ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ, ИНФОРМАТИЗАЦИЙ И ТЕЛЕКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

КУРСОВАЯ РАБОТА

«СЗБД»

ТЕМА: “Архитектура системы безапасности в MySQL server.”

Выполнил:Гр 232-10АХу
Аббосов Х.

Принял(а): Гулямов Ш.

Ташкент 2013

СОДЕРЖАНИЕ

Введение 2
ГЛАВА 1 ОПИСАНИЕ ЕЕ РАБОТЫ MYSQL 3
1.1. Общие принципы обеспечения безопасности 3
1.2. Как обезопасить MySQL от хакеров 7
1.3. Опции запуска mysqld, относящиеся к безопасности 10
1.4. Вопросы безопасности, относящиеся к команде LOAD DATA ….13
ГЛАВА 2 ОСНОВНЫЕ ФУНКЦИИ В СИСТЕМЕ MYSQL 14
2.1.Как работает система привилегий 14
2.2.Привилегии, предоставляемые MySQL 19
2.3.Синтаксис команд GRANT и REVOKE 24
2.4.Имена пользователей MySQL и пароли 32
Заключение 35
Список литературы 36

Введение

С данным разделом должны ознакомиться все, кто использует MySQL на компьютерах, подключенных к Internet, - чтобы избежать наиболее распространенных ошибок, приводящих к нарушению безопасности системы.

При обсуждении вопросов безопасности мы акцентируем внимание на необходимости защиты всего серверного хоста (а не одного лишь сервера MySQL) от всех возможных типов атак: перехвата, внесения изменений, считывания и отказа в обслуживании. Данный раздел не охватывает всех аспектов готовности к работе и отказоустойчивости.

Используемая в MySQL система безопасности для всех подключений, запросов и иных операций, которые может пытаться выполнить пользователь, базируется на списках контроля доступа ACLs (Access Control Lists). Обеспечивается также некоторая поддержка SSL-соединений между клиентами и серверами MySQL.

Многие из рассматриваемых здесь концепций не относятся исключительно к MySQL; те же общие соображения применимы практически ко всем приложениям.

ГЛАВА 1. ОПИСАНИЕ ЕЕ РАБОТЫ MYSQL

Общие проблемы безопасности и система привилегий доступа MySQL

MySQL имеет развитую, но нестандартную систему обеспечения безопасности и привилегий доступа. В этом разделе дается описание ее работы.

1.1.Общие принципы обеспечения безопасности

Не предоставляйте никому (за исключением пользователя mysql под именем root) доступа к таблице user в базе данных mysql! Это чрезвычайно важно. В MySQL зашифрованный пароль является реальным паролем. Узнав пароль, занесенный в таблицу user, и имея доступ к удаленному компьютеру, занесенному в соответствующую учетную запись, **войти в систему под именем зарегистрированного владельца пароля легко может кто угодно.**

Изучите систему прав доступа MySQL. Для управления доступом к MySQL служат команды GRANT и REVOKE. Предоставляйте ровно столько прав, сколько необходимо, и не больше. Никогда не предоставляйте права всем хостам. Полезно проводить следующие контрольные проверки:

Выполните команду mysql -u root. Если удается успешно установить соединение с сервером без получения запроса пароля, значит, у вас имеются проблемы. Это означает, что кто угодно может подсоединиться к вашему серверу MySQL как клиент MySQL под именем root, получая таким образом право неограниченного доступа! Проанализируйте инструкцию по инсталляции MySQL, обращая особое внимание на ту часть, которая касается задания пароля пользователя root.

С помощью команды SHOW GRANTS проверьте, кто и к каким ресурсам имеет доступ. Воспользуйтесь командой REVOKE, отмените права доступа, которые не являются необходимыми.

Не храните в базе данных незашифрованных паролей. Если злоумышленнику удастся получить доступ на ваш компьютер, то в его руках окажется полный список паролей, которыми он может воспользоваться. Применяйте для шифрования MD5(), SHA1() или другие односторонние хеш-функции.

