Install FreeRADIUS & daloRADIUS on Oracle Linux 8 + MySQL/MariaDB

FreeRADIUS is an open-source, scalable, modular, and high-performance RADIUS protocol server. FreeRADIUS is in fact the most popular and widely deployed RADIUS server.

It is used mainly to perform AAA i.e. Authorization, Authentication, Accounting services on various types of network access.

To try to put it simply, RADIUS is a protocol that lets you hand it a username and some other authentication info and then answers with a yes/no. In addition, it is designed to **A**uthenticate a user, **A**uthorize them for a specific action, and keep an **A**ccounting (log) of what happened.

It works as a daemon on UNIX and UNIX-like operating systems.

The FreeRADIUS server connects and talks with a client including Ethernet switches, Wireless access points, terminal servers, or a PC configured with the appropriate software (radiusclient, PortSlave, etc).

In this article, we will explain the step-by-step process of installing FreeRADIUS and DaloRADIUS on Oracle Linux 8 with MySQL or MariaDB.

Table of Contents

Prerequisites

- A server running Oracle Linux 8
- We recommend acting as a non-root sudo user. This is because if you are logged in as root you can easily destroy your system if you are not careful.

We will be using DNF (Dandified YUM) instead of YUM. DNF is the default package manager for Fedora 22, CentOS 8, Oracle Linux 8, and RHEL 8 distributions. DNF is the next-generation version intended to replace yum in RPM-based systems. It is a powerful package manager and capable of automatically resolving dependency issues.

Update package index:

```
# sudo dnf -y update
```

Video Version of this Tutorial

We have also included the video version of this tutorial, where go through it from start to finish. This may help in case there is situations where the writing seems unclear and it is easier to follow by watching us go through the steps.

Install LAMP Stack on Oracle Linux 8

LAMP stack is a set of open-source softwares that can be used to create web applications and websites. LAMP stands for its original components Linux, Apache, MySQL, PHP. The LAMP stack is no longer limited to the original open-source components.

Install Apache Web Server

Install the **httpd** package:

```
# sudo dnf install httpd
```

Press **Y** and **ENTER** if prompted.

Start Apache and enable it so it starts on boot:

```
# sudo systemctl start httpd
# sudo systemctl enable --now httpd
```

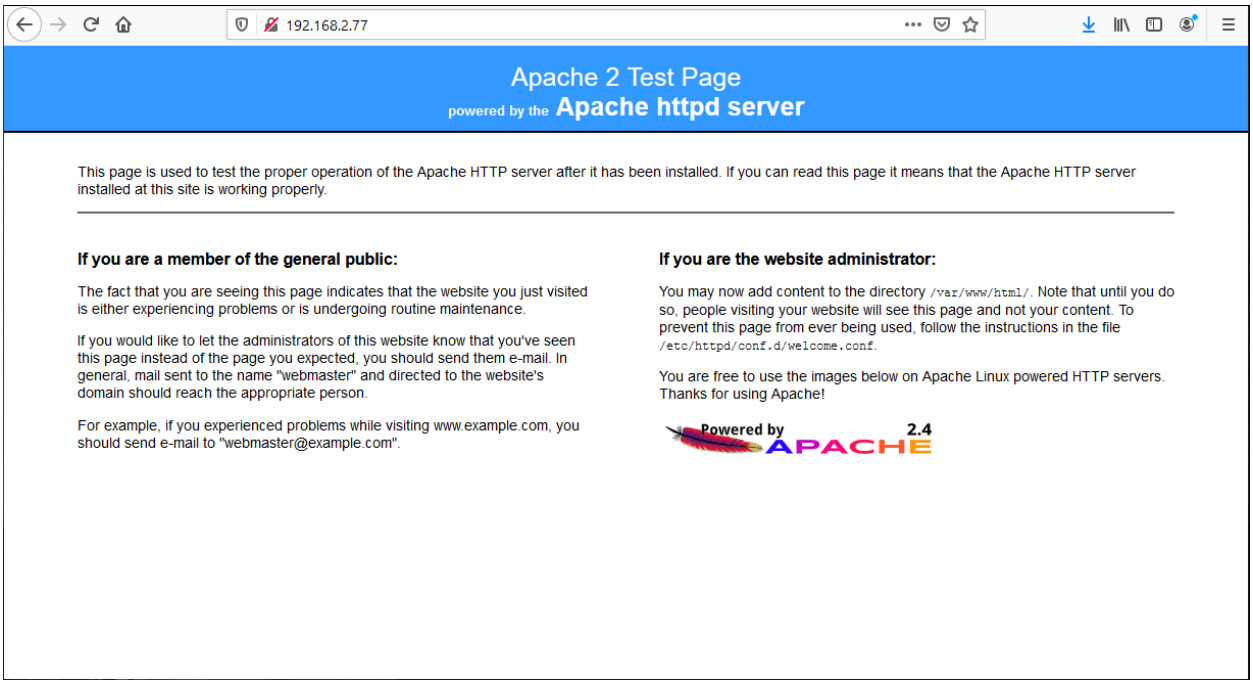
If you have firewalld enabled, we need to enable HTTP connections to Apache:

```
# sudo firewall-cmd --permanent --add-service={http,https}
```

