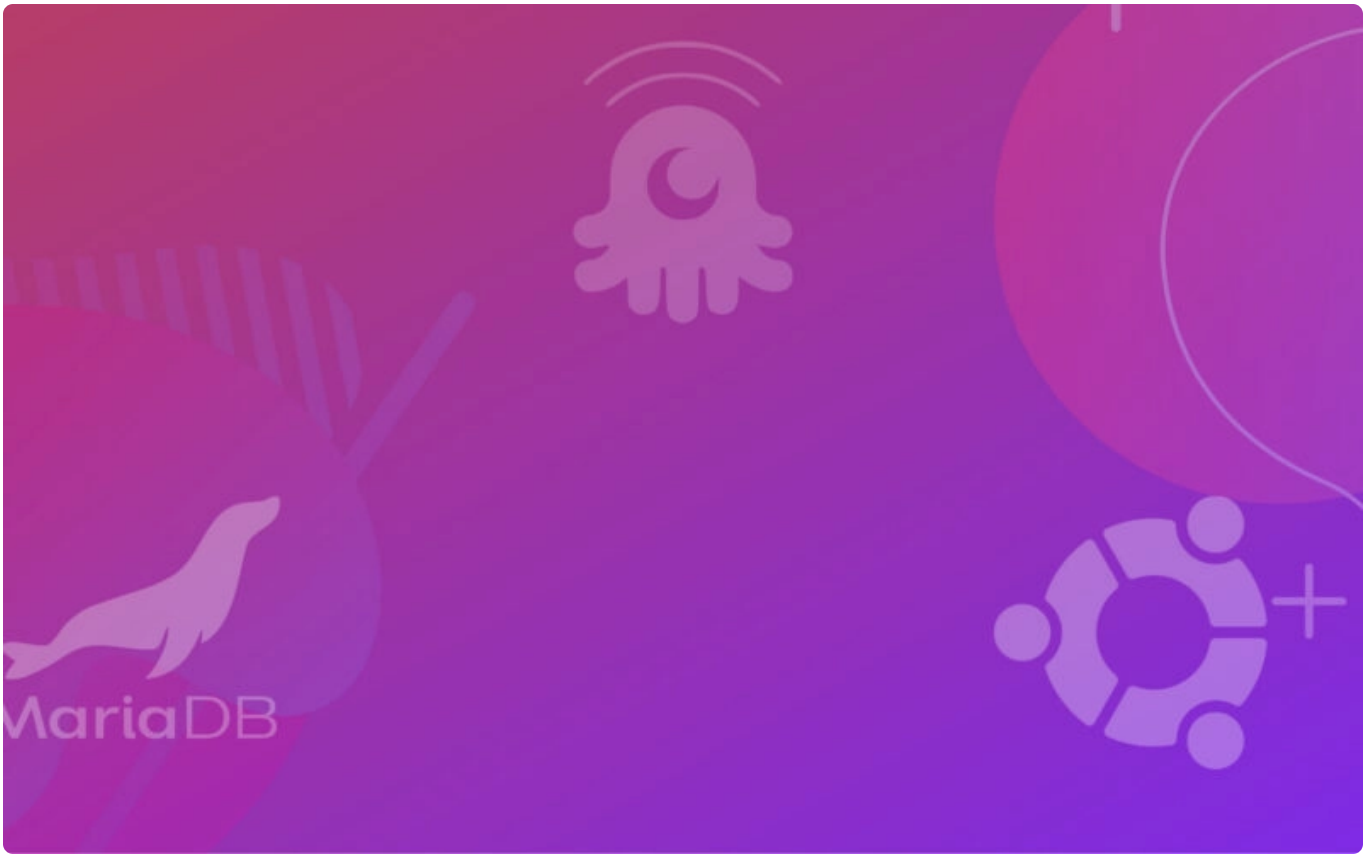
U — UBUNTU

Install FreeRADIUS & daloRADIUS on Ubuntu 20.04 + MySQL/MariaDB

by **EdXD** and **George H.** · March 8, 2021 · 18 shares · 23 comments · 16 minute read



Not using Ubuntu 20.04? Choose a different version or distro.

FreeRADIUS is a free and open-source implementation of the RADIUS protocol. It's the most popular and widely deployed open-source RADIUS server, being also used by many Fortune-500 companies, telecommunications companies, and Tier 1 ISPs.

FreeRADIUS most often refers to the RADIUS server, which is just one component of the FreeRADIUS suite. Other components included are:

- BSD-licensed RADIUS client library
- a PAM library
- an Apache RADIUS module
- additional utilities and development libraries related to RADIUS

> **What is the RADIUS Protocol**
> RADIUS (Remote Authentication Dial-In User Service) is a networking protocol that provides centralized Authentication, Authorization and Accounting (**AAA**) management for users who connect and network service. Here are some short descriptions of what each of the terms in **AAA** mean:
>
> - **Authentication:** the process of determining whether the client (which can be a user, device, or process) is a legitimate user of the system
> - **Authorization:** the process of determining what the client is allowed to do on the network
> - **Accounting:** is the process of monitoring the client's activity on the network, and providing the information necessary to bill for services
>
> **What are some real world examples for using the RADIUS protocol?**
> You can use RADIUS in situations when you have users that need to connect to your network and you need a solution to manage them – they need to **authenticate**, you give them various **authorizations** as to what they can do when connected to your network, and you keep track of how they use your network (**accounting**).
>
> A few rudimentary examples of real world services that use RADIUS are:
>
> - **Universities**: some may offer free WiFi access for students and staff, so they'll offer credentials to students and staff and only allow users with valid credentials to authenticate and use the University WiFi
> - **VPN providers**: using RADIUS you can offer authentication for your users, authorize them to use your service while they haven't exhausted their allocated bandwidth and paid their subscription, and keep track of their network usage

In this tutorial we'll install FreeRADIUS on a server running Ubuntu 20.04 and configure it to work with MySQL/MariaDB; we'll also install **daloRADIUS**, a RADIUS web management panel, which is basically a FreeRADIUS GUI, and then perform a simple test on the RADIUS server to make sure it works.

TABLE OF CONTENTS

Prerequisites

- A server running Ubuntu 20.04, and we recommend a minimum of 512RAM and 300MB storage space.
- Being logged in as a **non-root sudo user**. This is because when you're acting as root, you can do anything and the system won't ask. If you're not careful you can harm your system, and if you run malicious/buggy applications with root permissions, the application can harm your system. There is good reason why this has been the security model for years.