Не используйте в качестве пароля слова из словарей. Для взлома такого рода паролей имеются специальные программы. Даже слова типа ``xfish98'' - это очень плохие пароли. Куда лучше ``duag98'': здесь используется то же слово ``fish'', но при этом буквы в нем заменены ближайшими к ним слева буквами клавиатуры QWERTY. Еще один метод - составить парольное слово из первых букв слов какого либо словосочетания, например ``Mhall'' - по фразе ``Mary had a little lamb.'' Такой пароль легко запоминается и его легко вводить. А вот разгадать его тому, кто не знает ключевой фразы, будет непросто.

Приобретите брандмауэр. Эта мера обеспечит защиту как минимум от половины всех видов несанкционированного использования любого ПО, с которым вы работаете. Разместите MySQL за брандмауэром или в демилитаризованной зоне (demilitarised zone - DMZ). Полезно проводить следующие контрольные проверки:

Попробуйте просканировать ваши порты из Internet с помощью утилиты типа nmap. MySQL использует по умолчанию порт 3306. Этот порт должен быть недоступен с неблагонадежных компьютеров. Еще один простой способ проверить, открыт или нет ваш MySQL-порт, - попытаться выполнить с какой либо удаленной машины следующую команду, где server_host - имя хоста, на котором установлен ваш сервер MySQL:

shell> telnet server_host 3306

Если соединение будет установлено, и вы получите какие-либо бессмысленные символы, это будет означать, что порт открыт, и его нужно закрыть на брандмауэре или маршрутизаторе (если, конечно, нет

действительно веских причин держать его открытым). Если же telnet просто зависнет или в подсоединении будет отказано, тогда все в порядке: порт заблокирован.

Не доверяйте никаким данным, которые вводят пользователи. Возможны попытки перехитрить вашу программу путем ввода последовательностей специальных или экранированных символов в веб-формы, URL-ы или любое приложение, созданное вами. Убедитесь, что защита вашего приложения не будет нарушена, если пользователь введет что-нибудь типа "'; DROP DATABASE mysql;". Это крайний случай, но действия хакеров, использующих подобную технологию, могут привести к потере информации и появлению брешей в системе безопасности, если вы не готовы к ним. Не следует также забывать о необходимости проверки цифровых данных (распространенной ошибкой является защита только строк). Некоторые полагают, что если в базе данных хранятся только открытые данные, то в ее защите нет необходимости. Это неверно. Такие базы могут стать объектом успешных атак типа отказа от обслуживания. Простейший способ защиты от взломов такого типа - заключать числовые константы в кавычки: SELECT * FROM table WHERE ID='234', а не SELECT * FROM table WHERE ID=234. MySQL автоматически преобразует эту строку в число и выбросит из нее все нецифровые символы. Полезно проводить следующие контрольные проверки:

Для всех веб-приложений:

Попробуйте ввести во все ваши веб-формы одинарные и двойные кавычки - `" и `"". Если MySQL выдаст любое сообщение об ошибке, немедленно разберитесь, в чем дело.

Попробуйте видоизменять динамические URL, добавляя в них %22 (`""), %23 (`#'), и %27 (`").

Попробуйте модифицировать типы данных в динамических URL - замените числовые на символьные, используя символы из предыдущих

примеров. Ваше приложение должно быть устойчиво к подобного рода атакам.

Попробуйте вводить в числовые поля вместо цифр буквы, пробелы и специальные символы. Ваше приложение должно либо удалять их до передачи в MySQL, либо выдавать сообщение об ошибке. Пропускать в MySQL значения без проверки очень опасно!

Проверяйте размер данных перед тем, как они будут переданы в MySQL.

При подключении приложения к базе данных лучше использовать имя пользователя, отличное от того, которое вы используете для целей администрирования. Не предоставляйте своим приложениям больше прав доступа, чем это необходимо.

Пользователям PHP:

Проверьте функцию addslashes(). Что касается PHP 4.0.3, то в нем имеется функция mysql_escape_string(), базирующаяся на функции с тем же именем из MySQL C API.

Пользователям MySQL C API:

Проверьте API-вызов mysql_real_escape_string().

Пользователям MySQL++:

Проверьте такие модификаторы, как escape и quote, - для потоков запросов.

