LDAP Authentication Plugin

For MySQL Community Server 5.5.7

Charalampos Serenis

LDAP Authentication module for MySQL Community Server 5.5.7

by Charalampos Serenis

Table of Contents

Foreword

User management is one of the most common administrative tasks. Organizations are dependent on multiple resources that must be strictly regulated. Complex authorization rules may apply, but in the end security is always dependent on the way users manage their credentials. Complex authentication mechanisms create problems to users, that will always try to simplify their lives not necessarily in a secure manner. Therefore, user authentication must as transparent as possible, and account re-usage is a necessity within an organization. In modern computer environments, user information is usually maintained in directory services, which are fast, lightweight and provide a central point for storing user information. MySQL Server is a very common SQL engine today and adding support for authentication through directory services can solve numerous problems for system and database administrators.

Introduction

Since version 5.5.7, MySQL Community Server supports user authentication through plugins. This opens a new era in way database user management is performed. The plugins can have their very own business logic to authenticate the user and can even masquerade it's username. This is an important step, compared to built-in methods that authenticated the user against table entries in mysql.user. Not being able to reuse accounts created a hustle for system administrators, since they where required to regularly synchronize database users with directory services.

The motivation

In Infoscope's development servers we needed to provide our developers access to a MySQL server. The developers already had an LDAP account that was used to authenticate and authorize them to use organization resources. Our obvious thought was to configure MySQL to authenticate users against the same LDAP directory. Searching for available options we found out that no free LDAP authentication plugin existed. The only option was to use the commercial extension PAM authentication plugin. Even then, access to LDAP was provided indirectly through PAM. This created inconveniences, since PAM should know about the users. We did not want our developers to have system access to the system, nor did we wish to allocate uids for them. Although, we could configure PAM not to allow system login for these users, this seamed as an overkill both configuration wise and security wise. So the decision was made to write a new plugin for MySQL Community Server, enabling authentication against an LDAP directory and release it to the wild as a GPL software.

Plugin auth_ldap for MySQL Community Server

Auth_ldap is an authentication plugin, for MySQL Community Server 5.5.7 that allows database administrators to reuse user accounts stored in an LDAP directory. The plugin accesses the LDAP server using openLDAP library. Special care has been taken to allow compilation of the plugin at any point of system installation, without the need to recompile the entire MySQL server, or alter any official code. The plugin is self contained and does not pollute the namespace with any symbols. Configuration is done at compile time.

Prerequisites

Auth_ldap is an authentication plugin for MySQL Community Server 5.5.7. Plugable authentication is supported by MySQL Community Server from version 5.5.7 and up. Any attempt to compile this module against MySQL source code older than that will fail. Furthermore, in order to support LDAP connectivity the plugin requires openLDAP library dynamically compiled. The plugin was tested to compile against libopenldap version 2.4.23 and up. It is unknown if it compiles with older versions.

Configuration

Before compiling the plugin you need to configure it. Unfortunately, the plugin doesn't support dynamic configuration through a configuration file and therefore all its options are statically compiled in the plugin. Future releases, will provide a configuration file, but for the time being you will have to set the correct options by editing the config.h file located inside the src directory.

AUTH_LDAP_URI

AUTH_LDAP_URI is passed verbatim to the ldap_initialize function. The URI has the form: schema://hostname:port. If other fields are present, the behavior is undefined.

Schema

Apart from ldap, other (non-standard) recognized values of the schema field are ldaps (LDAP over TLS), ldapi (LDAP over IPC), and cldap (connectionless LDAP).

Hostname

The hostname on which the LDAP server is running. The host parameter may contain a blank-separated list of hosts to try to connect to, and each host may optionally be of the form host:port.

Port

The port number to which to connect. The port parameter is optional. If it is not present the default LDAP port (389) will be used. However, since the default LDAP port can be changed when compiling openLDAP, it is highly recommended that you always specify the port number.

For more information regarding the URI please check the ldap_initialize function man page:
http://linux.die.net/man/3/ldap_initialize

AUTH_LDAP_BASE_DN

The LDAP server base DN used for searches and macro expansions.

AUTH_LDAP_OPENLDAP_SO

The location of the openLDAP dynamic library (.so). Failing to provide the plugin with a correct path will led to errors during loading and a non functional plugin. Please, provide a full path to the library.

AUTH_LDAP_SOCKET

Posible values: 0,1

When AUTH_LDAP_SOCKET is enabled (set to 1) the plugin will only allow users to connect through a UNIX domain socket. Off curse, this affects only users that will authenticate with the auth_ldap module. If set to 1 please also see AUTH_LDAP_LOCAL option.

AUTH_LDAP_LOCAL

Possible values: 0,1

This option has no affect if the AUTH_LDAP_SOCKET is not enabled. When AUTH_LDAP_LOCAL is enabled (set to 1) the plugin will only allow users to connect through a UNIX domain socket or localhost. Off curse, this affects only users that will authenticate with the auth_ldap module. If set to 1 please also see AUTH_LDAP_LOCAL option.

AUTH_LDAP_DN_PATTERN

A pattern used to specify the user's DN for binding. The pattern enables you to use predefined macros that will be expanded in order to create the user's DN. The provided macros are:

Macro %u

The user's username as specified in the client application

Macro %i

When the username has the form: username@domain.tld the %i macro expands to the username part of the email. If the username does not contain the @ character %i expands the same as %u

Macro %d

When the username has the form: username@domain.tld the %d macro expands to the domain part of the email. If the username does not contain the @ character %d expands to a zero string.