Assuming you're on a fresh server running Ubuntu 20.04 install, first we'll update the server's package index and upgrade to the latest packages:

```
$ sudo apt update
$ sudo apt upgrade
```

Install LAMP Stack

The LAMP stack is a group of open-source softwares that can be used to create web applications and websites. LAMP is an acronym for **L**inux, **A**pache, **M**ySQL, **P**HP.

Install Apache Web Server

```
$ sudo apt -y install apache2
```

Enable Apache so it starts on boot:

```
$ sudo systemctl enable --now apache2
```

Include Apache's application profile in UFW's rules:

```
$ sudo ufw allow WWW
```

> **Sidenote**: If that does not work then the web server profile in UFW is probably called something else than `WWW`. You can also try `sudo ufw allow Apache` instead.

Check access to Apache by visiting the server's IP or hostname (http://ip_address)

You should see something like this in your browser:

Install PHP & Additional PHP Modules

```
$ sudo apt -y install php libapache2-mod-php
php-{gd,common,mail,mail-mime,mysql,pear,db,mbstring,xml,curl}
```

Check PHP version:

```
$ php -v
```

Check if PHP is working

A quick way you can make sure that PHP works, is by creating a simple PHP file in the Apache document root as follows.

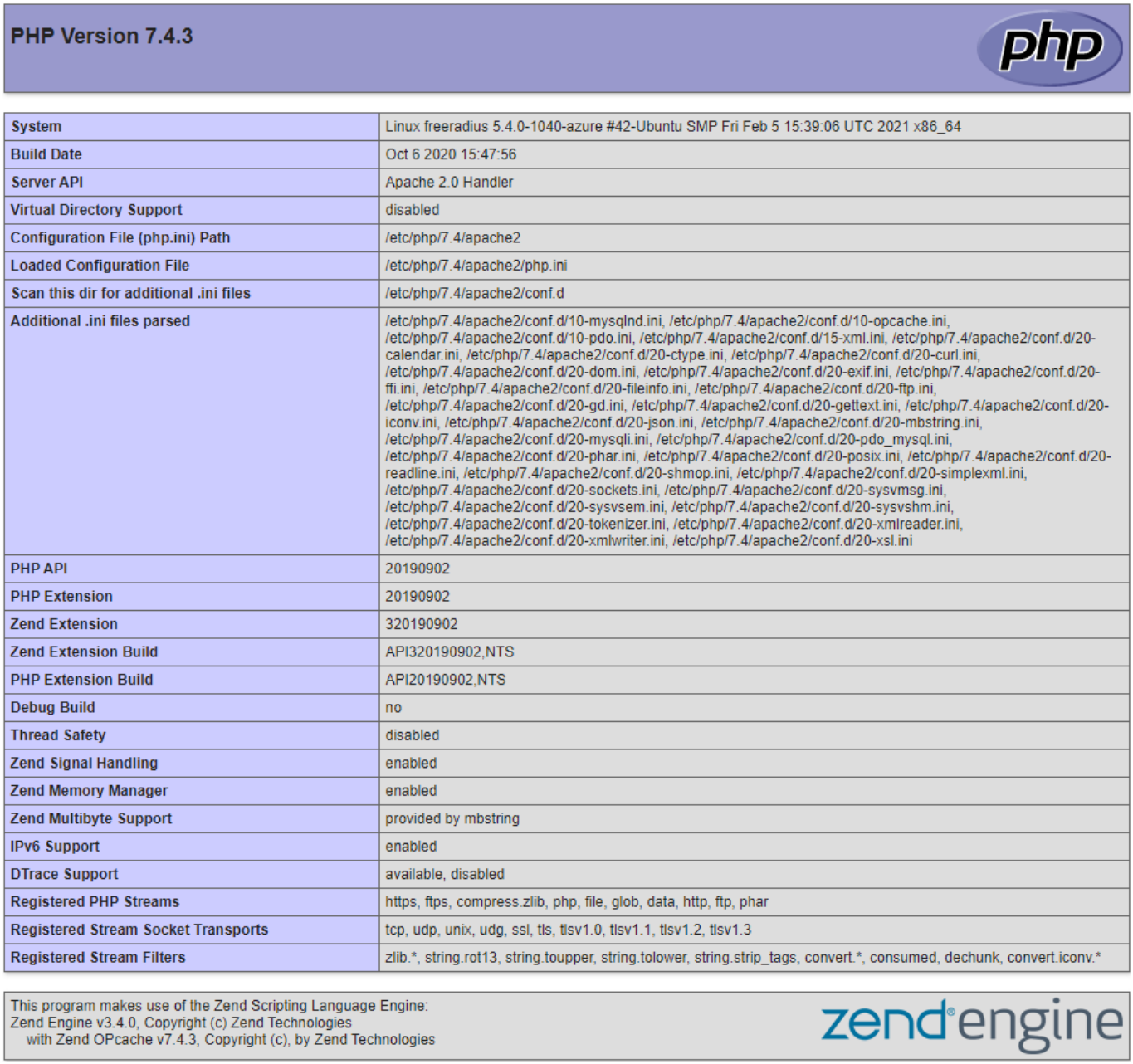
Create a file in **/var/www/html** called **phpinfo.php** (it can be any name, it doesn't matter)

```
$ sudo nano /var/www/html/phpinfo.php
```

And add the following line:

Save and close the file.

Now you can visit https://your_server_ip/phpinfo.php and you'll see something like:



PHP Version 7.4.3

System	Linux freeradius 5.4.0-1040-azure #42-Ubuntu SMP Fri Feb 5 15:39:06 UTC 2021 x86_64
Build Date	Oct 6 2020 15:47:56
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-curl.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gd.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-mysqli.ini, /etc/php/7.4/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-simplexml.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini, /etc/php/7.4/apache2/conf.d/20-xmlreader.ini, /etc/php/7.4/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.4/apache2/conf.d/20-xsl.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v3.4.0, Copyright (c) Zend Technologies
with Zend OPcache v7.4.3, Copyright (c), by Zend Technologies

Now remove the file, since it can be used by malicious entities to see information about your server:

```
$ sudo rm /var/www/html/phpinfo.php
```

Install MySQL or MariaDB

MariaDB has been a drop-in replacement for MySQL for years, however there are some

differences. From my experience, both work for our purposes, but you still might want to check the differences to make an informed decision. This is an easily skimmable article on the topic – MariaDB vs MySQL: Key Performance Differences .