Пользователям Perl DBI:

Проверьте метод quote() или используйте для проверки заполнители.

Пользователям Java JDBC:

Используйте для проверки объект PreparedStatement и символы-заполнители.

Не передавайте по Internet открытые (незашифрованные) данные. Они могут оказаться у кого угодно, имеющего достаточно времени и возможностей для того, чтобы перехватить их и использовать в своих целях.

Используйте вместо этого протоколы с шифрованием данных, такие как SSL и SSH. MySQL, начиная с версии 4.0.0, поддерживает собственные SSL-соединения. Пересылка по SSH (SSH Port Forwarding) может быть использована для создания туннеля передачи данных с шифрованием и сжатием.

Научитесь пользоваться утилитами tcpdump и strings. В большинстве случаев проверить, являются ли потоки данных MySQL зашифрованными, можно с помощью команды, подобной той, которая приведена ниже:

shell> tcpdump -l -i eth0 -w - src or dst port 3306 | strings

(Она работает под Linux, и будет, с незначительными изменениями, работать под другими системами.) Предупреждение: если вы не видите данных, это еще не гарантирует того, что они зашифрованы. Если требуется высокий уровень безопасности, обратитесь к экспертам в этой области.

1.2.Как обезопасить MySQL от хакеров

При подключении к серверу MySQL используется, как правило, пароль. По линии связи пароль не передается в виде открытого текста, но алгоритм шифрования не очень сложный. Толковый хакер, если ему удастся перехватить трафик между клиентом и сервером, при определенной настойчивости может взломать пароль. Поэтому если связь между клиентом и сервером осуществляется по ненадежной сети, для шифрования связи следует использовать SSH-туннель.

Вся остальная информация передается в текстовом виде и может быть прочитана кем угодно, кто в состоянии отлеживать подключение. Если это вас беспокоит, можно воспользоваться протоколом со сжатием данных (в MySQL 3.22 и последующих версиях), что значительно затруднит подобные действия. Чтобы еще более повысить безопасность связи, следует использовать протокол ssh. Open source-клиент ssh доступен на веб-сайте http://www.openssh.org/, а коммерческий ssh-клиент можно получить на веб-сайте http://www.ssh.com/. С помощью такого протокола можно обеспечить

зашифрованную связь по протоколу TCP/IP между сервером MySQL и клиентом MySQL.

Если вы используете MySQL 4.0, то можете также использовать предусмотренную в этой версии поддержку протокола OpenSSL. Обратитесь к разделу See section 4.3.9 Использование безопасных соединений.

Для обеспечения безопасности MySQL-системы необходимо строго придерживаться следующих рекомендаций:

У всех пользователей MySQL должны быть пароли. Для приложений клиент/сервер является общепринятым, что клиент может указывать любое имя пользователя, но если для other_user не задан пароль, то кто угодно может зайти под любым именем, просто введя mysql -u other_user db_name. Чтобы этого избежать, можно изменить пароль для всех пользователей, отредактировав скрипт mysql_install_db перед запуском приложения, или только пароль для root-пользователя MySQL, как это показано ниже:

shell> mysql -u root mysql

mysql> UPDATE user SET Password=PASSWORD('new_password')

mysql> FLUSH PRIVILEGES;

Не запускайте демон MySQL от имени пользователя Unix root. Это очень опасно, потому что любой пользователь, имеющий привилегию FILE, будет в состоянии создавать файлы как пользователь root (например ~root/.bashrc). Чтобы предотвратить это, mysqld откажется запускаться от имени пользователя root, если это не будет задано напрямую с помощью опции --user=root. В то же время mysqld может быть запущена от имени обычного непривилегированного пользователя. Можно также, в целях еще большего укрепления безопасности, создать новый аккаунт Unix-пользователя mysql. При запуске mysqld от имени другого пользователя Unix у вас отпадает необходимость заменять имя пользователя root в таблице user, так как имена пользователя в MySQL не имеют ничего общего с аккаунтами пользователей Unix. Для запуска mysqld от имени другого пользователя Unix

добавьте в группу [mysqld] файла опций `/etc/my.cnf' или файла опций `my.cnf', находящегося в каталоге данных сервера, строку user, задающую имя пользователя. Например:

[mysqld]

user=mysql

В результате сервер будет запущен от имени назначенного пользователя, независимо от того, производится запуск вручную или посредством safe_mysqld или mysql.server. Для получения дополнительной информации обратитесь к разделу See section A.3.2 Запуск MySQL от обычного пользователя.