Reload it to apply the changes:

```
# sudo firewall-cmd –reload
```

Check if Apache is running by visiting your server's IP address (with HTTP, not HTTPS), you should see something like this:



Make sure to visit http://your_ip and not https://your_ip. If you get an error check the address you're visiting, because your browser might have added https:// instead of http://

Install MySQL / MariaDB

```
# sudo dnf -y install mariadb-server
# sudo systemctl start mariadb
```

Now we will run a script that MariaDB ships with, which takes us through some steps to improve our MariaDB security options:

```
# sudo mysql_secure_installation
```

You will be taken through a series of prompts. We recommend you fill them out as follows:

Unless you know you set a password, just press enter when prompted:

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

Next confirm that you want to set a new root password and set a strong password:

```
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.
Set root password? [Y/n] Y
New password: Y0ur_Str0ng_Passw0rd
Re-enter new password: Y0ur_Str0ng_Passw0rd
Password updated successfully!
Reloading privilege tables..
... Success!
```

Next you can just press **ENTER** for the prompts that follow (unless you have other plans in mind that we don't cover in this tutorial).

Remove anonymous users:

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

Disallow root login remotely:

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

Remove 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 MariaDB (or MySQL) should be successfully set up on your machine.

Install PHP

Install PHP 7.3 / PHP 7.4 on Oracle Linux 8 (Optional)

The default version distributed with CentOS 8 is PHP7.2. If you want to install PHP 7.2, then you can skip this step.

Enable the EPEL Repository on Oracle Linux 8

Using Oracle provided EPEL repository contents, create EPEL repository file on your Oracle Linux 8 system.

```
sudo vim /etc/yum.repos.d/ol8-epel.repo
```

Paste the contents below into the file then save and quit.

```
[ol8_developer_EPEL]
name= Oracle Linux $releasever EPEL ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL8/developer/EPEL/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
```

Update Yum cache:

```
$ sudo dnf makecache

Oracle Linux 8 EPEL (x86_64)
4.3 kB/s | 2.5 kB     00:00
Oracle Linux 8 BaseOS Latest (x86_64)
5.2 kB/s | 2.7 kB     00:00
Oracle Linux 8 Application Stream (x86_64)
5.0 kB/s | 2.9 kB     00:00
Latest Unbreakable Enterprise Kernel Release 6 for Oracle Linux 8 (x86_64)
4.7 kB/s | 2.5 kB     00:00
Metadata cache created.
```

Using Fedora official EPEL repository

For the open EPEL repository, run the commands:

```
sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-
8.noarch.rpm
```

Accept installation when prompted.

```
Last metadata expiration check: 0:26:49 ago on Mon 11 May 2020 08:30:22 PM UTC.
epel-release-latest-8.noarch.rpm
10 kB/s |  22 kB     00:02
Dependencies resolved.
================================================================================
================================================================
 Package                            Architecture                   Version
Repository                          Size
================================================================================
================================================================
Installing:
 epel-release                       noarch                         8-8.el8
@commandline                        22 k
```

```
Transaction Summary
================================================================================
================================================================
Install  1 Package

Total size: 22 k
Installed size: 32 k
Is this ok [y/N]: y
```

Confirm EPEL is installed on Oracle Linux 8.

```
$ sudo yum repolist
repo id                                          repo name
epel                                             Extra Packages for Enterprise Linux 8
- x86_64
epel-modular                                     Extra Packages for Enterprise Linux
Modular 8 - x86_64
ol8_UEKR6                                        Latest Unbreakable Enterprise Kernel
Release 6 for Oracle Linux 8 (x86_64)
ol8_appstream                                    Oracle Linux 8 Application Stream
(x86_64)
ol8_baseos_latest                                Oracle Linux 8 BaseOS Latest (x86_64)
```

Enable the Remi Repository

```
# sudo dnf install -y dnf-utils http://rpms.remirepo.net/enterprise/remi-release-8.rpm
```

```
# sudo dnf module list php
# sudo dnf module reset php
```

Install PHP 7.3

```
# sudo dnf module enable php:remi-7.3
```

Install PHP 7.4

```
# sudo dnf module enable php:remi-7.4
```

Enable the PowerTools repository, as some packages may depend on some of its packages:

```
# sudo dnf config-manager --set-enabled ol8_codeready_builder
```

Install PHP and some popular PHP extensions:

```
# sudo dnf -y install @php
# sudo dnf -y install php-{common,opcache,cli,gd,curl,mysqlnd,devel,pear,mbstring,xml}
# sudo systemctl enable --now php-fpm
```

If you also intend to install daloRADIUS, the freeRADIUS GUI, you will also need to install PEAR, a PHP extension, from which we will install some dependencies.

```
# sudo dnf -y install php-pear
```

Next we install the DB and MDB2 libraries from the PEAR repository. These are required by daloRADIUS to work.