I typically go with MariaDB, but you can choose whichever you prefer.

MySQL

```
$ sudo apt -y install mysql-server
```

MariaDB

```
$ sudo apt -y install mariadb-server
```

MySQL/MariaDB comes with a script to set up your password to MySQL/MariaDB, as well as altering some less secure values. To start it run the following command:

```
$ sudo mysql_secure_installation
```

You'll be asked for the current root MySQL password for root:

```
Enter current password for root (enter for none):
```

If you followed this tutorial you haven't set it yet, so go ahead and press **Enter**. You'll be asked if you want to set a root password – press **Y** and **Enter** and set a new root MySQL password.

Validate Password Plugin

You can skip this section if you're not prompted by the VALIDATE PASSWORD PLUGIN.

If you install MySQL (and not MariaDB), when you run **mysql_secure_installation** you may be asked if you want your password validated to make sure it's strong.

This plugin will ask you to select from 3 levels of password strength to validate from, and depending on what you select your password will be graded and shown to you so you can decide if you want to continue with it or try entering a different one.

```
VALIDATE PASSWORD PLUGIN can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD
plugin?

Press y|Y for Yes, any other key for No:
```

I typically select **Y** but you can select **No** if you're sure of your password.

If you select **Y** then you'll be asked to select how strong your password should be.

```
There are three levels of password validation policy:

LOW Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special
characters
STRONG Length >= 8, numeric, mixed case, special characters
and dictionary file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 2
```

In the example I selected **2**.

My password contains lowercase letters, uppercase letters, numbers and symbols, and it's over 8 characters. I get password strength 100 and decide I want to continue so I input **Y**.

```
Estimated strength of the password: 100
Do you wish to continue with the password provided?(Press y|Y
for Yes, any other key for No) : Y
```

Next you'll be asked if you want to remove anonymous users, restrict remote root user access to the local machine, remove test databases, and reload tables. Answer **y/leave empty** and press enter for **Yes** to each – unless you have a good reason not to.

Remove anonymous users:

```
Remove anonymous users? [Y/n] y
... Success!
```

Disallow root login remotely:

```
Disallow root login remotely? [Y/n] y
... Success!
```

Remove the test database:

```
Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!
```

Reload privilege tables:

```
Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...
```

Now MySQL (or MariaDB) has been installed on your system and we can proceed with configuring FreeRADIUS to use it.

Install FreeRADIUS and Configure with MySQL/MariaDB on Ubuntu 20.04

Install FreeRADIUS along with two modules that FreeRADIUS will need:

- **freeradius-mysql** – MySQL module for FreeRADIUS, so the server can do accounting and authentication using MySQL.
- **freeradius-utils** – a module that adds additional useful features to the FreeRADIUS server

```
$ sudo apt -y install freeradius freeradius-mysql freeradius-utils -y
```

Test the FreeRADIUS Server

FreeRADIUS is expected to run well with the default configuration.

To quickly check that FreeRADIUS and up and running we'll run it in **debug mode**.

Stop the FreeRADIUS server, as it started automatically after installing it.

```
$ sudo systemctl stop freeradius
```

Run FreeARDIUS in debug mode (remember to use sudo):

```
$ sudo freeradius -X
```

The output should look something like this:

```
Listening on auth address * port 1812 bound to server default
Listening on acct address * port 1813 bound to server default
Listening on auth address :: port 1812 bound to server
default
Listening on acct address :: port 1813 bound to server
default
Listening on auth address 127.0.0.1 port 18120 bound to
server inner-tunnel
Listening on proxy address * port 52868
Listening on proxy address :: port 57983
Ready to process requests
```

Stop debug mode by pressing Ctrl+C.

Start and enable FreeRADIUS service so it runs on system boot:

```
$ sudo systemctl enable --now freeradius
```

Allow FreeRADIUS in Firewall

(If you have UFW running on Ubuntu 20.04)

FreeRADIUS uses UDP ports 1812 for authentication and 1813 for accounting. You need to make sure those ports are allowed. The method by which you allow them can also depend on the platform you're using.

If you're using UFW, then you can open them by running:

```
$ sudo ufw allow to any port 1812 proto udp
$ sudo ufw allow to any port 1813 proto udp
```

Configure FreeRADIUS to use MySQL/MariaDB

We'll create a database and a database user for FreeRADIUS to use.

You can use any credentials you like but make sure to remember to replace the credentials that I'm using with your own, throughout the rest of the tutorial.

The details we'll use are:

Database: `radius`
User: `radius`
Password: `Somestrongpassword_321`

To begin, access the MySQL/MariaDB console as **root**, by running the following command and then inputting your password at the prompt:

```
$ sudo mysql -u root -p
```

Create a database and user that will be used by FreeRADIUS:

```
MariaDB [(none)]> CREATE DATABASE radius;
MariaDB [(none)]> GRANT ALL ON radius.* TO radius@localhost
IDENTIFIED BY "Somestrongpassword_321";
MariaDB [(none)]> FLUSH PRIVILEGES;
MariaDB [(none)]> quit;
```

Now to populate the database with the RADIUS MySQL schema.

First we'll have to switch to using the **root** user, otherwise we get **Access denied** when trying to import, even if we're using **sudo**:

```
$ sudo su -
```

Now import the RADIUS MySQL schema:

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

Let's switch back to our **non-root user** (I'm using **edxd** so I'll switch to that):

```
$ sudo su - edxd
```

You can check the tables just created in the **radius** database by running the following command, and then entering your **root MySQL/MariaDB password**:

```
$ sudo mysql -u root -p -e "use radius;show tables;"
```

Output:

```
+-------------------+
| Tables_in_radius  |
+-------------------+
| nas               |
| radacct           |
| radcheck          |
| radgroupcheck     |
| radgroupreply     |
| radpostauth       |
| radreply          |
| radusergroup      |
+-------------------+
```

Create a soft link to the SQL module to **/etc/freeradius/3.0/mods-enabled**:

```
$ sudo ln -s /etc/freeradius/3.0/mods-available/sql
```

```
/etc/freeradius/3.0/mods-enabled/
```

Next we configure FreeRADIUS to use SQL. To do this open **/etc/freeradius/3.0/mods-enabled/sql** using your favorite text editor, so we can edit some parameters.

I'll install and use **nano** as my text editor, and open the file:

```
$ sudo apt install nano
$ sudo nano /etc/freeradius/3.0/mods-enabled/sql
```

There's quite a bit of text, but most of it is commented out. We'll just need to edit a few things.