Откажитесь от поддержки символических ссылок на таблицы (ее можно запретить с помощью опции --skip-symlink). Это особенно важно в том случае, если вы запускаете mysqld от имени пользователя root, поскольку у того, кто имеет право доступа для записи в каталоги данных mysqld, появляется возможность стереть любой файл в системе! Обратитесь к разделу See section 5.6.1.2 Использование символических ссылок для таблиц.

Удостоверьтесь, что пользователь Unix, от имени которого запускается mysqld, является единственным пользователем, имеющим привилегии чтения/записи в директории базы данных.

Не предоставляйте привилегии PROCESS всем пользователям. Команда mysqladmin processlist выводит текст запросов, обрабатываемых в данный момент. Следовательно, любой пользователь, имеющий право на выполнение этой команды, получает возможность прочитать, например, такой запрос другого пользователя, как UPDATE user SET password=PASSWORD('not_secure'). mysqld резервирует добавочное подключение для пользователей, имеющих привилегию PROCESS, так что пользователь MySQL под именем root может подключиться и осуществлять контроль даже в том случае, когда все обычные подключения заняты.

Не предоставляйте привилегии FILE всем пользователям. Любой пользователь, имеющий такую привилегию, может записать в любом месте файловой системы файл с привилегиями демона mysqld! Чтобы обеспечить здесь хоть минимальную защиту, все файлы создаваемые с помощью команды SELECT ... INTO OUTFILE, сделаны общедоступными для записи, но перезаписать существующие файлы нельзя. Привилегия FILE может быть также использована для чтения любого файла, доступного пользователю Unix, от имени которого запускается сервер. Можно также прочитать любой файл в текущую базу данных. Это может быть использовано в корыстных целях. Возможно, например, с помощью команды LOAD DATA загрузить `/etc/passwd' в таблицу и прочесть ее позже с помощью SELECT.

Если вы не доверяете своему DNS-серверу, используйте в таблицах привилегий вместо имен хостов IP-адреса. В любом случае следует очень осторожно относиться к внесению в таблицы привилегий записей, в которых значения имени хоста содержат шаблонные символы!

Чтобы ограничить число подключений, доступных для отдельного пользователя, можно в mysqld задать значение переменной max_user_connections.

1.3.Опции запуска mysqld, относящиеся к безопасности

К безопасности имеют отношение следующие опции mysqld:

--local-infile[=(0|1)]

При установке опции --local-infile=0 теряется возможность выполнять команду LOAD DATA LOCAL INFILE.

--safe-show-database

Если установлена эта опция, команда SHOW DATABASES возвращает только те базы данных, для которых пользователь имеет какую-либо привилегию. Начиная с версии 4.0.2 эта опция отменена и не служит ни для чего (она включена по умолчанию), т.к. сейчас у нас имеется привилегия

SHOW DATABASES. Обратитесь к разделу See section 4.3.1 Синтаксис команд GRANT и REVOKE.

--safe-user-create

При установке этой опции пользователь не может создавать новых пользователей с помощью команды GRANT, если у него отсутствует привилегия INSERT для таблицы mysql.user. Чтобы предоставить пользователю доступ именно для создания новых пользователей с теми привилегиями, которые он имеет право предоставлять, для этого пользователя следует установить следующую привилегию:

mysql> GRANT INSERT(user) ON mysql.user TO 'user'&hacute;ostname';

Задание такой привилегии гарантирует, что этот пользователь не сможет непосредственно вносить изменения ни в одном из столбцов привилегий, а для предоставления привилегий другим пользователям должен будет использовать команду GRANT.