Macro %b

Expands to the base DN as specified by AUTH_LDAP_BASE_DN

AUTH_LDAP_ERROR_LEVEL

Error level reporting through the system logs. Possible values:

AUTH_LDAP_ERROR
AUTH_LDAP_INFO
AUTH_LDAP_DEBUG
AUTH_LDAP_DEVEL

AUTH_LDAP_ERROR

Only report errors that are critical (fatal) for the plugin, and invalid login attempts.

AUTH_LDAP_INFO

Log informational messages such as successful plugin loading with minimal footprint on system logs. Error messages continue to be logged. This setting is recommended for production systems.

AUTH_LDAP_DEBUG

Log debugging messages about plugin operations. This creates a fairly large amount of logging data. This setting should not be used on production systems. On the other hand, if you wish to evaluate plugin's operation compiling with debug messages enabled is encouraged.

AUTH_LDAP_DEVEL

!Attention!

This setting should never be used in production systems. It creates security issues by logging memory addresses, user credentials etc. It is only used by developers.

Other configuration macros

The file config.h contains a few more configuration macros. These, are reserved for future use, and have not been implemented yet. Setting, any value to them doesn't have any impact on plugin functionality.

Compiling

After configuring the plugin you can compile it by typing `make' in the parent directory of the source. The build system will start with some basic tests about openldap availability, API and will continue compiling the module. This will create the auth_ldap.so plugin. The first time you compile the plugin it is recommended that you set the error logging level at least to AUTH_LDAP_INFO, in order to verify that the plugin loads correctly.

Installing

Type `make install' as a superuser to install the plugin. After, you installed your plugin you must add a configuration directive your my.cnf in order to instruct you server to load the plugin during initialization. Edit your my.cnf and add the following line:

plugin-load=auth_ldap.so

under the [mysqld] section. You now must restart your MySQL server for the plugin to load. If you check your system logs you should see the following lines, indicating that the plugin has loaded successfully:

Oct 14 04:48:58 infoscope myauth_ldap[21516]: info: loading plugin auth_ldap...
Oct 14 04:48:58 infoscope myauth_ldap[21516]: info: Copyright: Infoscope Hellas, L.P.
Oct 14 04:48:58 infoscope myauth_ldap[21516]: info: plugin auth_ldap loaded successfully

In order to see the above lines you must configure auth_ldap module with at least AUTH_LDAP_INFO error logging.

Troubleshooting

Compilation errors

/bin/sh: mysql_config: not found

If you are getting this error message, followed by a lot of compiler errors this means that the build system is unable to locate the mysql_config program. Under normal conditions this should be installed in the system paths. First of all make sure you have MySQL installed in your system. Then check if you can access mysql_config your self from the console. If you can access it, then there is a problem with the system paths during compilation. Please, find out the full path to your config program by typing which mysql_config and the use the following command for compiling the plugin:

make MYSQL_CONFIG=/usr/local/mysql/bin/mysql_config

If you cannot access mysql_config from console then you will have to locate it manually and give its path to the make program, by using the forementioned command. Please, note that if some Linux distributions have seperate packages for runtime libraries and development files. Just because you have installed MySQL to your system doesn't mean you have the necessary development files installed. In Debian and Ubuntu you should install libmysqlclient-dev package.

error: mysql/plugin_auth.h: No such file or directory

If you do not see a mysql_config: not found error and you are getting this error, this probably means that you are trying to compile the plugin against an old verion of MySQL. The plugin requires at least version 5.5.7 to compile. You can check your MySQL version by typing

mysql_config --version

in the console.

error: ldap.h: No such file or directory

You do not have openLDAP library installed, or it is not installed in system standard paths. If you need to specify the path the library header files are installed set the CFLAGS value when invoking make. e.g.:

make CFLAGS=-I/usr/local/include

Creating SQL users

After you have successfully installed and loaded the module, you must create your database users. Unfortunately, there is not way to automatically load username from the LDAP directory. Furthermore, the plugin is called only when a user specified to authenticate with the module tries to

log in. All other users are left unaffected.

In order to create a user named bob, authenticated against the LDAP directory, log in to your SQL server with an account having administrative privileges and execute the following SQL command:

CREATE USER 'bob'@'localhost' IDENTIFIED WITH auth_ldap;

From now on, whenever user `bob' tries to login from localhost the auth_ldap module is going to be invoked to handle authentication. You can write a custom script to synchronize your LDAP users and MySQL users and execute it via cron. Although this is sub-optimal, it is still viable since you do not need to provide MySQL with a password for the user. Therefore, your users can reuse their passwords. For more information about MySQL plugable authentication please read MySQL reference manual:

http://dev.mysql.com/doc/refman/5.5/en/pluggable-authentication.html

Make auth_ldap the default

Creating users using the IDENTIFIED WITH clause is mandatory for the time being. MySQL doesn't support any means to configure the default authentication plugin for the server. Scraping source code files we found in file sql/sql_acl.cc line 180:

```
/// @todo make it configurable
LEX_STRING *default_auth_plugin_name= &native_password_plugin_name;
```

So you can see that the plugin name is hard coded in the source code. Changing this to "auth_ldap" and recompiling the entire server might do the trick, but it hasn't been tested.

Copyright

Disclaimer

SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.