel 4.8 Perbandingan Kerberos dengan Radius**

| Radius | Kerberos |
|---|---|
| Sistem bekerja lambat, karena Radius bekerja dengan 3 konsepnya. Pertama otentikasi, memastikan apakah *client* terdaftar dalam jaringan. Kedua otorisasi, mengetahuai hak akses pada *client*. Ketiga akunting, merupakan pendaftaran *account* apakah *client* sah atau tidak di Radius. | Kerberos menawarkan satu kenyamanan pada user, hanya dengan memasukkan password sekali dan meminta ticket (TGT) pada TGS, maka kita bisa meminta layanan pada banyak akun yang kita inginkan. Selain itu, ticket tersebut berlaku dalam periode waktu yang pendek sampai masa expirednya |
| Skema proteksi yang dipakai adalah *stream-chiper*. Hal ini memungkinkan penyusup mendapatkan informasi *shared secret* apabila mereka melakukan sniffing kejaringan *wireless* dan mencoba masuk ke *Radius server*. | Tingkat keamanannya tinggi. *Password* tidak dikirimkan melintasi jaringan. |
| Tidak adanya autentikasi dan verifikasi terhadap *access request*. | Kerberos bersifat *transparent*. *User* tidak perlu mengetahui tahap-tahap otentikasi yang dilakukan di dalam jaringan. *User* hanya tinggal*login* ke jaringan melalui program inisialisasi kinit, memasukkan*username* dan *password*, lalu *user* memperoleh otentikasi ke *server* yang dituju. |
| Radius tidak menyediakan layanan SSH. | Kerberos menyediakan layanan SSH yang dapat mengelola server jarak jauh. |