--skip-grant-tables

Установка этой опции запрещает серверу вообще использовать систему привилегий. Это открывает кому бы то ни было полный доступ ко всем базам данных! (После запуска сервера можно заставить его снова использовать таблицы привилегий с помощью команды mysqladmin flush-privileges или mysqladmin reload.)

--skip-name-resolve

При установке данной опции имена хостов не разрешены. Все значения в столбцах Host таблиц привилегий должны быть либо IP-адресами, либо localhost.

--skip-networking

Не разрешает осуществлять подсоединений по протоколу TCP/IP через сеть (данная опция запрещает такие подсоединения). Все подсоединения к mysqld должны осуществляться посредством сокетов Unix. Для MySQL старше 3.23.27 эта опция непригодна для систем, в которых используются

MIT-потоки, так как MIT-потоки на тот момент не поддерживали сокеты Unix.

--skip-show-database

Разрешает выполнение команды SHOW DATABASES только в том случае, если пользователь имеет привилегию SHOW DATABASES. Начиная с версии 4.0.2 в этой опции больше нет необходимости, т.к. теперь доступ может предоставляться избирательно с помощью привилегии SHOW DATABASES.

1.4. Вопросы безопасности, относящиеся к команде LOAD DATA LOCAL

Чтобы решить проблемы безопасности, которые могут возникнуть при использовании команды LOAD DATA LOCAL,в MySQL 3.23.49 и MySQL 4.0.2 были добавлены новые опции.

При поддержке этой команды могут возникнуть две проблемы:

Первая: поскольку чтение файла инициируется сервером, теоретически имеется возможность создать ``доработанный" при помощи патча сервер MySQL, способный читать любые файлы на клиентской машине, к которой текущий пользователь имеет доступ для чтения, в то время, когда клиент направляет запрос к таблице.

Вторая: в веб-среде, в которой подсоединение клиентов осуществляется с веб-сервера, пользователь может использовать команду LOAD DATA LOCAL для чтения любых файлов, к которым процесс веб-сервера имеет доступ для чтения (если предположить, что пользователь может выполнять на сервере SQL любые команды).

Эти проблемы решаются с помощью двух следующих исправлений:

Если вы конфигурируете MySQL без опции --enable-local-infile, то команда LOAD DATA LOCAL будет запрещена для всех клиентов, если, конечно, они не будут вызывать mysql_options (... MYSQL_OPT_LOCAL_INFILE, 0). Обратитесь к разделу See section 8.4.3.39 mysql_options().

В случае клиента mysql, LOAD DATA LOCAL может быть разблокирована заданием опции --local-infile[=1] или заблокирована с помощью опции --local-infile=0.

По умолчанию все MySQL-клиенты и библиотеки компилируются с опцией --enable-local-infile для обеспечения совместимости с MySQL 3.23.48 и более старыми версиями.

Блокировку всех команд LOAD DATA LOCAL на MySQL-сервере можно осуществить путем запуска mysqld с опцией --local-infile=0.

В случае, если команда LOAD DATA LOCAL INFILE заблокирована на сервере или клиенте, вы получите следующее сообщение об ошибке (1148): The used command is not allowed with this MySQL version.

ГЛАВА 2. ОСНОВНЫЕ ФУНКЦИИ В СИСТЕМЕ MYSQL

Основной функцией системы привилегий MySQL является аутентификация пользователя, подключающегося с указанного хоста, и ассоциирование его с привилегиями базы данных, такими как SELECT, INSERT, UPDATE и DELETE.

К дополнительным функциональным возможностям системы привилегий относятся следующие: возможность обслуживания анонимного пользователя и предоставление привилегий для таких специфических для MySQL функций, как LOAD DATA INFILE и функции администрирования.

