

Lab Task 6: Securing Apache Web Server - 2

Task-1 (5 Marks):

If you want to fully secure your website, you must ensure that nobody can access it via HTTP. For this, we will use a specific module of Apache called `mod_rewrite`.

Step 1: You can use the `a2enmod` command to enable a module and the `a2dismod` command to disable a module. Enable the `mod_rewrite` module using `a2enmod` command.

`sudo a2enmod rewrite`

Step 2: Restart the Apache server

`sudo systemctl restart apache2`

Task-2 (7 Marks):

In this task, we will utilise a rudimentary authentication mechanism of Apache. The premise is that not all users can access your site. It can be accessed only by properly authenticated users.

Step 1: Add users to your Apache web server using the following command

`sudo htpasswd -c /etc/apache2/.htpasswd username` (change username with the username that you want for your first user)

- Add a second user to your Apache server using the `htpasswd` command.

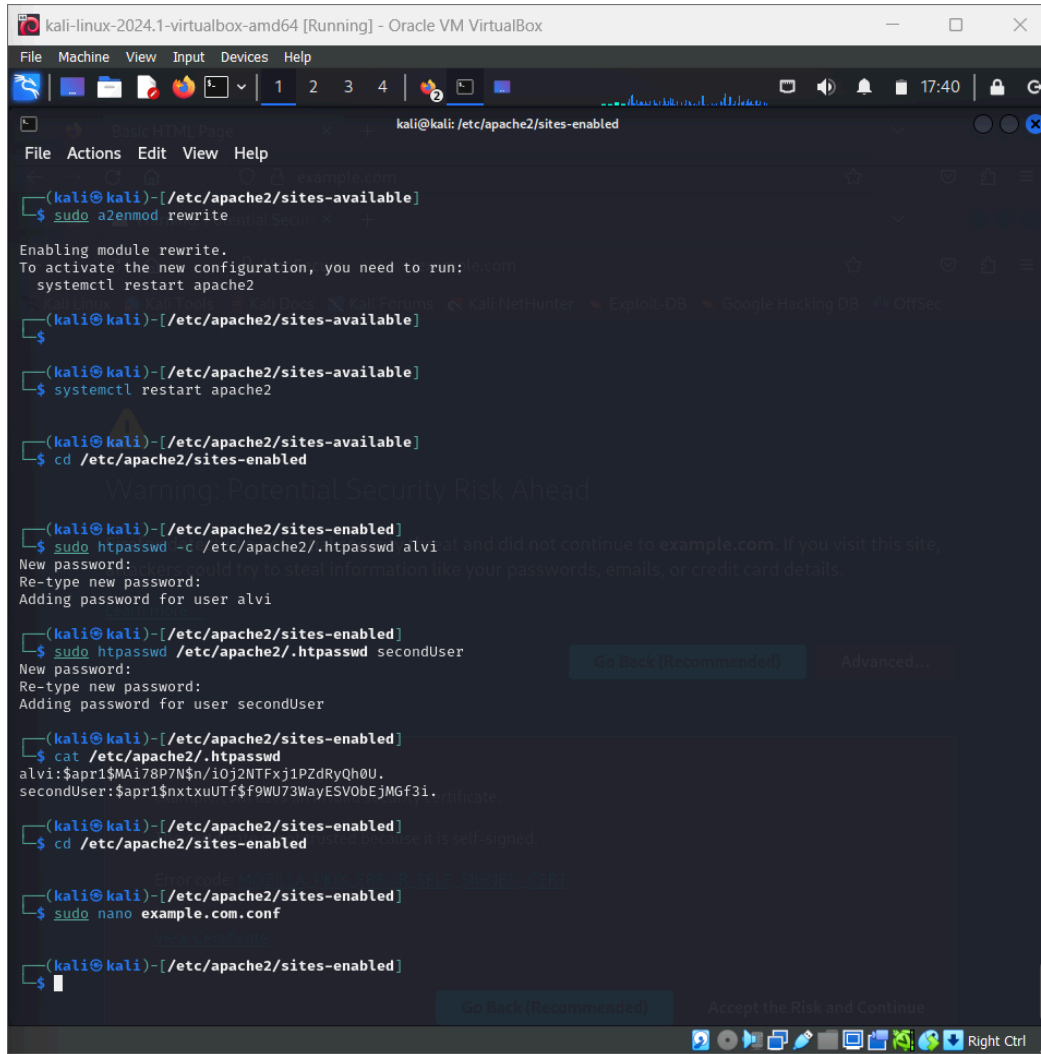
Step 2: Use the following command to cat the contents of `.htpasswd` file:

`cat /etc/apache2/.htpasswd`

You will see something like the following, containing the usernames and their corresponding hashed passwords.

