By Falko Timme
Published: 2008-06-17 18:08

How To Set Up WebDAV With MySQL Authentication On Apache2 (Debian Etch)

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 06/12/2008

This guide explains how to set up WebDAV with MySQL authentication (using mod_auth_mysql) on Apache2 on a Debian Etch server. WebDAV stands for ***Web-based Distributed Authoring and Versioning*** and is a set of extensions to the HTTP protocol that allow users to directly edit files on the Apache server so that they do not need to be downloaded/uploaded via FTP. Of course, WebDAV can also be used to upload and download files.

I do not issue any guarantee that this will work for you!

1 Preliminary Note

I'm using a Debian Etch server with the hostname *`server1.example.com`* and the IP address *`192.168.0.100`* here.

2 Installing Apache2, WebDAV, MySQL, mod_auth_mysql

Unfortunately *`libapache2-mod-auth-mysql`* is available as a Debian package only for Debian Lenny (testing) and Sid (unstable), but not for Etch. Therefore we will install the *`libapache2-mod-auth-mysql`* package from Lenny. To do this, open */etc/apt/sources.list* and add the line *`deb http://ftp2.de.debian.org/debian/ lenny main`*; your */etc/apt/sources.list* could then look like this:

```
vi /etc/apt/sources.list
```

```
deb http://ftp2.de.debian.org/debian/ etch main
deb-src http://ftp2.de.debian.org/debian/ etch main
```

```
deb http://ftp2.de.debian.org/debian/ lenny main

deb http://security.debian.org/ etch/updates main contrib
deb-src http://security.debian.org/ etch/updates main contrib
```

Of course (in order not to mess up our system), we want to install packages from Lenny only if there's no appropriate package from Etch - if there are packages from Etch and Lenny, we want to install the one from Etch. To do this, we give packages from Etch a higher priority in */etc/apt/preferences*:

```
vi /etc/apt/preferences
```

```
Package: *
Pin: release a=etch
Pin-Priority: 700

Package: *
Pin: release a=lenny
Pin-Priority: 650
```

(The terms *etch* and *lenny* refer to the appropriate terms in */etc/apt/sources.list*; if you're using *stable* and *testing* there, you must use *stable* and *testing* instead of *etch* and *lenny* in */etc/apt/preferences* as well.)

Afterwards, we update our packages database:

```
apt-get update
```

If you're getting an error like this:

Segmentation faultsts... 96%

or this one:

E: Dynamic MMap ran out of room

open */etc/apt/apt.conf* and add a line for *APT::Cache-Limit* with a very high value, e.g. like this:

```
vi /etc/apt/apt.conf
```

```
APT::Cache-Limit "100000000";
```

Then run

```
apt-get update
```

again and upgrade the installed packages:

```
apt-get upgrade
```

(If you see any questions, you can accept the default values.)

To install Apache2, WebDAV, MySQL, and mod_auth_mysql, we run:

```
apt-get install apache2 mysql-server mysql-client libapache2-mod-auth-mysql
```

Create a password for the MySQL user root (replace *yourrootsqlpassword* with the password you want to use):

```
mysqladmin -u root password yourrootsqlpassword
```

Then check with

```
netstat -tap | grep mysql
```

on which addresses MySQL is listening. If the output looks like this:

```
tcp        0      0 localhost.localdo:mysql *:*                     LISTEN     2713/mysqld
```

which means MySQL is listening on `localhost.localdomain` only, then you're safe with the password you set before. But if the output looks like this:

```
tcp        0      0 *:mysql *:*                     LISTEN     2713/mysqld
```

you should set a MySQL password for your hostname, too, because otherwise anybody can access your database and modify data:

```
mysqladmin -h server1.example.com -u root password yourrootsqlpassword
```

Afterwards, enable the WebDAV and mod_auth_mysql modules:

```
a2enmod dav_fs

a2enmod dav

a2enmod auth_mysql
```

Reload Apache:

```
/etc/init.d/apache2 force-reload
```

3 Creating A Virtual Host

I will now create a default Apache vhost in the directory */var/www/web1/web*. For this purpose, I will modify the default Apache vhost configuration in */etc/apache2/sites-available/default*. If you already have a vhost for which you'd like to enable WebDAV, you must adjust this tutorial to your situation.