При работе в MySQL старайтесь следовать приведенным ниже инструкциям:

2.1.Как работает система привилегий

Система привилегий MySQL обеспечивает пользователям возможность выполнять только те действия, которые им разрешены в соответствии с их обязанностями. Когда вы подсоединяетесь к серверу MySQL, ваша личность устанавливается по имени **хоста, с которого вы подсоединяетесь**, и **имени пользователя, которое вы указываете**. Система предоставляет привилегии в соответствии с вашей личностью и **тем, что вы хотите делать.**

MySQL идентифицирует пользователя как по имени хоста, так и по имени пользователя, т.к. нет оснований полагать что данное имя пользователя принадлежит во всей Сети единственному человеку. Например, пользователь joe, устанавливающий соединение с office.com, вовсе не обязательно один и тот же человек, что и пользователь joe, подсоединяющийся с elsewhere.com. MySQL решает эту проблему, обеспечивая возможность различать пользователей, подсоединяющихся с различных хостов под одним и тем же именем пользователя: вы можете предоставлять joe один набор привилегий, если он подсоединяется с

office.com, и другой набор привилегий, если joe подсоединяется с elsewhere.com.

Управление доступом в MySQL осуществляется в два этапа:

Этап 1: сервер проверяет, имеется ли у вас вообще разрешение на подсоединение. 2item Этап 2: если таковое имеется, сервер проверяет каждый из ваших запросов, чтобы убедиться в том, что у вас имеется достаточно привилегий для его выполнения. Например, если вы пытаетесь выбрать строки в таблице базы данных или удалить таблицу из базы данных, сервер в первом случае проверяет, имеется ли у вас для этой таблицы привилегия SELECT, а во втором - имеется ли у вас для этой базы данных привилегия DROP.

На обеих стадиях управления доступом сервер использует таблицы user, db и host из базы данных mysql. Ниже перечислены поля этих таблиц привилегий:

Имя таблицы	user	db	host
Поля контекста	Host	Host	Host
	User	Db	Db
	Password	User	
Поля привилегий	Select_priv	Select_priv	Select_priv
	Insert_priv	Insert_priv	Insert_priv
	Update_priv	Update_priv	Update_priv
	Delete_priv	Delete_priv	Delete_priv
	Index_priv	Index_priv	Index_priv
	Alter_priv	Alter_priv	Alter_priv
	Create_priv	Create_priv	Create_priv
	Drop_priv	Drop_priv	Drop_priv
	Grant_priv	Grant_priv	Grant_priv
	References_priv		
	Reload_priv		

	Shutdown_priv		
	Process_priv		
	File_priv		

На втором этапе управления доступом (верификация запросов) сервер может (в случае, если запрос относится к таблицам базы данных) дополнительно обратиться к таблицам tables_priv и columns_priv. Поля этих таблиц привилегий перечислены ниже:

Имя таблицы	tables_priv	columns_priv
Поля контекста	Host	Host
	Db	Db
	User	User
	Table_name	Table_name
		Column_name
Поля привилегий	Table_priv	Column_priv
	Column_priv	
Иные поля	Timestamp	Timestamp
	Grantor	

Каждая таблица привилегий включает в себя поля контекста и поля привилегий.

Поля контекста определяют область действия каждой из записей в таблицах, т.е. контекст, к которому имеет отношение та или иная запись. Например, запись в таблице user, в полях Host и User которой указаны значения 'thomas.loc.gov' 'bob', предназначена для аутентификации подсоединений к серверу с хоста thomas.loc.gov под именем пользователя bob. Аналогично запись в таблице db, в полях Host, User и Db которой указаны значения 'thomas.loc.gov', 'bob' и 'reports', будет использоваться при попытке пользователя по именем bob подсоединиться с хоста thomas.loc.gov и получить доступ к базе данных reports. В полях контекста в таблицах

tables_priv и columns_priv указаны таблицы или комбинации таблиц/столбцов, к которым применяется каждая запись.

При контроле доступа сравнение значений в полях Host осуществляется без учета регистра. Значения в полях User, Password, Db и Table_name также являются независимыми от регистра символов. Значения в поле Column_name являются независимыми от регистра символов, начиная с MySQL 3.22.12.

В полях привилегий указываются привилегии, предоставляемые записью в таблице, т.е. то, какие операции разрешено выполнять. Сервер формирует полное описание привилегий пользователя, комбинируя информацию, хранящуюся в разных таблицах привилегий. Это осуществляется по правилам, которые описаны в разделе See section 4.2.10 Управление доступом, этап 2: верификация запросов.