Step 3: Next, add the following into your `https` configuration file for `example.com`.

Step 4: Restart the apache server.



```
kali-linux-2024.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
kali@kali: /etc/apache2/sites-enabled
File Actions Edit View Help
(kali@kali)-[/etc/apache2/sites-available]
$ sudo a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
systemctl restart apache2
(kali@kali)-[/etc/apache2/sites-available]
$ systemctl restart apache2
(kali@kali)-[/etc/apache2/sites-enabled]
$ cd /etc/apache2/sites-enabled
Warning: Potential Security Risk Ahead
(kali@kali)-[/etc/apache2/sites-enabled]
$ sudo htpasswd -c /etc/apache2/.htpasswd alvi
New password:
Re-type new password:
Adding password for user alvi
(kali@kali)-[/etc/apache2/sites-enabled]
$ sudo htpasswd /etc/apache2/.htpasswd secondUser
New password:
Re-type new password:
Adding password for user secondUser
(kali@kali)-[/etc/apache2/sites-enabled]
$ cat /etc/apache2/.htpasswd
alvi:$apr1$MAi78P7N$n/i0j2NTFxj1P2dRyQh0U.
secondUser:$apr1$nxTxuUTf$f9WU73WayESV0bEjMGf3i.
(kali@kali)-[/etc/apache2/sites-enabled]
$ cd /etc/apache2/sites-enabled
(kali@kali)-[/etc/apache2/sites-enabled]
$ sudo nano example.com.conf
(kali@kali)-[/etc/apache2/sites-enabled]
$
```

Task-3 (8 Marks):

Modern web servers often rely on a database to authenticate a user. In this task, you will need an authentication mechanism that relies on MySQL database.

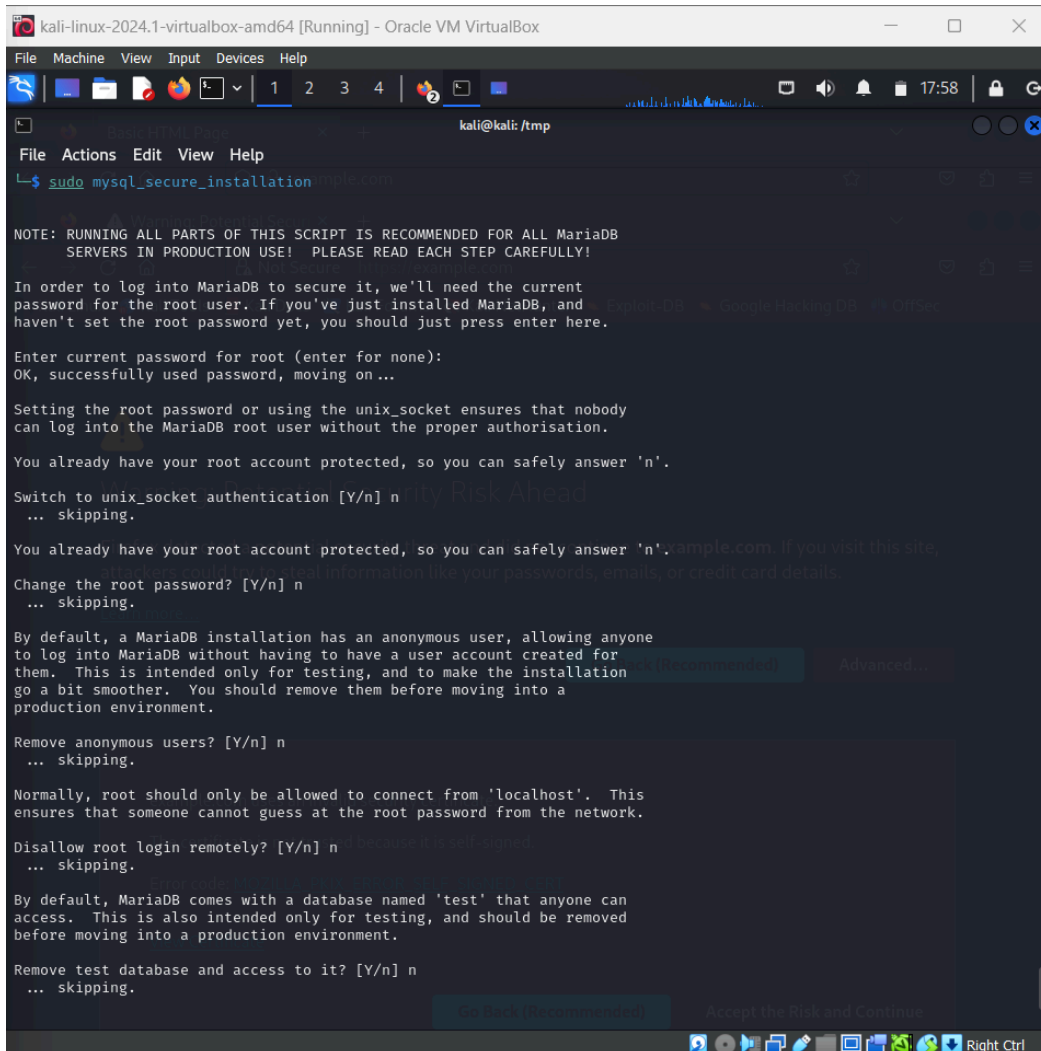
Step 1: Install MySQL server on your ubuntu using the following commands:

```
sudo apt-get update
sudo apt-get install mysql-server
sudo apt-get install libapril1-dbd-mysql
```