Warning

The MySQL extension was deprecated in PHP 5.5.0, and it was removed in PHP 7.0.0. Instead, the MySQLi (**My SQL I**mproved) extension should be used.

```
# sudo pear channel-update pear.php.net
# sudo pear install DB MDB2 MDB2_Driver_mysqli
```

We will check if the DB and MDB2 libraries were successfully installed.

```
# sudo pear channel-update pear.php.net
# sudo pear list |grep -e DB -e MDB2
DB                 1.10.0  stable
MDB2               2.4.1   stable
MDB2_Driver_mysqli 1.4.1   stable
```

Install freeRADIUS and Configure with MySQL/MariaDB on CentOS 8

Install FreeRADIUS

Use the **dnf module list** command to list the modules available to our system related to **freeradius**.

```
# sudo dnf module list freeradius
```

You should see something like this:

```
Oracle Linux 8 Application Stream (x86_64)
7.9 kB/s | 3.1 kB     00:00
Oracle Linux 8 Application Stream (x86_64)
387 kB/s |  24 MB     01:02
Oracle Linux 8 CodeReady Builder (x86_64) - Unsupported
6.3 kB/s | 2.8 kB     00:00
Oracle Linux 8 CodeReady Builder (x86_64) - Unsupported
365 kB/s | 4.2 MB     00:11
Remi's Modular repository for Enterprise Linux 8 - x86_64
2.8 kB/s | 3.5 kB     00:01
Remi's Modular repository for Enterprise Linux 8 - x86_64
174 kB/s | 743 kB     00:04
Safe Remi's RPM repository for Enterprise Linux 8 - x86_64
2.6 kB/s | 3.0 kB     00:01
Safe Remi's RPM repository for Enterprise Linux 8 - x86_64
273 kB/s | 1.7 MB     00:06
Latest Unbreakable Enterprise Kernel Release 6 for Oracle Linux 8 (x86_64)
8.2 kB/s | 2.5 kB     00:00
Oracle Linux 8 Application Stream (x86_64)
Name             Stream           Profiles             Summary
freeradius       3.0 [d]          server [d]           High-performance and highly
configurable free RADIUS server

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

Install the FreeRADIUS module, FreeRADIUS client utilities, and the MySQL module for FreeRADIUS.

```
# sudo dnf install -y @freeradius freeradius-utils freeradius-mysql
```

Start the RADIUS service:

```
# sudo systemctl enable --now radiusd.service
```

Check the status of the service to make sure it is active and there are no issues so far:

```
# systemctl status radiusd.service
```

Example output:

```
● radiusd.service - FreeRADIUS high performance RADIUS server.
Loaded: loaded (/usr/lib/systemd/system/radiusd.service; enabled; vendor preset:
disabled)
Active: inactive (dead) since Fri 2020-06-12 07:28:35 UTC; 5h 23min ago
Process: 8163 ExecStart=/usr/sbi/radiusd -d /etc/raddb (code=exited,
status=0/SUCCESS)
Process: 8160 ExecStartPre=/usr/sbin/radiusd -C (code=exited, status=0/SUCCESS)
Process: 8158 ExecStartPre=/bin/chown -R radiusd.radiusd /var/run/radiusd
(code=exited, status=0/SUCCESS)
Main PID: 8166 (code=exited, status=0/SUCCESS)
Jun 12 07:21:21 bytexdradius systemd[1]: Starting FreeRADIUS high performance RADIUS
server....
Jun 12 07:21:21 bytexdradius systemd[1]: Started FreeRADIUS high performance RADIUS
server..
```

Configure Oracle Linux 8 Firewall for FreeRADIUS

We will need to configure Firewalld to allow **radius** packets.

RADIUS uses ports 1812 for authentication and port 1813 for accounting, so we will need to allow traffic on these ports. Upon installing FreeRADIUS, it added it's configuration to firewalld, so to allow **radius** traffic in and out run the following command:

```
# sudo firewall-cmd --add-service=radius –permanent
```

Reload the firewall for changes to take effect:

```
# sudo firewall-cmd –reload
```

You can check the configuration file added by FreeRADIUS to Firewalld by running:

```
# cat /usr/lib/firewalld/services/radius.xml
```

Example output:

```
<!--?xml version="1.0" encoding="utf-8"?-->
RADIUS
The Remote Authentication Dial In User Service (RADIUS) is a protocol for user
authentication over networks. It is mostly used for modem, DSL or wireless user
authentication. If you plan to provide a RADIUS service (e.g. with freeradius),
enable this option.
```

Confirm that the changes have been successfully added to the default zone. The services we mainly want to see are **http**, **https**, and **radius**.

```
$ sudo firewall-cmd --get-default-zone
public

$ sudo firewall-cmd --list-services --zone=public
cockpit dhcpv6-client http https radius ssh
```

The services we allowed so far in this guide (**http**, **https**, and **radius**) are listed in the output, so we can proceed.

You can use iptables.service instead. Just allow the following equivalent ports:

https = tcp 80
https = tcp 443
radius = tcp 1812, udp 1812, tcp 1813, udp 1813

Test the RADIUS Server

To test that FreeRADIUS works we will run it in **debug mode**. To do this we will first have to stop the current running process. If we run it in debug mode while the other process is still running, then we will get an error.

To stop the current process stop the service by running:

```
# sudo systemctl stop radiusd.service
```

Run the RADIUS server in debug mode:

```
# sudo radiusd -X
```

You should get an output ending in:

```
...
Listening on auth address 127.0.0.1 port 18120 bound to server inner-tunnel
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 proxy address * port 33167
Listening on proxy address :: port 57427
Ready to process requests
```

If the output looks good then stop debug mode by pressing Ctrl+C, and start the service again by running:

```
# sudo systemctl start radiusd.service
```

Configure freeRADIUS to use MySQL/MariaDB

Now we will configure MySQL/MariaDB for FreeRADIUS.

First, we will create a database and a database user for FreeRADIUS, then create a database, and a user identified by a password.

In this example we'll use the following details:

Database: radius
User: radius
Password: Somestrongpassword_321

You can replace the names and password with whatever you like, but you will have to pay attention in configurations we will do later on, to appropriately replace values.

Start by accessing the MySQL/MariaDB console as root:

```
# mysql -u root –p
```

Run the commands to create the database and user:

```
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;
```

Next import the RADIUS MySQL schema into the newly created database:

```
# sudo su -
# mysql -u root -p radius < /etc/raddb/mods-config/sql/main/mysql/schema.sql
```

Create a soft link for SQL under /etc/raddb/mods-enabled/

```
# sudo ln -s /etc/raddb/mods-available/sql /etc/raddb/mods-enabled/
```

Now we will configure FreeRADIUS to use MySQL. We do this by editing the file **/etc/raddb/mods-available/sql**. You can use your favorite text editor. I will install and use **nano**:

```
# sudo dnf install -y nano
# sudo nano /etc/raddb/mods-available/sql
```

The file is long, due explanations and lines that are commented out, but we will just edit a few lines:

1. Change driver = "rlm_sql_null" to driver = "rlm_sql_mysql"
2. Change dialect = "sqlite" to dialect = "mysql"

3. Uncomment **server**, **port**, **login**, and **password,** and also change some of their values. They initially look like this:

```
# Connection info:
#
# server = "localhost"
# port = 3306
# login = "radius"
# password = "radpass"
# Database table configuration for everything except Oracle
radius_db = "radius"
```

   Change them by uncommenting them and changing their values to correspond to the database and user you created earlier:

```
# Connection info:
#
server = "localhost"
port = 3306
login = "radius"
password = "Somestrongpassword_321"
# Database table configuration for everything except Oracle
radius_db = "radius"
```

4. Uncomment the line containing read_clients = yes, by removing the # symbol at the beginning of the line.
5. Save the file. If sudo vi /etc/sysconfigyou are using **nano** then press Ctrl+X and ENTER to save the file and exit.

Now change the group rights of the file we just edited to **radiusd**:

```
# sudo chgrp -h radiusd /etc/raddb/mods-enabled/sql
```

And restart the **radiusd** service:

```
# sudo systemctl restart radiusd
```

Since we have made significant changes, we will test again in debug mode to make sure FreeRADIUS is working.

Stop the **radiusd** service:

```
# sudo systemctl stop radiusd
```

And run it in debug mode:

```
# sudo radiusd -X
```

You should get a long output ending in something like:

```
Listening on auth address 127.0.0.1 port 18120 bound to server inner-tunnel
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 proxy address * port 45059
Listening on proxy address :: port 54555
Ready to process requests
```

Now FreeRADIUS is installed and working with MySQL or MariaDB on your Oracle Linux 8 server.

The following steps are to install daloRADIUS, a FreeRADIUS web panel. As such, these are optional, and only if you want to use the web panel.

Install & Configure daloRADIUS (FreeRADIUS GUI) on Oracle Linux 8 (Optional)

daloRADIUS is an advanced RADIUS web management application. It is aimed at managing hotspots and general-purpose ISP deployments. It offers multiple excellent features like a powerful graphic interface, advanced user management, billing engine, integration with Google Maps, and more.

Install **wget**

```
# sudo dnf -y install wget
```

Download the daloRADIUS from Github:

```
# cd /tmp && wget https://github.com/lirantal/daloradius/archive/master.zip
```

Install **unzip** if you don't already have it, unzip the daloRADIUS archive, and move it into the DocumentRoot**.**

Document root file is the folder where website files for a specific domain are stored. It's important to have a unique folder for each domain as cPanel allows for multiple domains (subdomains and add-on domains).

After installation of Apache, the document root file is located at the /var/www/html/ by default but we can change the location of the directory later.

```
# sudo dnf -y install unzip
# unzip master.zip
# sudo mv daloradius-master/ /var/www/html/daloradius
```

Navigate via **cd** in the daloradius folder **/var/www/html/daloradius** so we can easily import daloRADIUS MySQL tables:

```
# cd /var/www/html/daloradius
# mysql -u root -p radius < contrib/db/fr2-mysql-daloradius-and-freeradius.sql
# mysql -u root -p radius < contrib/db/mysql-daloradius.sql
```

Now change the ownership of the **daloradius folder** to the Apache webserver, and we will make the **daloradius.conf.php** configuration file writable by the webserver.