Поля контекста - это строковые значения, объявленные, как показано ниже; устанавливаемым по умолчанию значением для каждого из них является пустая строка:

Имя поля	**Тип**	**Примечания**
Host	CHAR(60)	
User	CHAR(16)	
Password	CHAR(16)	
Db	CHAR(64)	(CHAR(60) для таблиц tables_priv и columns_priv tables)
Table_name	CHAR(60)	
Column_name	CHAR(60)	

В таблицах user, db и host все поля привилегий имеют объявленный тип ENUM('N','Y'), т.е. возможно одно из двух значений - 'N' и 'Y', а устанавливаемым по умолчанию является 'N'.

В таблицах tables_priv and columns_priv поля привилегий объявляются как SET:

Имя таблицы	**Имя поля**	**Допустимые элементы набора**
tables_priv	Table_priv	'Select', 'Insert', 'Update', 'Delete', 'Create', 'Drop', 'Grant', 'References', 'Index', 'Alter'
tables_priv	Column_priv	'Select', 'Insert', 'Update', 'References'
columns_priv	Column_priv	'Select', 'Insert', 'Update', 'References'

Если кратко, то сервер использует таблицы привилегий следующим образом:

Поля контекста таблицы user определяют, разрешить входящее подсоединение или отказать в нем. Для разрешенных подсоединений любые привилегии, предоставленные в таблице user, означают глобальные привилегии пользователя (привилегии суперпользователя). Эти привилегии распространяются на все базы данных, размещенные на сервере.

Таблицы db и host используются совместно:

Поля контекста таблицы db определяют, каким пользователям, при подсоединении с каких хостов разрешен доступ к каким базам данных. Поля привилегий определяют разрешенные операции.

Таблица host используется в качестве расширения таблицы db в случае, если необходимо применить некоторую запись из таблицы db к разным хостам. Например, если вы хотите предоставить пользователю возможность обращаться к базе данных с различных хостов сети, оставьте пустым поле в записи этого пользователя в таблице db, а затем внесите в таблицу host запись для каждого из этих хостов. Более подробно данный механизм описан в разделе See section 4.2.10 Управление доступом, этап 2: верификация запросов.

Таблицы tables_priv и columns_priv подобны таблице db, но областью их действия является уже уровень таблиц и столбцов, а не баз данных.

Заметим, что привилегии администрирования (RELOAD, SHUTDOWN и т.д..) задаются только в таблице user. Это связано с тем, что операции

администрирования являются операциями над самим сервером, а не над базами данных, поэтому не смысла перечислять такие привилегии в других таблицах привилегий. Фактически для того, чтобы выяснить, имеете ли вы привилегии выполнять операции администрирования, достаточно обратиться только к таблице user.

Привилегия FILE также задается только в таблице user. Она не является привилегией администрирования как таковой, но возможность производить чтение или запись файлов на серверном хосте не связана с базой данных, к которой вы получаете доступ.

Сервер mysqld считывает содержимое таблиц привилегий единожды, при его запуске. О том, каким образом изменения, вносимые в таблицы привилегий, вступают в силу, рассказывается в разделе See section 4.3.3 Когда изменения в привилегиях вступают в силу.

При внесении изменений в таблицы привилегий стоит убедиться в том, что ваши изменения задают привилегии именно так, как задумано вами. Помощь по диагностике проблем вы найдете в разделе See section 4.2.11 Причины появления ошибок Access denied ("в доступе отказано"). По вопросам, связанным с безопасностью, следует обращаться к разделу See section 4.2.2 Как обезопасить MySQL от хакеров.

Полезным диагностическим инструментом является скрипт mysqlaccess, которым Ив Карлье (Yves Carlier) укомплектовал дистрибутив MySQL. Запустите mysqlaccess с опцией --help чтобы посмотреть, как он работает. Заметим, что mysqlaccess контролирует доступ, используя только таблицы user, db и host. Он не проверяет привилегии на уровне таблиц или столбцов.