Step 2: Configuring MySQL using the following command:

Mysql_secure_installation

Select no for every option except the last when you are prompted to reload the privilege table.



```
kali@kali: /tmp
$ sudo mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
... skipping.

You already have your root account protected, so you can safely answer 'n'.
Change the root password? [Y/n] n
... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] n
... skipping.

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
... skipping.

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

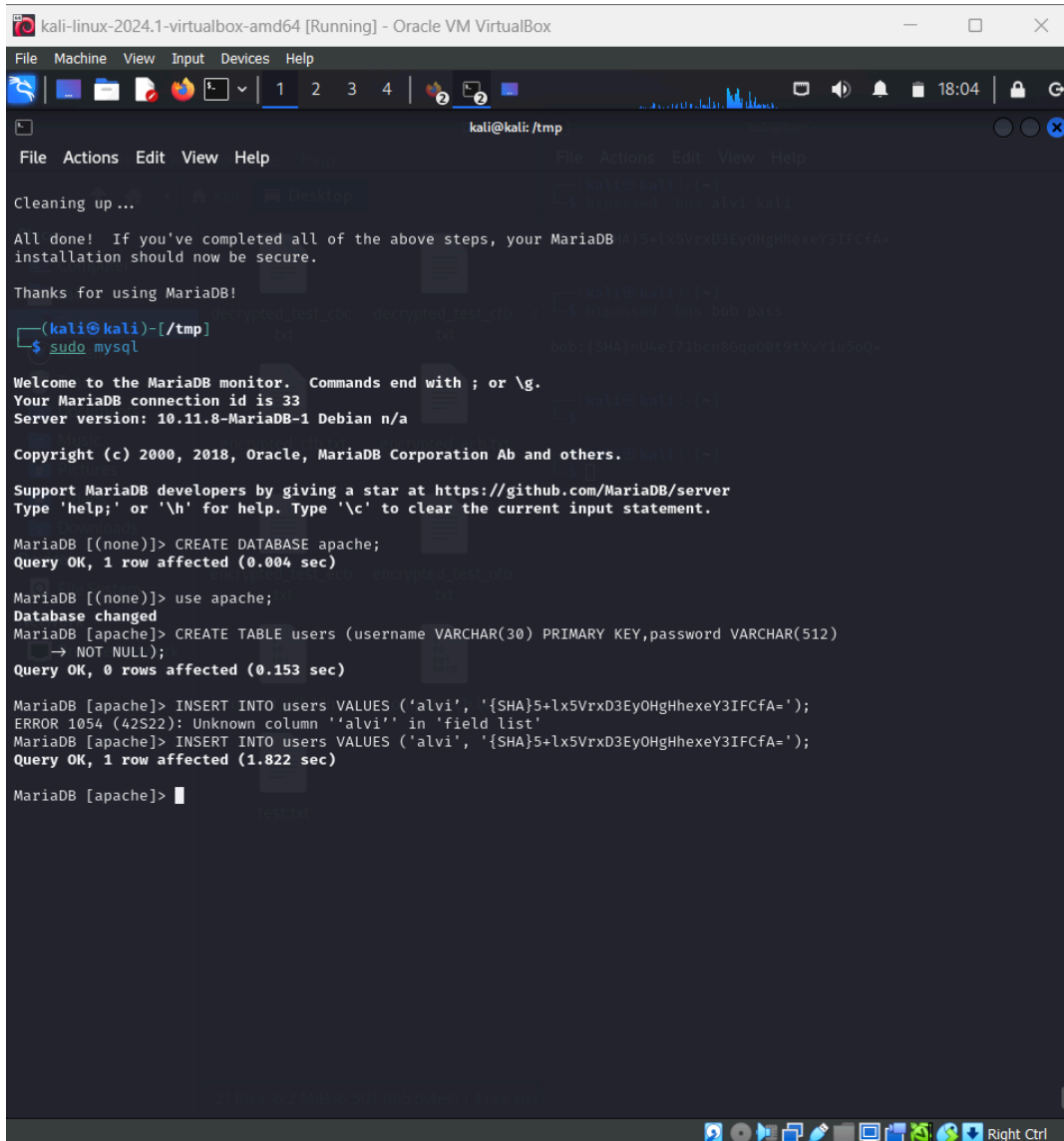
Remove test database and access to it? [Y/n] n
... skipping.

Go Back (Recommended) Accept the Risk and Continue
```