1. Change **dialect = "sqlite"** to **dialect = "mysql"**
2. Change **driver = "rlm_sql_null"** to **driver = "rlm_sql_${dialect}"**
3. If we use MySQL the FreeRADIUS configuration assumes the use of TLS certs by default. For the purpose of this tutorial we won't be using TLS certs, so we'll comment out the MySQL TLS section, by adding a **#** sign in at the beginning of every line in the **tls** section.The TLS section looks something like this:

```
mysql {
    # If any of the files below are set, TLS encryption
is enabled
    tls {
            ca_file = "/etc/ssl/certs/my_ca.crt"
            ca_path = "/etc/ssl/certs/"
            certificate_file = "/etc/ssl/certs/private
/client.crt"
            private_key_file = "/etc/ssl/certs/private
/client.key"
            cipher = "DHE-RSA-AES256-SHA:AES128-SHA"

            tls_required = yes
            tls_check_cert = no
            tls_check_cert_cn = no
    }

    # If yes, (or auto and libmysqlclient reports
warnings are
    # available), will retrieve and log additional
warnings from
```

```
    # the server if an error has occured. Defaults to
'auto'
    warnings = auto
}
```

And this is how it looks with the **tls** section commented out:

```
mysql {
    # If any of the files below are set, TLS encryption
is enabled
    # tls {
    #       ca_file = "/etc/ssl/certs/my_ca.crt"
    #       ca_path = "/etc/ssl/certs/"
    #       certificate_file = "/etc/ssl/certs/private
/client.crt"
    #       private_key_file = "/etc/ssl/certs/private
/client.key"
    #       cipher = "DHE-RSA-AES256-SHA:AES128-SHA"

    #       tls_required = yes
    #       tls_check_cert = no
    #       tls_check_cert_cn = no
    #}

    # If yes, (or auto and libmysqlclient reports
warnings are
    # available), will retrieve and log additional
warnings from
    # the server if an error has occured. Defaults to
'auto'
    warnings = auto
}
```

4. Next we'll uncomment the **Connection info** section and add in the connection details to our MySQL/MariaDB database.First uncomment (remove the **#** signs) from the beginning of the lines starting with **server**, **port**, **login**, **password**.
 `server` – this is the server where the database is located. In this case it's the local server so we can leave "localhost"
 `port` – is set to 3306, which is the default port for the classic MySQL protocol. Leave it as is, unless you changed the MySQL port.
 `login` – this is the database user you created earlier for FreeRADIUS to use. I created

the user **radius** so I'll leave it as is. You change it if your user is something else.
`password` – the password for that MySQL user that you also set earlier.

This is it's initial state:

```
# Connection info:
#
# server = "localhost"
# port = 3306
# login = "radius"
# password = "radpass"
```

And here it is edited.

```
# Connection info:
#
server = "localhost"
port = 3306
login = "radius"
password = "Somestrongpassword_321"
```

5. A few lines lower we need to configure the name of the database. By default it looks like this:

```
# Database table configuration for everything except
Oracle
radius_db = "radius"
```

Instead of **radius**, input the database you created. Since I created the database **radius** I'll leave it as is:

```
# Database table configuration for everything except
Oracle
radius_db = "radius"
```

6. Further down we'll uncomment a line containing **read_clients = yes**. This is to enable FreeRADIUS to read clients from the database.Here is how it looks:

```
# Set to 'yes' to read radius clients from the database
```

```
('nas' table)
# Clients will ONLY be read on server startup.
# read_clients = yes
```

And just remove the **#** sign to uncomment it:

```
# Set to 'yes' to read radius clients from the database
('nas' table)
# Clients will ONLY be read on server startup.
read_clients = yes
```

7. Just a few lines lower, we want **client_table = "nas"** to be uncommented. It should be uncommented by default, but just check to make sure it looks like this:

```
# Table to keep radius client info
client_table = "nas"
```

Now change the group rights of the file we just edited:

```
$ sudo chgrp -h freerad /etc/freeradius/3.0/mods-
available/sql
$ sudo chown -R freerad:freerad /etc/freeradius/3.0/mods-
enabled/sql
```

And restart the FreeRADIUS service:

```
$ sudo systemctl restart freeradius.service
```

Since we've done quite a few edits, we should run FreeRADIUS in debug mode so we know if we made any mistake, before going further.

First stop the FreeRADIUS service since we can't have 2 instances of the service running simultaneously:

```
$ sudo systemctl stop freeradius.service
```

And run FreeRADIUS in debug mode:

```
$ sudo freeradius -X
```

The output looks something like this:

```
Listening on auth address * port 1812 bound to server default
Listening on acct address * port 1813 bound to server default
Listening on auth address :: port 1812 bound to server
default
Listening on acct address :: port 1813 bound to server
default
Listening on auth address 127.0.0.1 port 18120 bound to
server inner-tunnel
Listening on proxy address * port 52025
Listening on proxy address :: port 42807
Ready to process requests
```

Exit debug mode by pressing **Ctrl+C** and then start FreeRADIUS again by running:

```
$ sudo systemctl start freeradius.service
```

Now FreeRADIUS is installed on your Ubuntu 20.04 machine and is configured to work with MySQL or MariaDB.

Next we'll install daloRADIUS, which is a web control panel to manage our FreeRADIUS server. This step is optional, for those who want a GUI for their FreeRADIUS server.

Install & Configure daloRADIUS (FreeRADIUS GUI) on Ubuntu 20.04 (Optional)

> **daloRADIUS** is a popular RADIUS web management panel, that offers user management, graphical reporting, accounting, a billing engine, integrates with GoogleMaps, and more. It's one of the most popular solutions if you need a **FreeRADIUS GUI**.

First we'll download daloRADIUS from the Github repository .

