



# MySQL 5.1 Reference Manual :: A Problems and Common Errors :: A.2 Common Errors When Using MySQL Programs :: A.2.3 Client does not support authentication protocol

## Search the MySQL manual:

Manual 5.1

Go

### [MySQL 5.1 Reference Manual](#)

#### [A.2 Common Errors When Using MySQL Programs](#)

##### [A.2.1 Access denied](#)

##### [A.2.2 Can't connect to \[local\] MySQL server](#)

##### [A.2.3 Client does not support authentication protocol](#)

##### [A.2.4 Password Fails When Entered Interactively](#)

##### [A.2.5 Host 'host\\_name' is blocked](#)

##### [A.2.6 Too many connections](#)

##### [A.2.7 Out of memory](#)

##### [A.2.8 MySQL server has gone away](#)

##### [A.2.9 Packet too large](#)

##### [A.2.10 Communication Errors and Aborted Connections](#)

##### [A.2.11 The table is full](#)

##### [A.2.12 Can't create/write to file](#)

##### [A.2.13 Commands out of sync](#)

##### [A.2.14 Ignoring user](#)

##### [A.2.15 Table 'tbl\\_name' doesn't exist](#)

##### [A.2.16 Can't initialize character set](#)

##### [A.2.17 File Not Found](#)

[Get the MySQL Language Reference and MySQL Administrator's Guide from MySQL Press!](#)

Learn about new MySQL releases, technical articles, events and more.

Subscribe to the monthly MySQL Newsletter!

sakila@example.com

Subscribe

## A.2.3. Client does not support authentication protocol

MySQL 5.1 uses an authentication protocol based on a password hashing algorithm that is incompatible with that used by older (pre-4.1) clients. If you upgrade the server from 4.1, attempts to connect to it with an older client may fail with the following message:

```
shell> mysql
Client does not support authentication protocol requested
by server; consider upgrading MySQL client
```

To solve this problem, you should use one of the following approaches:

- Upgrade all client programs to use a 4.1.1 or newer client library.
- When connecting to the server with a pre-4.1 client program, use an account that still has a pre-4.1-style password.
- Reset the password to pre-4.1 style for each user that needs to use a pre-4.1 client program. This can be done using the `SET PASSWORD` statement and the `OLD_PASSWORD()` function:

```
mysql> SET PASSWORD FOR
-> 'some_user'@'some_host' = OLD_PASSWORD('newpwd');
```

Alternatively, use `UPDATE` and `FLUSH PRIVILEGES`:

```
mysql> UPDATE mysql.user SET Password = OLD_PASSWORD('newpwd')
-> WHERE Host = 'some_host' AND User = 'some_user';
mysql> FLUSH PRIVILEGES;
```

Substitute the password you want to use for "newpwd" in the preceding examples. MySQL cannot tell you what the original password was, so you'll need to pick a new one.

- Tell the server to use the older password hashing algorithm:
  - Start `mysqld` with the `--old-passwords` option.
  - Assign an old-format password to each account that has had its password updated to the longer 4.1 format. You can identify these accounts with the following query:

```
mysql> SELECT Host, User, Password FROM mysql.user
-> WHERE LENGTH>Password) > 16;
```

For each account record displayed by the query, use the `Host` and `User` values and assign a password using the `OLD_PASSWORD()` function and either `SET PASSWORD` or `UPDATE`, as described earlier.

**Note:** In older versions of PHP, the `mysql` extension does not support the authentication protocol in MySQL 4.1.1 and higher. This is true regardless of the PHP version being used. If you wish to use the `mysql` extension with MySQL 4.1 or newer, you may need to follow one of the options discussed above for configuring MySQL to work with old clients. The `mysql_i` extension (stands for "MySQL, Improved"; added in PHP 5) is compatible with the improved password hashing employed in MySQL 4.1 and higher, and no special configuration of MySQL need be done in order to use this MySQL client library. For more information about the `mysql_i` extension, see [http://php.net/mysql\\_i](http://php.net/mysql_i).

For additional background on password hashing and authentication, see [Section 5.7.9 "Password Hashing in MySQL 4.1"](#)