Step 5: Create a database called apache using the following command in the MySQL console:

CREATE DATABASE apache;

Then create a database that will hold the user data and passwords.



```
kali@kali: /tmp
File Actions Edit View Help
Cleaning up ...
All done! If you've completed all of the above steps, your MariaDB installation should now be secure.
Thanks for using MariaDB!
(kali@kali)-[/tmp]
$ sudo mysql

Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 33
Server version: 10.11.8-MariaDB-1 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

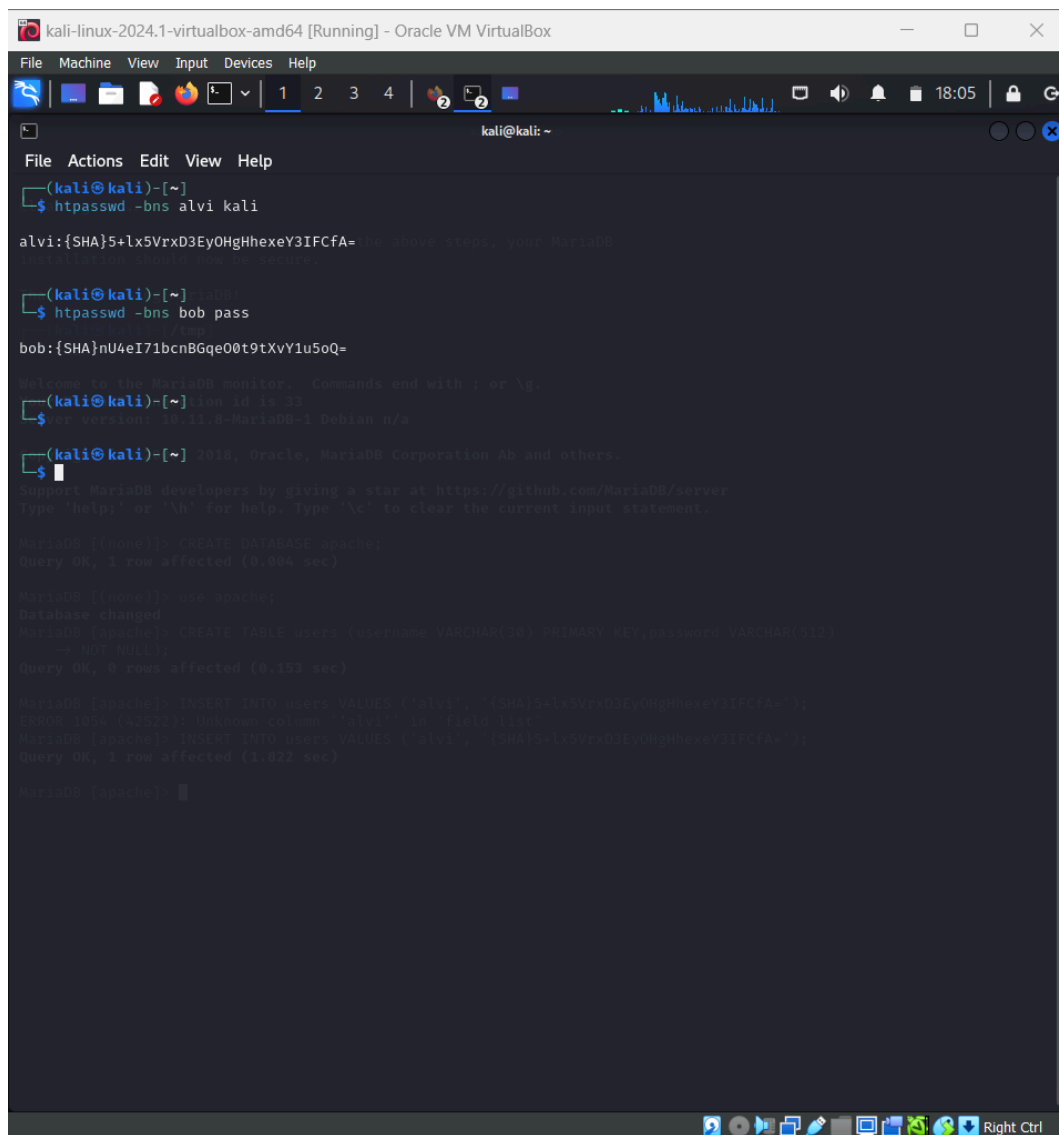
MariaDB [(none)]> CREATE DATABASE apache;
Query OK, 1 row affected (0.004 sec)

MariaDB [(none)]> use apache;
Database changed
MariaDB [apache]> CREATE TABLE users (username VARCHAR(30) PRIMARY KEY,password VARCHAR(512)
→ NOT NULL);
Query OK, 0 rows affected (0.153 sec)

MariaDB [apache]> INSERT INTO users VALUES ('alvi', '{SHA}5+lx5VrxD3Ey0HgHhexeY3IFCfA=');
ERROR 1054 (42S22): Unknown column 'alvi' in 'field list'
MariaDB [apache]> INSERT INTO users VALUES ('alvi', '{SHA}5+lx5VrxD3Ey0HgHhexeY3IFCfA=');
Query OK, 1 row affected (1.822 sec)

MariaDB [apache]> 
```