```
# sudo chown -R apache:apache /var/www/html/daloradius/
```

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

Open the **daloradius.conf.php** configuration file so we can edit MySQL information:

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

Change the values to your database user/password/database name:

```
$configValues['CONFIG_DB_HOST'] = 'localhost';
$configValues['CONFIG_DB_PORT'] = '3306';
$configValues['CONFIG_DB_USER'] = 'radius';
$configValues['CONFIG_DB_PASS'] = ‘Somestrongpassword_321’;
$configValues['CONFIG_DB_NAME'] = 'radius';
```

Press Ctrl+X and ENTER to save and exit, if you’re using nano to edit the file.

Restart the **radiusd** service and check it’s status to make sure it’s working.

```
# sudo systemctl restart radiusd.service httpd
# systemctl status radiusd.service httpd
```

By default, Oracle Linux 8 has SELinux enabled and in enforcing mode.

SELinux (Security-Enhanced Linux) is a Linux Kernel security module. It gives administrators more control over who can access the system. It was first introduced in RHEL 4 and improved in the later RHEL versions.

We will have to make changes to the SELinux policy to allow apache user access, and we’ll do this using the **semanage** command.

SEmanage is used to configure certain elements of SELinux policy without modifying or recompiling the policy resources. Semanage command can be used to adjust port context, file context, and booleans.

This allows us to browse the existing default context policies and create our own policies.

To enable the semanage command, we will install SELinux Policy Core Python Utilities, which contains **semanage**.

```
# sudo dnf -y install policycoreutils-python-utils
```

You might face problems in deploying web applications on Oracle Linux 8 while not using the default Apache directories (for content or log). We can create custom policies to apply the proper SELinux context types to your files and directories. This will give you independence in the placement of your application files.

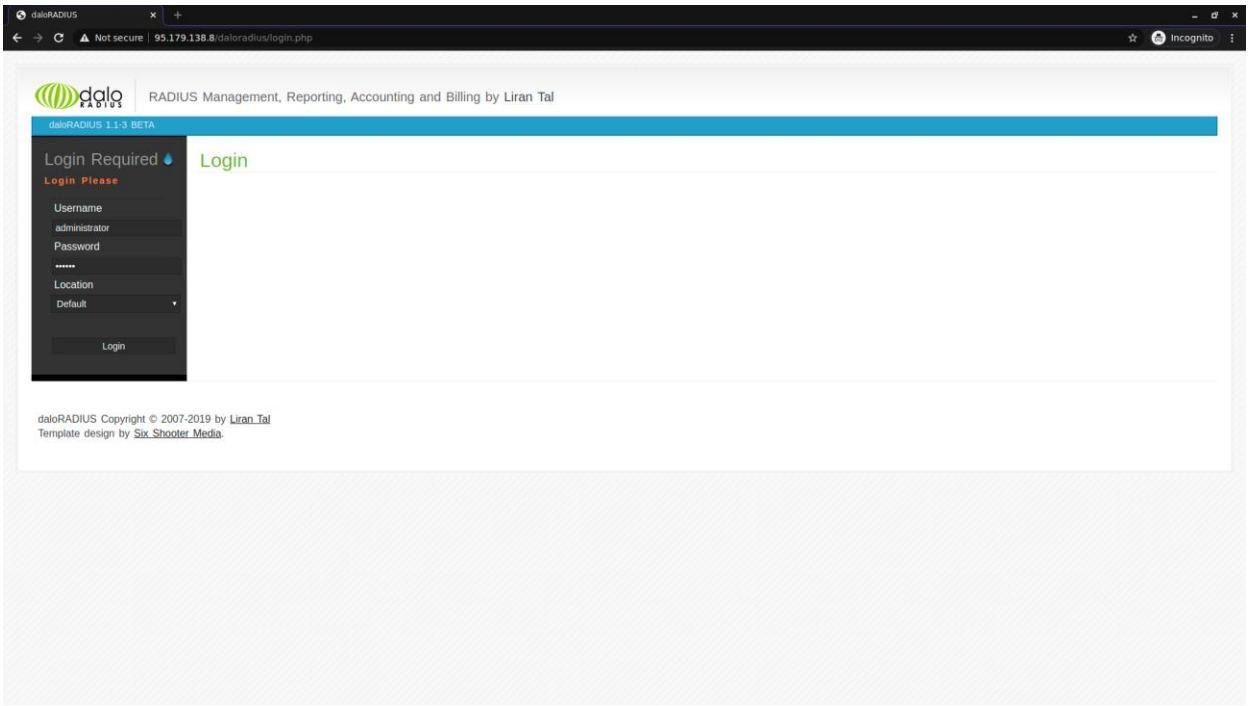
This will protect you if (for some reason) your context settings disappear. You can quickly solve the problem by running a context restore to get your application running again in no time!

Now we will create and apply the policy to allow Apache to read and write daloRADIUS files:

```
# sudo semanage fcontext -a -t httpd_sys_rw_content_t "/var/www/html/daloradius(/.*)?"
# sudo restorecon -Rv /var/www/html/daloradius
```

Now daloRADIUS should be installed and working.

To access it visit http://your_server_ip_or_domain/daloradius. If you get an error then check to see if your browser changed http:// into https:// and change it back.



Change daloRADIUS Administrator Password

Having a default login like administrator / radius is a security vulnerability for anyone scanning for servers with daloRADIUS installed, so you will want to change your password right away.

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

Change daloRADIUS Administrator Password

Testing daloRADIUS Web Panel

To test that FreeRADIUS and daloRADIUS are working, we will perform some basic operations in daloRADIUS and then send a test Authentication Request from another computer to our FreeRADIUS server.

1. Creating a 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, and 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 create the NAS table, in the top menu navigate to **Management**, and in the submenu click on **NAS**. Then in the left sidebar click **New NAS**.

You will have to fill in the following:

NAS IP/Host: the IP or fully qualified hostname from which you are trying to connect

NAS Secret: a password for connecting to the NAS, but it is referred to as a **secret**. It is 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 will go for 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 will fill in:

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

2. Create a User

To create a user navigate in the top menu to **Management**, in the submenu **Users,** and next in the left sidebar **New User**.

We will just fill in **Username** and **Password** and leave the **Password Type** and **Group** as they are.

For our example, we will fill in:

Username: new_customer
Password: customer_strong_passwd_123

There are more attributes to configure, but for the purpose of this tutorial, we are filling in basic info to get FreeRADIUS set up and working.

Now we can test our NAS table and user.

Important Note

Every time a NAS is added, **you need to restart** FreeRADIUS so it fetches the updated table.

To test FreeRADIUS we will run it in debug mode so we can see the output when we try to connect with our newly created user.

The service is probably still running normally, so we will first stop it and then run it in debug mode as we have done when we first tested it:

Stop the **radiusd** service:

```
# sudo systemctl stop radiusd
```

And run it in debug mode:

```
# sudo radiusd -X
```

The output should be 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 41582
Listening on proxy address :: port 35140
Ready to process requests
```

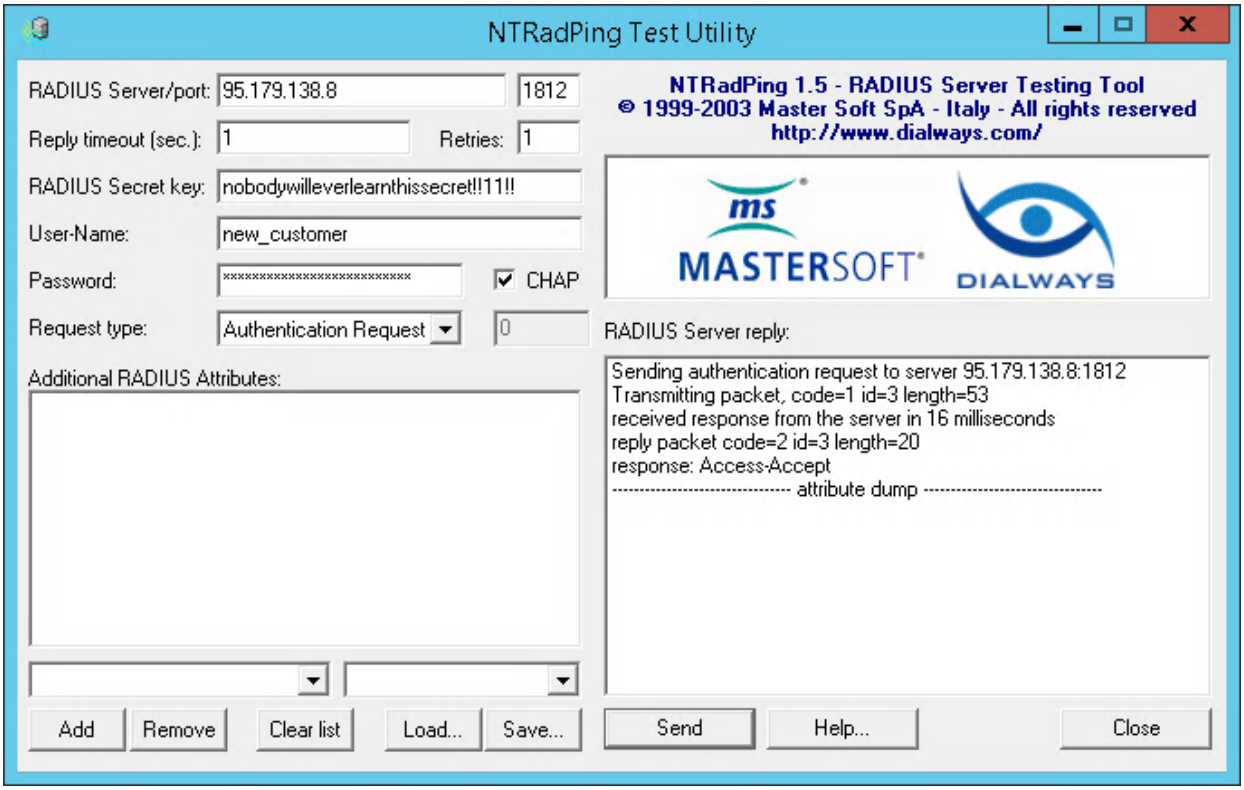
3. Test with NTRadPing

For convenience, we will 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 will fill in the details in NTRadPing. We will 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 have 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:

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 is working.