I'll use **wget** to download it, so I'll have to install it since it's not installed by default, and **unzip** since we'll be downloading a **.zip** file:

```
$ sudo apt -y install wget unzip
```

Now download daloRADIUS and **cd** into the newly created **daloradius-master** folder:

```
$ wget https://github.com/lirantal/daloradius/archive
/master.zip
$ unzip master.zip
$ cd daloradius-master
```

Populate the database with the daloRADIUS schema:

```
$ sudo mysql -u root -p radius < contrib/db/fr2-mysql-
daloradius-and-freeradius.sql
$ sudo mysql -u root -p radius < contrib/db/mysql-
daloradius.sql
```

`cd` out of the `daloradius-master` directory, and move the folder into the document root as `daloradius`:

```
$ cd ..
$ sudo mv daloradius-master /var/www/html/daloradius
```

Next we'll change the owner and group for the **daloradius** folder to www-data:www-data, which are the user and group under which the Apache Web Server runs.

```
$ sudo chown -R www-data:www-data /var/www/html/daloradius/
```

Now we'll need to create the daloRADIUS configuration file. Right now we're just provided a sample file, so we'll make a copy from that sample file:

```
$ sudo cp /var/www/html/daloradius/library
/daloradius.conf.php.sample /var/www/html/daloradius/library
/daloradius.conf.php
```

We'll also change the permissions for the daloRADIUS configuration file:

```
$ sudo chmod 664 /var/www/html/daloradius/library
/daloradius.conf.php
```

Next we edit a few variables in the daloRADIUS connection file, so it's able to connect to the FreeRADIUS database.

Open the configuration file using your favorite editor:

```
$ sudo nano /var/www/html/daloradius/library
/daloradius.conf.php
```

Similarly to what we've done earlier, when editing the FreeRADIUS config file, we just need to adjust the variables for the database user, their password, and the database name. Those are all the edits for the scope of this tutorial.

This is how they initially look in the daloRADIUS configuration file:

```
$configValues['CONFIG_DB_USER'] = 'root';
$configValues['CONFIG_DB_PASS'] = '';
$configValues['CONFIG_DB_NAME'] = 'radius';
```

This is how it looks after editing with the details for the database I created earlier:

```
$configValues['CONFIG_DB_USER'] = 'radius';
$configValues['CONFIG_DB_PASS'] = 'Somestrongpassword_321';
$configValues['CONFIG_DB_NAME'] = 'radius'
```

Lastly restart FreeRADIUS and Apache to make sure everything works:

```
$ sudo systemctl restart freeradius.service apache2
```

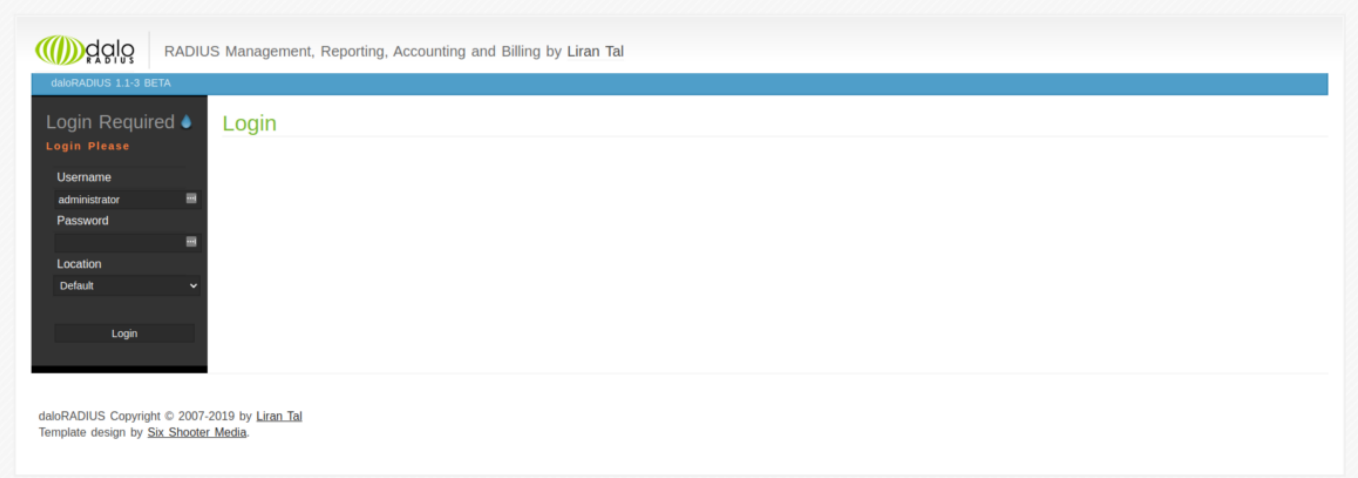
Access daloRADIUS

You can access daloRADIUS through a web browser by visiting:

```
http://server_ip_address/daloradius
```

Make sure it's `http://` and that your browser doesn't automatically change it to `https://` because you may not be able to access daloRADIUS since we haven't configured it to use **SSL**.

The daloRADIUS start page looks something like this:



Default daloRADIUS username/password:

```
username: administrator
password: radius
```

Change daloRADIUS username & password

Having default credentials such as **administrator/radius** is a security vulnerability, and there are bots that scan absolutely all IPs and try known default credentials for certain software.

In this manner someone can be scanning all of the possible IPs and they're trying to detect if daloRADIUS is installed by visiting **http://random_ip/daloradius** and they'll try out to log in using **administrator/radius**, and chances are they'll succeed some of the time because some people don't change their default credentials.

You can change a user password by logging into daloRADIUS > **Config** (In the top menu) > **Operators** (In the submenu) > **List Operators** (In the gray sidebar) > Click on user (in our case **administrator**) and in the next screen change the password and click Apply.

[Video] Change daloRADIUS Administrator Password

Change daloRADIUS Administrator Password

To create a new daloRADIUS user (called **Operator**) go to **Config** > **Operators** (in the submenu) > **New Operator** (in the gray sidebar) > input **Operator Username** and **Operator Password** and click **Apply**.

[Video] Create New daloRADIUS Operator

Create New daloRADIUS Operator

To get acquainted with daloRADIOUS, next we'll create a NAS Client Table and a user, and

then we'll test if everything works correctly by sending an Authentication Request using a software called **NTRadPing**.

Testing FreeRADIUS & daloRADIUS

For the last part of this tutorial we'll test our FreeRADIUS server and the daloRADIUS web panel.

In short, we'll send an **Authentication Request** from another computer to our server to see if it works.

To do this we'll need to add a **NAS** (explanation below), a **User**, and another computer from where to send the request (this can be your computer, for example).

Note: For this demo you'll need to install a Windows software, called **NTRadPing**.

1. Creating a NAS Client Table in daloRADIUS

For another computer to use our RADIUS server, it first needs to be added to the **NAS Client Table**.

The **Network Access Server (NAS) client table** acts as a gateway that guards a protected resource. For another computer to connect to our RADIUS server, it needs to be added to the NAS client table.

The NAS is an intermediary that a client connects to, then the NAS asks the resource (in our case the RADIUS server) if the credentials are valid, and based on this the NAS will allow or disallow access to the protected resource.