Step 6: Now, we will add users to our table. But before that we need to create a hashed password which will be stored in the database.



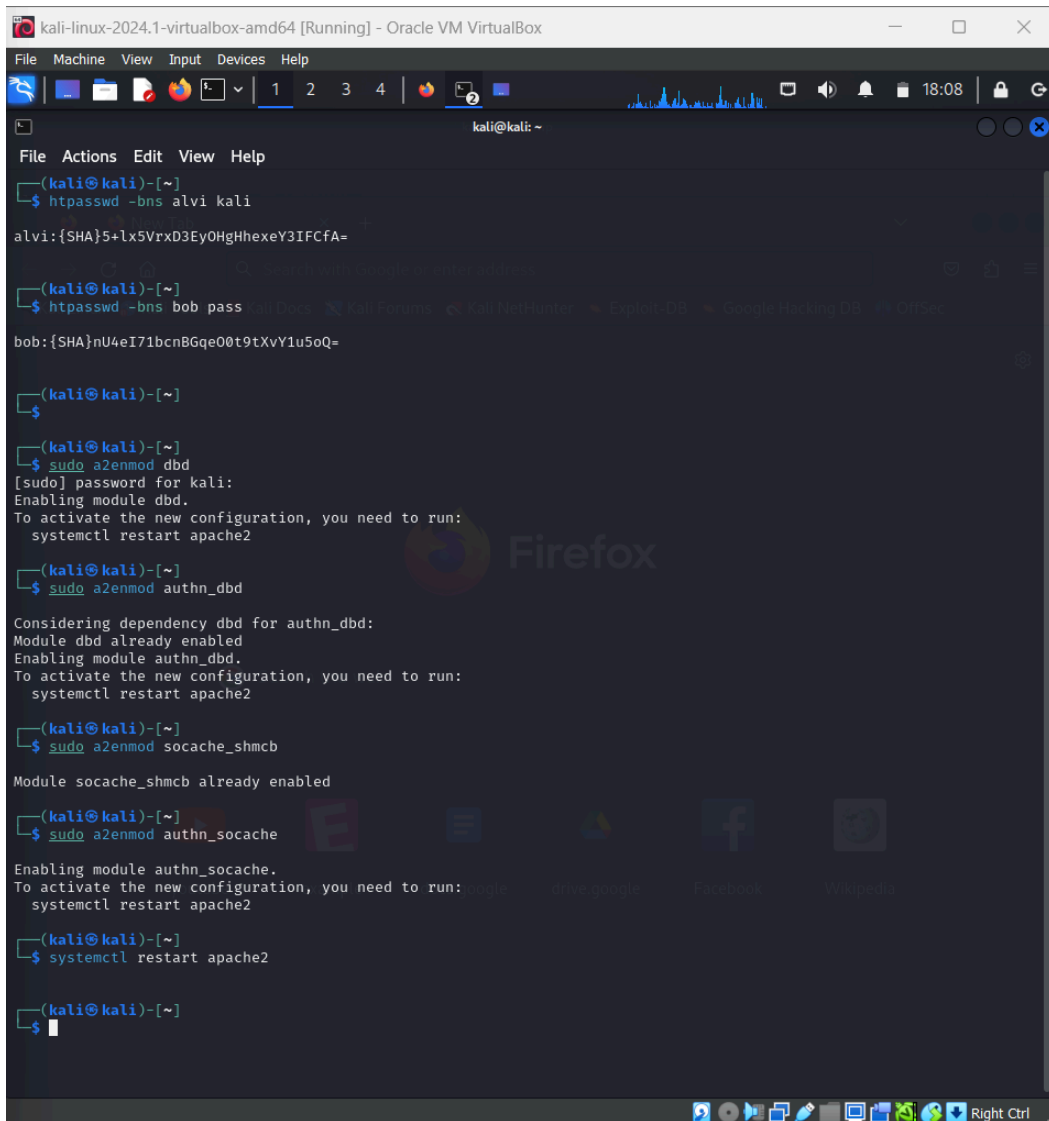
```
kali@kali: ~  
File Actions Edit View Help  
$(kali@kali)-[~]  
$ htpasswd -bns alvi kali  
alvi:{SHA}5+lx5VrxD3Ey0HgHhexeY3IFCfA=  
$(kali@kali)-[~]  
$ htpasswd -bns bob pass  
bob:{SHA}nU4eI71bcnBGqe00t9tXvY1u5oQ=  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
$(kali@kali)-[~]  
$  
MariaDB [(none)]> CREATE DATABASE users;  
Query OK, 1 row affected (0.004 sec)  
MariaDB [(none)]> use apache;  
Database changed  
MariaDB [apache]> CREATE TABLE users (username VARCHAR(30) PRIMARY KEY,password VARCHAR(30) NOT NULL);  
Query OK, 0 rows affected (0.153 sec)  
MariaDB [apache]> INSERT INTO users VALUES ('alvi','{SHA}5+lx5VrxD3Ey0HgHhexeY3IFCfA=');  
ERROR 1064 (42000): Unknown column 'alvi' in 'field list'  
MariaDB [apache]> INSERT INTO users VALUES ('bob','{SHA}nU4eI71bcnBGqe00t9tXvY1u5oQ=');  
Query OK, 1 row affected (1.622 sec)  
MariaDB [apache]>
```

The copy the username and password in the user table using **INSERT** command.