The output should look something like this:

```
Sending authentication request to server 95.179.138.8:1812
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-----------------
```

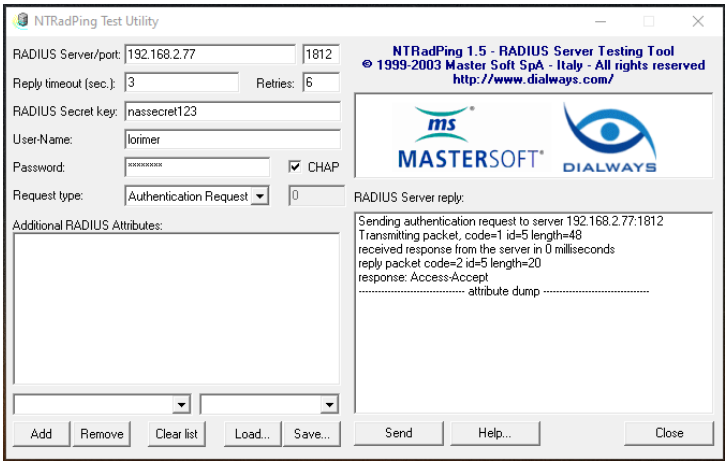
Sample actual test output on server running radius –X:

```
(4) Received Access-Request Id 5 from 192.168.2.128:61195 to 192.168.2.77:1812 length 48
(4)   User-Name = "lorimer"
(4)   CHAP-Password = 0x9472be10e94170b57f72c22f3b59b84146
(4) # Executing section authorize from file /etc/raddb/sites-enabled/default
(4)   authorize {
(4)     policy filter_username {
(4)       if (&User-Name) {
(4)       if (&User-Name)  -> TRUE
(4)       if (&User-Name)  {
(4)         if (&User-Name =~ / /) {
(4)         if (&User-Name =~ / /)  -> FALSE
(4)         if (&User-Name =~ /@[^@]*@/ ) {
(4)         if (&User-Name =~ /@[^@]*@/ )  -> FALSE
(4)         if (&User-Name =~ /\.\./ ) {
(4)         if (&User-Name =~ /\.\./ )  -> FALSE
(4)         if ((&User-Name =~ /@/) && (&User-Name !~ /@(.+)\.(.+)$/))  {
(4)         if ((&User-Name =~ /@/) && (&User-Name !~ /@(.+)\.(.+)$/))   -> FALSE
(4)         if (&User-Name =~ /\.$/)  {
(4)         if (&User-Name =~ /\.$/)   -> FALSE
(4)         if (&User-Name =~ /@\./)  {
(4)         if (&User-Name =~ /@\./)   -> FALSE
(4)       } # if (&User-Name)  = notfound
(4)     } # policy filter_username = notfound
(4)     [preprocess] = ok
(4) chap:   &control:Auth-Type := CHAP
(4)     [chap] = ok
```

```
(4)     [mschap] = noop
(4)     [digest] = noop
(4) suffix: Checking for suffix after "@"
(4) suffix: No '@' in User-Name = "lorimer", looking up realm NULL
(4) suffix: No such realm "NULL"
(4)     [suffix] = noop
(4) eap: No EAP-Message, not doing EAP
(4)     [eap] = noop
(4)     [files] = noop
(4) sql: EXPAND %{User-Name}
(4) sql:    --> lorimer
(4) sql: SQL-User-Name set to 'lorimer'
rlm_sql (sql): Closing connection (14): Hit idle_timeout, was idle for 1702 seconds
rlm_sql (sql): You probably need to lower "min"
rlm_sql_mysql: Socket destructor called, closing socket
rlm_sql (sql): Closing connection (15): Hit idle_timeout, was idle for 1702 seconds
rlm_sql (sql): You probably need to lower "min"
rlm_sql_mysql: Socket destructor called, closing socket
rlm_sql (sql): Closing connection (13): Hit idle_timeout, was idle for 1702 seconds
rlm_sql (sql): You probably need to lower "min"
rlm_sql_mysql: Socket destructor called, closing socket
rlm_sql (sql): 0 of 0 connections in use.  You  may need to increase "spare"
rlm_sql (sql): Opening additional connection (16), 1 of 32 pending slots used
rlm_sql_mysql: Starting connect to MySQL server
rlm_sql_mysql: Connected to database 'radius' on Localhost via UNIX socket, server version 10.3.27-
MariaDB, protocol version 10
rlm_sql (sql): Reserved connection (16)
(4) sql: EXPAND SELECT id, username, attribute, value, op FROM radcheck WHERE username = '%{SQL-User-
Name}' ORDER BY id
(4) sql:    --> SELECT id, username, attribute, value, op FROM radcheck WHERE username = 'lorimer' ORDER
BY id
(4) sql: Executing select query: SELECT id, username, attribute, value, op FROM radcheck WHERE username =
'lorimer' ORDER BY id
(4) sql: User found in radcheck table
(4) sql: Conditional check items matched, merging assignment check items
(4) sql:   Cleartext-Password := "password"
(4) sql: EXPAND SELECT id, username, attribute, value, op FROM radreply WHERE username = '%{SQL-User-
Name}' ORDER BY id
(4) sql:    --> SELECT id, username, attribute, value, op FROM radreply WHERE username = 'lorimer' ORDER
BY id
(4) sql: Executing select query: SELECT id, username, attribute, value, op FROM radreply WHERE username =
'lorimer' ORDER BY id
rlm_sql (sql): 1 of 1 connections in use.  You  may need to increase "spare"
rlm_sql (sql): Opening additional connection (17), 1 of 31 pending slots used
rlm_sql_mysql: Starting connect to MySQL server
rlm_sql_mysql: Connected to database 'radius' on Localhost via UNIX socket, server version 10.3.27-
MariaDB, protocol version 10
rlm_sql (sql): Reserved connection (17)
rlm_sql (sql): Released connection (17)