You can read a bit more about the NAS on this page from the FreeRADIUS wiki .

To add a NAS, go in the daloRADIUS dashboard, **Management > NAS** (in the blue submenu) **> New NAS** (in the left, dark gray, sidebar).

`NAS IP/Host` : the IP or fully qualified hostname from which you're trying to connect
`NAS Secret` : a password for connecting to the NAS, but it's referred to as a secret. It's used to communicate between the client/NAS and RADIUS server.
`NAS Type` : There are a few types that are recognized, including livingston, cisco, portslave. This is passed to the external checklogin program when it is called to detect double logins. For the purposes of this tutorial we'll select **other**.

`NAS Shortname` : An alias that can be used in place of the IP address or fully qualified hostname provided under NAS IP/Host

For our example we'll fill in:

`NAS IP/Host` : IP of another computer we're using as a client
`NAS Secret` : nobodywilleverlearnthissecret!!11!!
`NAS Type` : other
`NAS Shortname` : ProductionServer

2. Creating a User in daloRADIUS

To test our RADIUS server we'll also need to have a user.

We can easily create one by navigating in the top menu to **Management > Users** (in the blue submenu) **>New User** (in the left, dark gray, sidebar)

For our example all we need is a **Username** and **Password**. There are other attributes, but these will be enough for our purposes.

I'll fill in the following:

Username: new_customer
Password: customer_strong_passwd_123

3. Run FreeRADIUS in Debug Mode

We want to see for ourselves what's happening on the server, so we'll run FreeRADIUS in debug mode.

First stop the running process:

```
sudo systemctl stop freeradius.service
```

And run the following command to run FreeRADIUS debug mode:

```
sudo freeradius -X
```

> **Important Note**
> Every time a new NAS is added **you need to restart FreeRADIUS** so it fetches the updated table.
>
> We're already doing this in this demo, since we're stopping it and running it in debug mode, but you should remember this in the future.

Now we're ready to test the server.

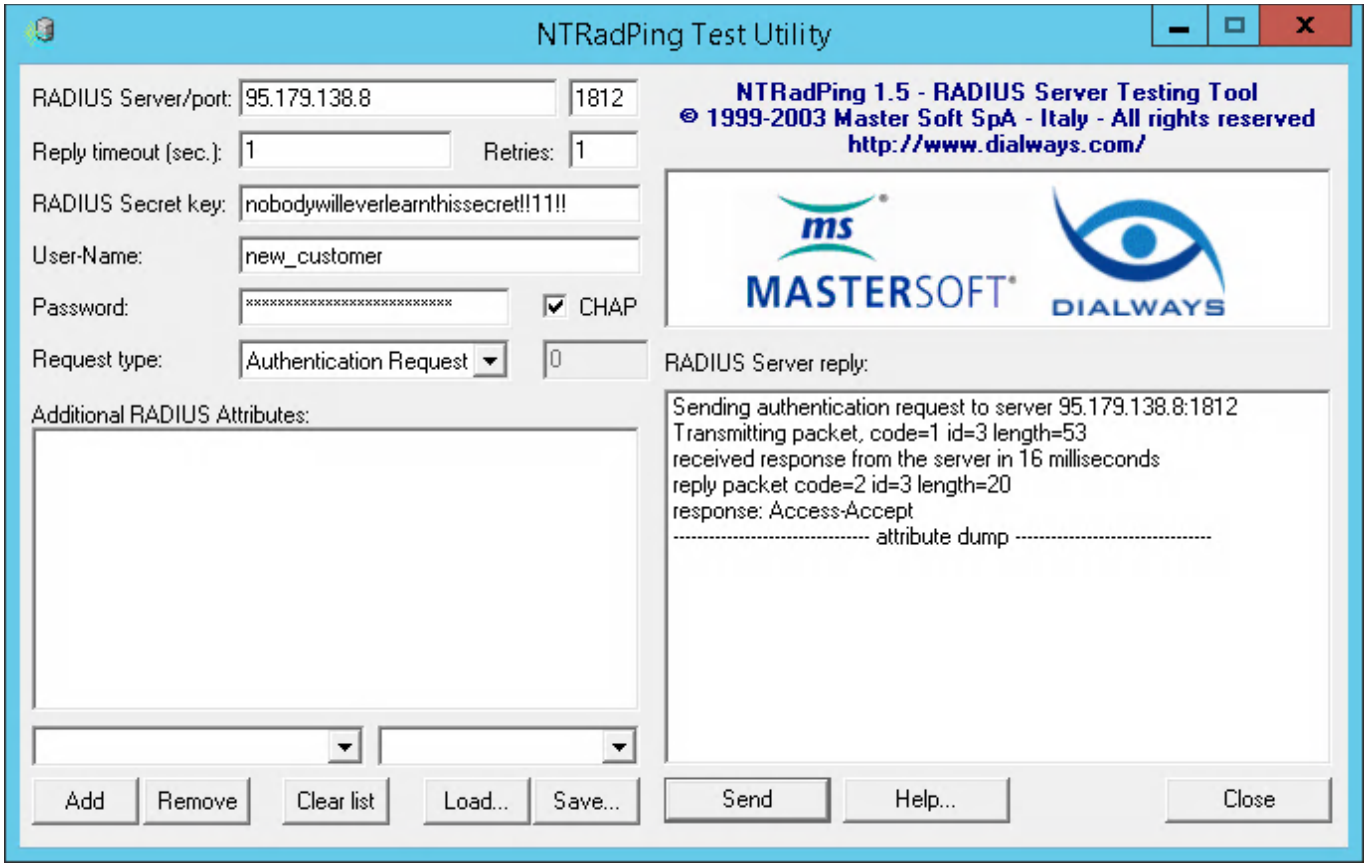
4. Test daloRADIUS with NTRadPing

For convenience, we'll test the server using a free software for Windows, called NTRadPing.

You can download it here https://community.microfocus.com/t5/OES-Tips-Information/NTRadPing-1-5-RADIUS-Test-Utility/ta-p/1777768 . This is a direct link to the archive https://community.microfocus.com/dcvta86296/attachments/dcvta86296/OES_Tips/148/1/ntradping.zip

To run it just unzip the archive and run the executable.

This is how it looks like and how we'll fill in the details in NTRadPing. We'll use it to send an Authentication Request to the RADIUS server while it's running in debug mode, so we can see first hand how it accepts the request.