First, we create the directory */var/www/web1/web* and make the Apache user (*www-data*) the owner of that directory:

```
mkdir -p /var/www/web1/web

chown www-data /var/www/web1/web
```

Then we back up the default Apache vhost configuration (*/etc/apache2/sites-available/default*) and create our own one:

```
mv /etc/apache2/sites-available/default /etc/apache2/sites-available/default_orig

vi /etc/apache2/sites-available/default
```

```
NameVirtualHost *
<VirtualHost *>
    ServerAdmin webmaster@localhost

    DocumentRoot /var/www/web1/web/
    <Directory /var/www/web1/web/>
        Options Indexes MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
```

```
</VirtualHost>
```

Then reload Apache:

```
/etc/init.d/apache2 reload
```

4 Configure The Virtual Host For WebDAV

You can find the documentation for mod_auth_mysql in the */usr/share/doc/libapache2-mod-auth-mysql* directory. To read it, you have to gunzip the *DIRECTIVES.gz* and *USAGE.gz* files:

```
cd /usr/share/doc/libapache2-mod-auth-mysql

gunzip DIRECTIVES.gz

vi DIRECTIVES
```

```
gunzip USAGE.gz

vi USAGE
```

Having read these two files, we create a MySQL database called *webdav* in which we will create the table *mysql_auth* which will contain our users and passwords. In addition to that we create the MySQL user *webdav_admin* - this user will be used by mod_auth_mysql to connect to MySQL later on:

```
mysqladmin -u root -p create webdav
```

```
mysql -u root -p
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON webdav.* TO 'webdav_admin'@'localhost' IDENTIFIED BY 'webdav_admin_password';

GRANT SELECT, INSERT, UPDATE, DELETE ON webdav.* TO 'webdav_admin'@'localhost.localdomain' IDENTIFIED BY 'webdav_admin_password';

FLUSH PRIVILEGES;
```

(Replace *webdav_admin_password* with a password of your choice.)

```
USE webdav;
```

```
create table mysql_auth (

username char(25) not null,

passwd char(32),

groups char(25),

primary key (username)

);
```

(Of course, you can as well use existing tables holding your user credentials, and you can as well have additional fields in the table, such as a field that defines if a user is active or not, for example.)

Now we insert the user *test* into our *mysql_auth* table with the password *test* (MD5 encrypted); this user belongs to the group *testgroup*:

```
INSERT INTO `mysql_auth` (`username`, `passwd`, `groups`) VALUES('test', MD5('test'), 'testgroup');
```

We will later on use the URL *http://192.168.0.100/webdav* to connect to WebDAV. When you do this on a Windows XP client and type in the user name *test*, Windows translates this to *192.168.0.100test*. Therefore we create a second user account now:

```
INSERT INTO `mysql_auth` (`username`, `passwd`, `groups`) VALUES('192.168.0.100\\test', MD5('test'), 'testgroup');
```

(We must use a second backslash here in the user name to escape the first one!)

You don't have to do this if you specify the port in the WebDAV URL, e.g. *http://192.168.0.100:**80***/webdav - in this case Windows will simply look for the user *test*, not *192.168.0.100test*.

Then we leave the MySQL shell:

```
quit;
```

Now we modify our vhost in */etc/apache2/sites-available/default* and add the following lines to it:

```
vi /etc/apache2/sites-available/default
```

```
[...]
Alias /webdav /var/www/web1/web
<Location /webdav>
DAV On
AuthBasicAuthoritative Off
AuthUserFile /dev/null
AuthMySQL On
AuthName "webdav"
AuthType Basic
Auth_MySQL_Host localhost
Auth_MySQL_User webdav_admin
```

```
Auth_MySQL_Password webdav_admin_password
AuthMySQL_DB webdav
AuthMySQL_Password_Table mysql_auth
Auth_MySQL_Username_Field username
Auth_MySQL_Password_Field passwd
Auth_MySQL_Empty_Passwords Off
Auth_MySQL_Encryption_Types PHP_MD5
Auth_MySQL_Authoritative On
require valid-user
</Location>
[...]
```

The *Alias* directive makes (together with *<Location>*) that when you call */webdav*, WebDAV is invoked, but you can still access the whole document root of the vhost. All other URLs of that vhost are still "normal" HTTP.

