

Configuration de FreeRADIUS avec MySQL

Salif Biaye
Ndeye Astou Diagouraga
Mouhamadou Tidiane Seck
Ouleymatou Sadiya Cissé

Introduction

Ce rapport décrit la configuration de FreeRADIUS pour l'intégration avec une base de données MySQL, permettant ainsi une gestion centralisée des utilisateurs et des authentications.

Installation des paquets nécessaires

Commandes :

```
apt update
apt install freeradius freeradius-mysql mariadb-server
```

Rôle : Installer FreeRADIUS et le module MySQL requis pour l'intégration.

Configuration des clients dans FreeRADIUS

Commande :

```
nano /etc/freeradius/3.0/clients.conf
```

Rôle : Cette commande permet de définir les clients (routeurs, points d'accès Wi-Fi, etc.) autorisés à communiquer avec le serveur RADIUS.

```
#clients per_socket_clients {
    client routeur {
        ipaddr = 192.168.1.10
        secret = passer
        nastype = cisco
    }

    client switch {
        ipaddr = 192.168.1.20
        secret = passer
        nastype = huawei
    }
    client wifi_ap {
        ipaddr = 192.168.1.30
        secret = passer
        nastype = alcatel
    }
}
```

Configuration des utilisateurs dans FreeRadius

```
nano /etc/freeradius/3.0/users
```



Rôle

Permet de définir manuellement les utilisateurs locaux dans FreeRADIUS avec leurs mots de passe et messages de réponse.

```
#####
# You should add test accounts to the TOP of this file! #
# See the example user "bob" above.                      #
#####

testaccount Cleartext-Password := "Test123"

"Salif"    Cleartext-password := "passer"
           Reply-Message = "Bonjour , %{User-Name} bien connecte"

"Astou"    Cleartext-password := "passer"
           Reply-Message = "Bonjour , %{User-Name} bien connecte"

"Ouley"    Cleartext-password := "passer"
           Reply-Message = "Bonjour , %{User-Name} bien connecte"
```

NB: On a créé des utilisateurs avec chacun un mot de passe

Redémarrer le service Freeradius

```
service freeradius restart
```

Vérifier le status du service Freeradius

```
service freeradius status


● freeradius.service – FreeRADIUS multi-protocol policy server
   Loaded: loaded (/lib/systemd/system/freeradius.service; disabled; vendor preset: enabled)
   Active: active (running) since Thu 2025-01-02 21:14:10 GMT; 2h 56min ago
     Docs: man:radiusd(8)
           man:radiusd.conf(5)
           http://wiki.freeradius.org/
           http://networkradius.com/doc/
   Process: 21852 ExecStartPre=/usr/sbin/freeradius $FREERADIUS_OPTIONS -Cxm -lstdout (code=exited, status=0/SUCCESS)
  Main PID: 21854 (freeradius)
    Status: "Processing requests"
     Tasks: 6 (limit: 4584)
    Memory: 77.4M
   CGroup: /system.slice/freeradius.service
           └─21854 /usr/sbin/freeradius -f

san 02 21:14:10 server.dic.sn freeradius[21852]: Please use tls_min_version and tls_max_version instead of disable_tlsv1_2
san 02 21:14:10 server.dic.sn freeradius[21852]: tls: Using cached TLS configuration from previous invocation
san 02 21:14:10 server.dic.sn freeradius[21852]: tls: Using cached TLS configuration from previous invocation
san 02 21:14:10 server.dic.sn freeradius[21852]: rlm_mschap (mschap): using internal authentication
san 02 21:14:10 server.dic.sn freeradius[21852]: Ignoring "sql" (see raddb/mods-available/README.rst)
san 02 21:14:10 server.dic.sn freeradius[21852]: Ignoring "ldap" (see raddb/mods-available/README.rst)
san 02 21:14:10 server.dic.sn freeradius[21852]: # Skipping contents of 'if' as it is always 'false' --- /etc/freeradius/3.0/sites-enabled/in>
san 02 21:14:10 server.dic.sn freeradius[21852]: radiusd: ##### Skipping IP addresses and Ports #####
san 02 21:14:10 server.dic.sn freeradius[21852]: Configuration appears to be OK
san 02 21:14:10 server.dic.sn systemd[1]: Started FreeRADIUS multi-protocol policy server
```

Tester un utilisateur créé

```
echo "User-Name = testaccount , User-Password = Test123" | radclient localhost:1812 auth testing123

Sent Access-Request Id 42 from 0.0.0.0:46581 to 127.0.0.1:1812 length 69
Received Access-Accept Id 42 from 127.0.0.1:1812 to 127.0.0.1:46581 length 38
```

 **Role**

Teste un utilisateur en envoyant une requête au serveur RADIUS sur le port 1812. Si tout fonctionne, un message Access-Accept est renvoyé.