We've filled the fields as follows:

`RADIUS Server/port` : IP of the server we have FreeRADIUS installed on / port **1812**
`Reply timeout (sec.)` : **1**
`Retries` : 1
`RADIUS Secret key` : nobodywilleverlearnthissecret!!11!!
`User-Name` : new_customer
`Password` : customer_strong_passwd_123

Lastly check the **CHAP** checkbox. This is so the request is made using a CHAP password, instead of the default PAP password.

Now you can test the RADIUS server. Just click Send in NTRadPing and if you get an **Access-Accept** response we can assume it's working.

The output in NTRadPing should look something like this:

```
Sending authentication request to server 40.76.122.52:1812
transmiting Packet, code=1 id=3 length=53
recieved response from the server in 16 milliseconds
replay packet code=2 id=3 length=20
response: Access-Accept
```

```
------------------attribute dump-----------------
```

And the output in the terminal, where you're running FreeRADIUS in debug mode should end with something like this:

```
...
(2) Sent Access-Accept Id 7 from 10.0.2.4:1812 to
213.136.66.127:52163 length 0
(2) Finished request
Waking up in 4.9 seconds.
(2) Cleaning up request packet ID 7 with timestamp +67
Ready to process requests
```

Conclusion

Well done. Hopefully you successfully installed FreeRADIUS with MySQL or MariaDB, along with daloRADIUS, on your Ubuntu 20.04 machine.

If you've encountered any issues then don't hesitate to contact us in the comment section, email, or on social media, and we'll try to assist as soon as possible.

Frequent Errors

"Failed binding with auth address [...]" when running in debug mode

```
Failed binding to auth address * port 1812 bound to server
default: Address already in use
/etc/freeradius/3.0/sites-enabled/default[59]: Error binding
to port for 0.0.0.0 port 1812
```

If you get the following error when running FreeRADIUS in debug mode, it most likely means that the FreeRADIUS service is already running and you need to stop it first.

```
$ sudo systemctl stop freeradius.service
```

TAGS: DaloRADIUS FreeRADIUS Ubuntu 20.04 VPN

18 SHARE 18 TWEET

EdXD

George H.

Subscribe Login

Join the discussion

{} [+]

Name*

Email*

Website

POST COMMENT

23 COMMENTS Oldest

Abebe 8 months ago

hello,

Following The installation manual i installed freeradius, mriadb along with daloradius after the installation i got error in the log section with daloradius and i attached it below.

error reading log file: /tmp/daloradius.log
looked for log file in /tmp/daloradius.log but couldn't find it.
if you know where your daloradius log file is located, set it's location in your library/daloradius.conf file

please can you help me how to correct it.

1 Reply

George H. (@stelixd) 7 months ago

Admin

Reply to Abebe

Hello,

The default location for the radius log is /var/log/freeradius/radius.log

Please add it to the daloradius config file, library/daloradius.conf, as advised by the error message.

-1 Reply

Howard 6 months ago

Do you have any guidance on how to add HTTPS to this installation?

1 Reply

EdXD (@edxd) 6 months ago

Author

Reply to Howard

Hi Howard,

Thanks for getting in touch.

I've been meaning to cover this but haven't been able to do that yet.

I'm thinking this video tutorial may help https://www.youtube.com /watch?v=nFij84u8_Bw

Let me know if it works out?

0 Reply

Howard 5 months ago

Thanks, I will have a look at this. Many thanks, and great tutorial! Bookmarked your site ?

1 Reply

EdXD (@edxd) 5 months ago

Author

Reply to Howard

We're happy to be of service. Thanks so much for the kind words! We very much appreciate it!

0 Reply

gjjb 5 months ago

hi
when I login into Daloradius it show me
Database connection error
Error Message: DB Error: extension not found
and I used ubuntu20.04LTS, mysql8.0 Daloradius version 1.1-2 and Freeradius version 3.
thanks in advance.

0 Reply

EdXD (@edxd) 5 months ago

Author

Reply to gjjb

Hi,

Apologies for the delay.
Did you install FreeRADIUS using this tutorial or different one?

Looking forward to hearing from you.

0 Reply

gjjb 5 months ago

Reply to EdXD

I have followed this tutorial, I have already install Freeradius and daloRADIUS with my sql. but when I tried to login thru daloRADIUS login page, it show error with database extension?

0 Reply

EdXD (@edxd) 5 months ago

Author

Reply to gjjb

Hi,

Apologies for the delay,

I just went through the tutorial and installed MySQL instead of MariaDB and I'm not getting the error.

Did you manage to solve this?

0 Reply

EdXD (@edxd) 5 months ago

Reply to gjjb

Author

Hi,

You have posted the exact same comment twice, so I'll mark the other one with [duplicate comment].

Please let me know if you installed FreeRADIUS by following this tutorial or another tutorial, so I can try to determine the cause of the error.

Thanks!

0 Reply

gjjb 5 months ago

[duplicate comment]

0 Reply

Ray 5 months ago

Hi sir,

i follow the process you mention above and made some modification like installing LAMP using this https://ubuntu.com/server/docs/lamp-applications and I am installing this in Lubuntu 18.04.

Everything went well EXCEPT the last part; using the NTRadPing.

The NAS secrect do not recognize the freeradius server. I try to look in google and give an answer by editing /etc/freeradius/3.0/clients.conf. example below:

client ray {
ipaddr = 192.168.254.0/24
secret = 123
}

Everything is working now and thing that question me is the NAS.
What is NAS?
What is the use of NAS?
Why NAS did not work in this particular.

0 Reply

EdXD (@edxd) 5 months ago

Author

Reply to Ray

Hi,

I tried to explain what a NAS is, and the use, etc. I'm thinking maybe this will help clarify? https://wiki.freeradius.org/glossary/NAS

In my mind I see NAS (Network Access Server) as me telling the FreeRADIUS: "Hey, so, don't let anyone access our stuff if they don't have this IP (NAS IP/Host) and this password (NAS secret)".

I think that maybe the term NAS/Network Access Server just makes everything sound more complicated than it actually is, when you're first exposed to this term. I'm just assuming.

Can you show me a screenshot of NTRadPing with private details blurred out if needed?

I just went through the tutorial to make sure everything works, and I didn't get any errors.

Looking forward to hearing from you.

0 Reply

Alfredo 5 months ago

thanks I did it.