The *AuthBasicAuthoritative Off* and *AuthUserFile /dev/null* are there to prevent that you get errors like these ones in your Apache error log (*/var/log/apache2/error.log*):

```
[Wed Jun 11 17:02:45 2008] [error] Internal error: pcfg_openfile() called with NULL filename
  [Wed Jun 11 17:02:45 2008] [error] [client 127.0.0.1] (9)Bad file descriptor: Could not open password file: (null)
```

If you have additional fields in your MySQL table that define if a user is allowed to log in or not (e.g. a field called *active*), you can add the *Auth_MySQL_Password_Clause* directive, e.g.:

```
[...]
Auth_MySQL_Password_Clause " AND active=1"
[...]
```

(It is important that the string within the quotation marks begins with a space!)

The *require valid-user* directive makes that each user listed in the *mysql_auth* table can log in as long as he/she provides the correct password. If you only want certain users to be allowed to log in, you'd use something like

```
[...]
require user jane joe
[...]
```

instead. And if you only want members of certain groups to be allowed to log in, you'd use something like this:

```
[...]
require group testgroup
[...]
```

The final vhost should look like this:

```
NameVirtualHost *
<VirtualHost *>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/web1/web/
        <Directory /var/www/web1/web/>
                Options Indexes MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
        </Directory>

Alias /webdav /var/www/web1/web
```

```
<Location /webdav>
DAV On
AuthBasicAuthoritative Off
AuthUserFile /dev/null
AuthMySQL On
AuthName "webdav"
AuthType Basic
Auth_MySQL_Host localhost
Auth_MySQL_User webdav_admin
Auth_MySQL_Password webdav_admin_password
AuthMySQL_DB webdav
AuthMySQL_Password_Table mysql_auth
Auth_MySQL_Username_Field username
Auth_MySQL_Password_Field passwd
Auth_MySQL_Empty_Passwords Off
Auth_MySQL_Encryption_Types PHP_MD5
Auth_MySQL_Authoritative On
require valid-user
</Location>
</VirtualHost>
```

Reload Apache afterwards:

```
/etc/init.d/apache2 reload
```

5 Testing WebDAV

We will now install *cadaver*, a command-line WebDAV client:

```
apt-get install cadaver
```

To test if WebDAV works, type:

```
cadaver http://localhost/webdav/
```

You should be prompted for a user name. Type in *test* and then the password for the user *test*. If all goes well, you should be granted access which means WebDAV is working ok. Type *quit* to leave the WebDAV shell:

```
server1:~# cadaver http://localhost/webdav/
  Authentication required for webdav on server `localhost':
  Username: test
  Password:
  dav:/webdav/> quit
  Connection to `localhost' closed.
server1:~#
```

Now test again with the username *192.168.0.100test* (this is the format that Windows XP needs):

```
cadaver http://localhost/webdav/
```

```
server1:~# cadaver http://localhost/webdav/
  Authentication required for webdav on server `localhost':
  Username: 192.168.0.100test
  Password:
  dav:/webdav/> quit
  Connection to `localhost' closed.
server1:~#
```

6 Configure A Windows XP Client To Connect To The WebDAV Share

This is described on ***http://www.howtoforge.com/setting-up-webdav-with-apache2-on-debian-etch-p2***.

If Windows keeps asking and asking about the username and password, you should specify the port in the WebDAV URL, e.g. *http://192.168.0.100:80* /webdav (see chapter four).

7 Configure A Linux Client (GNOME) To Connect To The WebDAV Share

This is described on ***http://www.howtoforge.com/setting-up-webdav-with-apache2-on-debian-etch-p3***.

8 Troubleshooting

It's a good idea to watch the Apache error log (*/var/log/apache2/error.log*) while you're trying to connect to WebDAV, e.g. with this command:

```
tail -f /var/log/apache2/error.log
```

If you get an error like this:

```
[Wed Jun 11 15:39:04 2008] [error] [client 192.168.0.46] (13)Permission denied: Could not open property database. [500, #1]
```

this means that */var/lock/apache2* is not owned by the Apache user (*www-data* on Debian). You can fix this problem by running:

```
chown www-data /var/lock/apache2
```

If Windows keeps asking and asking about the username and password, you should specify the port in the WebDAV URL, e.g. *http://192.168.0.100:80* /webdav (see chapter four).

9 Links

- WebDAV: ***http://www.webdav.org***
- Apache: ***http://httpd.apache.org***
- Debian: ***http://www.debian.org***
- mod_auth_mysql: ***http://modauthmysql.sourceforge.net***