Need 1 more connections to reach min connections (3)
rlm_sql (sql): Opening additional connection (18), 1 of 30 pending slots used
rlm_sql_mysql: Starting connect to MySQL server
rlm_sql_mysql: Connected to database 'radius' on Localhost via UNIX socket, server version 10.3.27-
MariaDB, protocol version 10
(4) sql: EXPAND SELECT groupname FROM radusergroup WHERE username = '%{SQL-User-Name}' ORDER BY priority
(4) sql:    --> SELECT groupname FROM radusergroup WHERE username = 'lorimer' ORDER BY priority
(4) sql: Executing select query: SELECT groupname FROM radusergroup WHERE username = 'lorimer' ORDER BY
priority
(4) sql: User not found in any groups
rlm_sql (sql): Released connection (16)
(4)     [sql] = ok
(4)     [expiration] = noop
(4)     [logintime] = noop
(4) pap: WARNING: Auth-Type already set.  Not setting to PAP
(4)     [pap] = noop
(4)   } # authorize = ok
(4) Found Auth-Type = CHAP
(4) # Executing group from file /etc/raddb/sites-enabled/default
(4)   Auth-Type CHAP {
```

```
(4) chap: Comparing with "known good" Cleartext-Password
(4) chap: CHAP user "lorimer" authenticated successfully
(4)     [chap] = ok
(4)   } # Auth-Type CHAP = ok
(4) # Executing section post-auth from file /etc/raddb/sites-enabled/default
(4)   post-auth {
(4)     if (session-state:User-Name && reply:User-Name && request:User-Name && (reply:User-Name ==
request:User-Name)) {
(4)     if (session-state:User-Name && reply:User-Name && request:User-Name && (reply:User-Name ==
request:User-Name))  -> FALSE
(4)     update {
(4)       No attributes updated for RHS &session-state:
(4)     } # update = noop
(4) sql: EXPAND .query
(4) sql:    --> .query
(4) sql: Using query template 'query'
rlm_sql (sql): Reserved connection (16)
(4) sql: EXPAND %{User-Name}
(4) sql:    --> lorimer
(4) sql: SQL-User-Name set to 'lorimer'
(4) sql: EXPAND INSERT INTO radpostauth (username, pass, reply, authdate) VALUES ( '%{SQL-User-Name}',
'%{%{User-Password}:-%{Chap-Password}}', '%{reply:Packet-Type}', '%S')
(4) sql:    --> INSERT INTO radpostauth (username, pass, reply, authdate) VALUES ( 'lorimer',
'0x9472be10e94170b57f72c22f3b59b84146', 'Access-Accept', '2021-04-12 13:09:05')
(4) sql: Executing query: INSERT INTO radpostauth (username, pass, reply, authdate) VALUES ( 'lorimer',
'0x9472be10e94170b57f72c22f3b59b84146', 'Access-Accept', '2021-04-12 13:09:05')
(4) sql: SQL query returned: success
(4) sql: 1 record(s) updated
rlm_sql (sql): Released connection (16)
(4)     [sql] = ok
(4)     [exec] = noop
(4)     policy remove_reply_message_if_eap {
(4)       if (&reply:EAP-Message && &reply:Reply-Message) {
(4)       if (&reply:EAP-Message && &reply:Reply-Message)  -> FALSE
(4)       else {
(4)         [noop] = noop
(4)       } # else = noop
(4)     } # policy remove_reply_message_if_eap = noop
(4)   } # post-auth = ok
(4) Sent Access-Accept Id 5 from 192.168.2.77:1812 to 192.168.2.128:61195 length 0
(4) Finished request
Waking up in 4.9 seconds.
(4) Cleaning up request packet ID 5 with timestamp +2295
Ready to process requests
```

Output on the client side running NTRadPing.exe:



Conclusion

Well done! You have set up FreeRADIUS, configured it to use MySQL or MariaDB, along with daloRADIUS, on an Oracle Linux 8 server.

FreeRADIUS FAQ

What is FreeRADIUS used for?

FreeRADIUS is an open-source implementation of the RADIUS protocol server that is mainly used to authenticate various types of network access.

FreeRADIUS provides protocols for:

- Authorization
- Authentication
- Accounting

What is a RADIUS server and how does it work?

RADIUS is short for Remote Authentication Dial-in User Service. It is a networking protocol that provides AAA (authorization, authentication, and account) management protocols. The radius server works on the application layer and uses TCP/UDP protocols on the transport layer.

What are some RADIUS server use cases?

The RADIUS server is a client/server protocol and software. It enables remote access servers to communicate with a central server in order to authenticate users and authorize their access to the requested server or network.