Tester avec un autre utilisateur

```
radtest Salif passer 127.0.0.1 1812 testing123
Sent Access-Request Id 212 from 0.0.0.0:34171 to 127.0.0.1:1812 length 75
  User-Name = "Salif"
  User-Password = "passer"
  NAS-IP-Address = 192.168.1.27
  NAS-Port = 1812
  Cleartext-Password = "passer"
Received Access-Accept Id 212 from 127.0.0.1:1812 to 127.0.0.1:34171 length 69
  Message-Authenticator = 0x1e54a0bbca4e3addf8ffe3121ac84f01
  Reply-Message = "Bonjour , Salif bien connecte"
```


Configuration de FreeRADIUS pour utiliser MySQL

Création de la base de données RADIUS

Initialisation de la base de données

Commandes :

```
mysql -u root -p
CREATE DATABASE radius;
CREATE USER 'radius'@'localhost' IDENTIFIED BY 'passer';
GRANT ALL PRIVILEGES ON radius.* TO 'radius'@'localhost';
FLUSH PRIVILEGES;
```

 **Role**

Crée une base de données nommée radius et un utilisateur associé avec tous les droits

Importation de la structure de la base de données

Commande :

```
mysql -u radius -p radius < /etc/freeradius/3.0/mods-config/sql/main/mysql/schema.sql
```

[↗ Role](#)

Importer la structure de la base de données nécessaire (tables comme radcheck , radreply , etc.).

Édition du module SQL

Commande :

```
nano /etc/freeradius/3.0/mods-available/sql
```

[↗ Role](#)

Configurer les paramètres MySQL dans FreeRADIUS.

configuration :

```
dialect = "mysql"
server = "localhost"
port = 3306
login = "radius"
password = "passer"
radius_db = "radius"
```

Activation du module SQL :

```
ln -s /etc/freeradius/3.0/mods-available/sql /etc/freeradius/3.0/mods-enabled/
```

[↗ Role](#)

Active le module SQL en créant un lien symbolique vers le dossier mods-enabled

Modification des sites activés

Commande :

```
nano /etc/freeradius/3.0/sites-enabled/default
```

[↗ Role](#)

Vérifie que les sections authorize, accounting et session contiennent l'option sql ou -sql, sinon ajouter le. Cela assure que FreeRADIUS interagira avec la base de données pour gérer les utilisateurs

Exemple :

```
authorize {
    sql
}

accounting {
    sql
}

session {
    sql
}
```

Ajout d'utilisateurs dans la base de données

Commande :

```
mysql -u radius -p radius
```

Requête SQL :

```
INSERT INTO radcheck (username, attribute, op, value) VALUES ('jdoe', 'Cleartext-Password', ':=', 'passer');
```

Rôle : Ajouter un utilisateur à la table `radcheck` pour l'authentification.

Tester la configuration

Validation de la configuration

Commande :

```
freeradius -CX
```

Rôle : S'assurer que les fichiers de configuration sont valides.

Test d'un utilisateur

Commande :

```
radtest jdoe passer 127.0.0.1 1812 testing123

ent Access-Request Id 212 from 0.0.0.0:34171 to 127.0.0.1:1812 length 75
  User-Name = "joe"
  User-Password = "passer"
  NAS-IP-Address = 192.168.1.27
  NAS-Port = 1812
  Cleartext-Password = "passer"
Received Access-Accept Id 212 from 127.0.0.1:1812 to 127.0.0.1:34171 length 69
  Message-Authenticator = 0x1e54a0bbca4e3addf8ffe3121ac84f01
```



Role

Tester l'authentification d'un utilisateur via le serveur MySQL. Un résultat `Access-Accept` indique que tout fonctionne correctement.

Authentification des utilisateurs

Une fois la configuration terminée et validée, les utilisateurs définis dans MySQL peuvent être authentifiés automatiquement via FreeRADIUS.