1 Reply

EdXD (@edxd) 5 months ago

Author

Reply to Alfredo

Hi Alfredo,

Really happy to hear! I appreciate you letting us know. It's nice knowing this article is useful. 😀

0 Reply

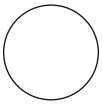
Laco 5 months ago

Hi,
thank you very much for tutorial.
I have installed freeradius, mariadb and daloradius by following this tutorial. GUI daloradius is not working. I can see login page but I end up with blank screen after login

name and password insertion.

Thanks for the answer.

0 Reply



Author

EdXD (@edxd) 5 months ago

Reply to Laco

Hello,

I just went through the tutorial, in case some thing got messed up because of some update, and everything worked, including logging into daloRADIUS.

I have no solid arguments for this, but can you run:
`$ pear install DB`

And then run:
`$ sudo systemctl restart freeradius.service apache2`

Looking forward to hearing from you.

0 Reply

Laco 5 months ago

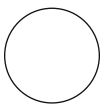
Reply to EdXD

It did not help. The freeradius is working. I have inserted user and NAS by CLI. I have tested function with NTRadPing. But GUI Daloradius still not works. IE reports “This page is not working. HTTP error 500.”. What should be set for “SQL engine” in daloradius.conf.php – mysql?

I have set this:
include (dirname(__FILE__).’/version.php’);

$configValues[‘FREERADIUS_VERSION’] = ‘2’;
$configValues[‘CONFIG_DB_ENGINE’] = ‘mysqli’;
$configValues[‘CONFIG_DB_HOST’] = ‘localhost’;
$configValues[‘CONFIG_DB_PORT’] = ‘3306’;
$configValues[‘CONFIG_DB_USER’] = ‘radius’;
$configValues[‘CONFIG_DB_PASS’] = ‘P@ssw0rd’;
$configValues[‘CONFIG_DB_NAME’] = ‘radius’;

0 Reply



Author

EdXD (@edxd) 5 months ago

Reply to Laco

Hello,

Yes **SQL Engine** should be `$configValues['CONFIG_DB_ENGINE'] = 'mysqli';`

Apologies for the delay. I've been trying to find a solution for this but it's difficult without being able to test out solutions.

Did you manage to solve it?

I can log into your machine and take a look if you'd like me to. Just let me know via email/twitter. It's the most painless way I can think of, and I obviously understand if you prefer not to since you may be running a sensitive system and I'm a complete stranger.

0 Reply

Aeros 5 months ago

Hello,

> I followed your tutorial and everything went well. Is it possib

0 Reply

Tech-lee 1 month ago

/etc/freeradius/3.0/mods-enabled/sql[341]: Reference "${dialect}" not found
Errors reading or parsing /etc/freeradius/3.0/radiusd.conf

I get this error I followed the steps exactly

1 Reply

EdXD (@edxd) 1 month ago

> *Reply to Tech-lee*

Author

Hi Tech-lee,

I think this is because the CMS I'm using – WordPress – converts straight "$dialect") to curly quotes ("${dialect}").

If you copy pasted these $dialect lines, then you must've copied the commands with curly quotes:

```
Change dialect = "sqlite" to dialect = "mysql"
Change driver = "rlm_sql_null" to driver = "rlm_sql_${dialect
```

My apologies, I should have caught onto that when writing the tutorial. I was used to

— PREV ARTICLE
Using WordOps to Install WordPress Automatically on Ubuntu 20.04

NEXT ARTICLE —
Install Chrome Remote Desktop on a Headless Ubuntu 20.04 Server

YOU MAY ALSO LIKE

›sites a U ell, and I overlooked it. It should be fixed now.

:c/freeradius/3.0/mods-enabled/sql and make
gain:

```
sqlite" to dialect = "mysql"
lm_sql_multi" to dialect = "${dialect
```

I went through the tutorial again and it works for me.

Looking forward to hearing from you.

Last edited 1 month ago by EdXD

0 Reply

Using WordOps to Install WordPress Automatically on Ubuntu 20.04

by **EdXD** · March 4, 2021

WordOps is a simple tool that provides the ability to deploy WordPress sites from the command line using…

U — UBUNTU

Using SlickStack to Install WordPress Automatically on Ubuntu 20.04

by **EdXD** · November 23, 2021

SlickStack is essentially a collection of scripts for quickly and easily installing WordPress, with Nginx as a web…

U — UBUNTU

Installing NFS Server and NFS Client on Ubuntu 20.04

by **Ozair Malik** · October 27, 2021

What is an NFS Server? NFS server is also known as Network File System server, was developed by…

M — MISCELLANEOUS

How to Setup Samba Server in Ubuntu 20.04

by **Muhammad Aziz** · October 31, 2021

Samba is a utility present in Linux that allows sharing of folders and printers across different operating systems…

U — UBUNTU

How to Create Sudo User in Ubuntu

by **EdXD** · March 26, 2021

The sudo command is one of the most popular and powerful commands in Linux distros. It originally stood…

U — UBUNTU

How to Install & Configure VNC Server on Ubuntu 20.04

by **EdXD** . July 23, 2021

VNC (Virtual Network Computing) is a visual connection system that enables you to interact with the graphical desktop…

CATEGORIES

Select Category

FEATURED POSTS

Find Large Files and Directories in Linux

December 6, 2021

How to Use the du Command to Find Disk Usage in Linux

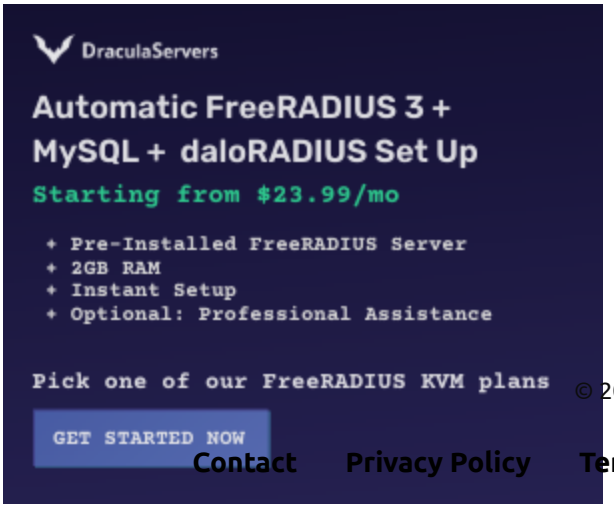
December 4, 2021

How to Use mkdir Command to Create Directories in Linux

December 3, 2021

Cheapest 240Hz Monitors for Gaming

December 3, 2021



© 2021 ByteXD.com