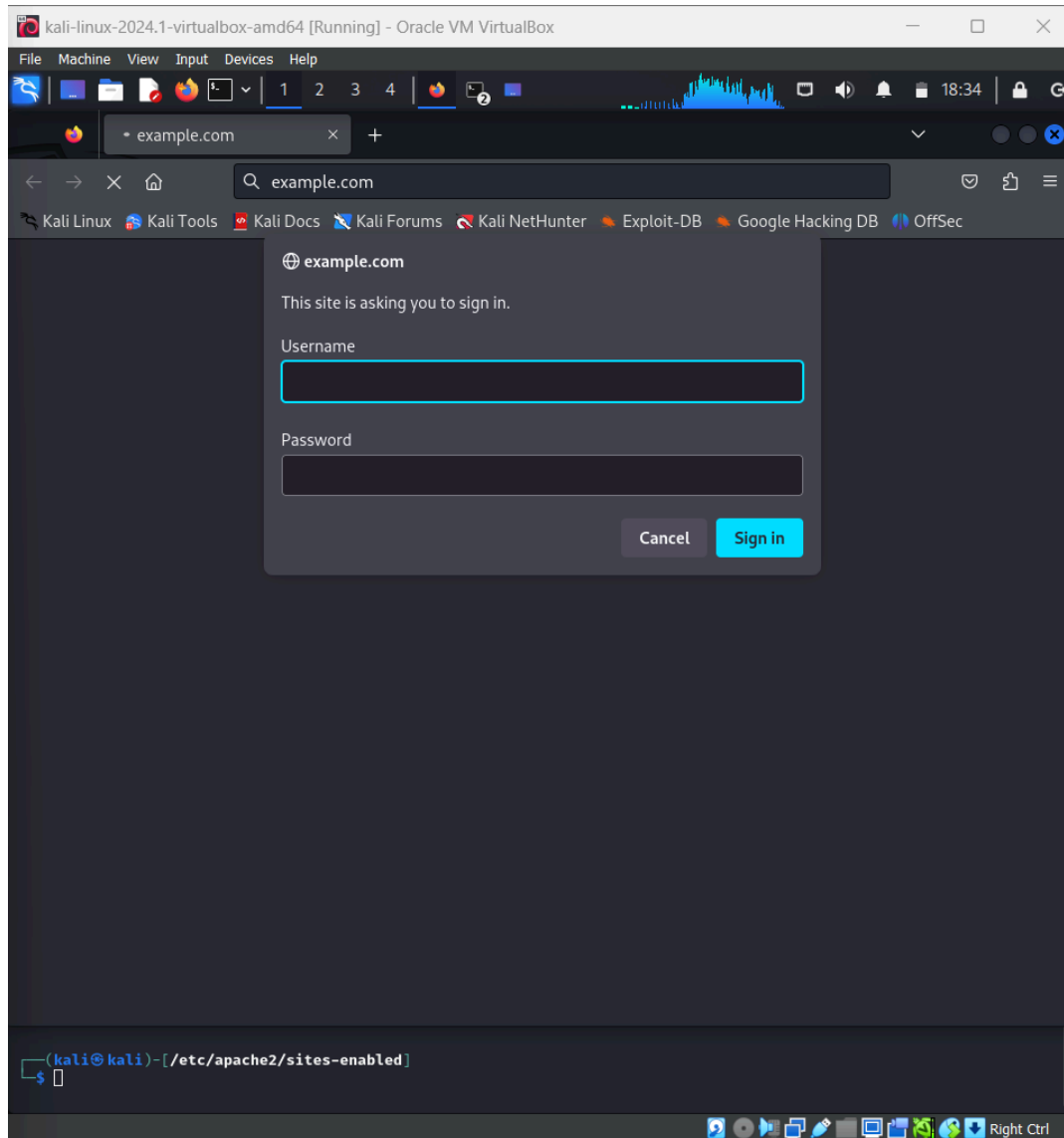
Step 9: Next, issue the following commands to enable the mod_authn_dbd module of Apache. This can be done by enabling the following modules:

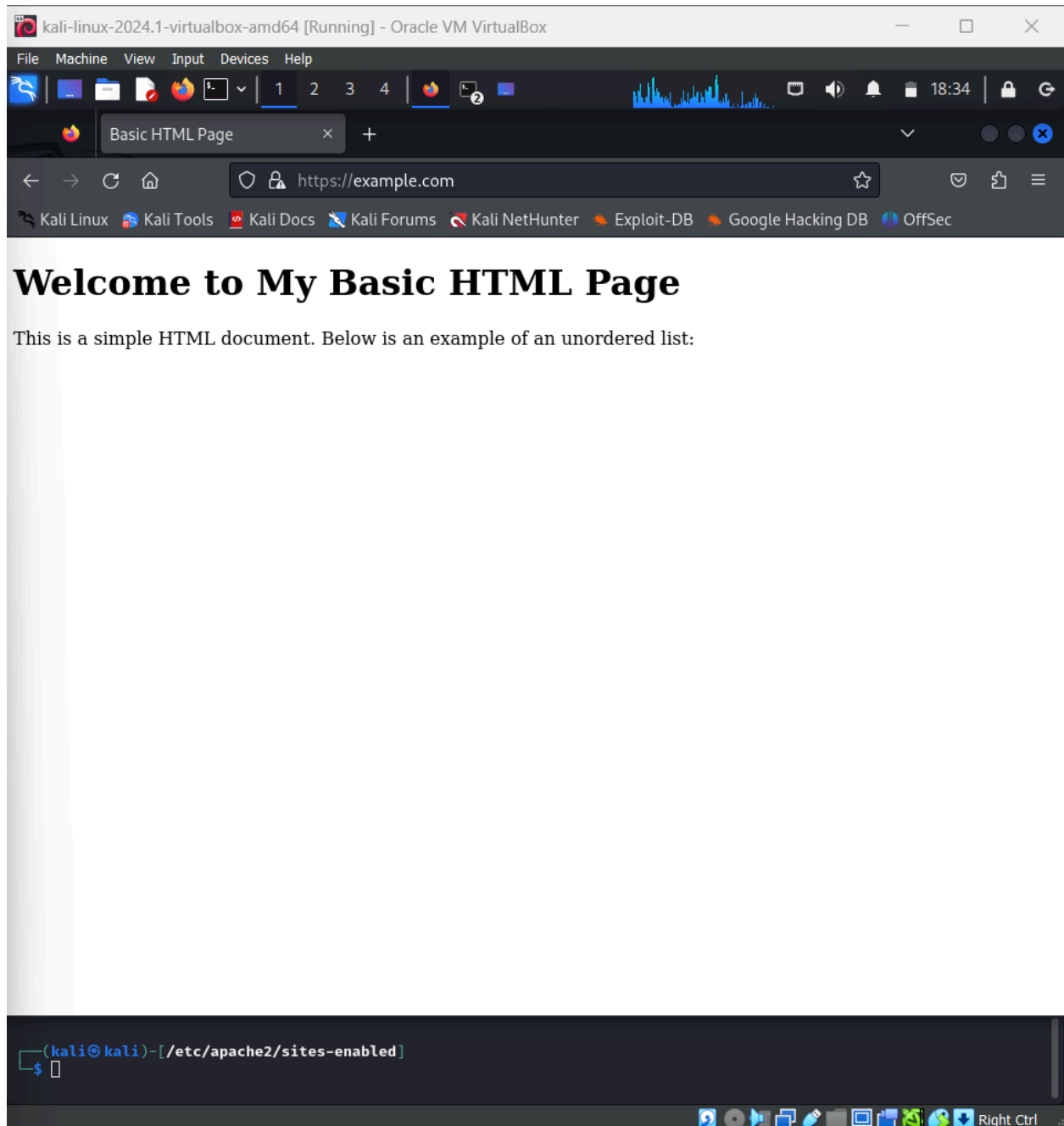
```
sudo a2enmod dbd  
sudo a2enmod authn_dbd  
sudo a2enmod socache_shmcb  
sudo a2enmod authn_socache
```

Now restart the apache server and try to access the website. There will be a username / password prompt, and upon providing correct data, everything works good.



```
kali-linux-2024.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
kali@kali: ~
File Actions Edit View Help
(kali@kali)~]
$ htpasswd -bns alvi kali
alvi:{SHA}5+lx5VrxD3Ey0HgHhexeY3IFCfA=
(kali@kali)~]
$ htpasswd -bns bob pass
bob:{SHA}nU4eI71bcnBGqe00t9tXvY1u5oQ=
(kali@kali)~]
$
(kali@kali)~]
$ sudo a2enmod dbd
[sudo] password for kali:
Enabling module dbd.
To activate the new configuration, you need to run:
systemctl restart apache2
(kali@kali)~]
$ sudo a2enmod authn_dbd
Considering dependency dbd for authn_dbd:
Module dbd already enabled
Enabling module authn_dbd.
To activate the new configuration, you need to run:
systemctl restart apache2
(kali@kali)~]
$ sudo a2enmod socache_shmcb
Module socache_shmcb already enabled
(kali@kali)~]
$ sudo a2enmod authn_socache
Enabling module authn_socache.
To activate the new configuration, you need to run:
systemctl restart apache2
(kali@kali)~]
$ systemctl restart apache2
(kali@kali)~]
$
```





If wrong information is provided, the website can not be accessed and the user is identified as "Unauthorized".