2.2.Привилегии, предоставляемые MySQL

Информация о привилегиях пользователя хранится в таблицах user, db, host, tables_priv и columns_priv базы данных mysql (т.е. в базе данных с именем mysql). Сервер MySQL считывает содержимое этих таблиц во время

запуска, и в случаях, указанных в разделе See section 4.3.3 Когда изменения в привилегиях вступают в силу.

Ниже приведены имена, используемые в данном руководстве для ссылок на привилегии, предоставляемые в MySQL 4.0.2. Здесь же указаны имена табличных столбцов, ассоциированных с каждой из привилегий в таблицах привилегий, наряду с контекстом, в котором эти привилегии имеют силу:

Привилегия	**Столбец**	**Контекст**
ALTER	Alter_priv	таблицы
DELETE	Delete_priv	таблицы
INDEX	Index_priv	таблицы
INSERT	Insert_priv	таблицы
SELECT	Select_priv	таблицы
UPDATE	Update_priv	таблицы
CREATE	Create_priv	базы данных, таблицы или индексы
DROP	Drop_priv	базы данных или таблицы
GRANT	Grant_priv	базы данных или таблицы
REFERENCES	References_priv	базы данных или таблицы
CREATE TEMPORARY TABLES	Create_tmp_table_priv	администрирование сервера
EXECUTE	Execute_priv	администрирование сервера
FILE	File_priv	доступ к файлам на сервере
LOCK TABLES	Lock_tables_priv	администрирование сервера
PROCESS	Process_priv	администрирование сервера
RELOAD	Reload_priv	администрирование сервера
REPLICATION CLIENT	Repl_client_priv	администрирование сервера
REPLICATION SLAVE	Repl_slave_priv	администрирование сервера

SHOW DATABASES	Show_db_priv	администрирование сервера
SHUTDOWN	Shutdown_priv	администрирование сервера
SUPER	Super_priv	администрирование сервера

Привилегии SELECT, INSERT, UPDATE и DELETE позволяют выполнять операции над строками таблиц баз данных.

Для операторов SELECT привилегия SELECT требуется только в том случае, если они действительно извлекают строки из таблицы.. В ряде случае можно выполнять операторы SELECT, даже не имея разрешения на доступ ни к одной базе данных на сервере. Например: клиент mysql вы можете использовать в качестве обычного калькулятора:

mysql> SELECT 1+1;

mysql> SELECT PI()*2;

Привилегия INDEX позволяет создавать или уничтожать (удалять) индексы.

Привилегия ALTER позволяет использовать команду ALTER TABLE.

Привилегии CREATE и DROP позволяют создавать новые базы данных и таблицы или уничтожать (удалять) существующие базы данных и таблицы.

Заметим, что в случае, если пользователю предоставляется привилегия DROP по отношению к базе данных mysql, он может уничтожить базу данных, в которой хранятся привилегии доступа в MySQL!

Привилегия GRANT позволяет вам предоставлять другим пользователям привилегии, которыми обладаете вы сами.

Привилегия FILE дает вам право читать и записывать файлы на сервере с помощью операторов LOAD DATA INFILE и SELECT ... INTO OUTFILE. Любой пользователь, которому предоставлена такая привилегия, имеет право прочитать или записать любой файл, который может прочитать или записать сервер MySQL. Пользователь также может прочитать любой файл в каталоге текущей базы данных. Однако существующие файлы перезаписывать нельзя.

Остальные привилегии используются для операций администрирования, выполняемых с помощью программы mysqladmin. В таблице, приведенной ниже, для каждой из привилегий администрирования показано, какие команды mysqladmin она позволяет выполнять.

Привилегия	**Команды, разрешенные обладателю привилегии**
RELOAD	reload, refresh, flush-privileges, flush-hosts, flush-logs, and flush-tables
SHUTDOWN	Shutdown
PROCESS	Processlist
SUPER	Kill

Команда reload заставляет сервер перечитать таблицы привилегий. Команда refresh очищает все таблицы, а также открывает и закрывает файлы журналов. flush-privileges является синонимом reload. Остальные команды flush-* выполняют функции, аналогичные функциям команды refresh, но с более узкой областью действия. В некоторых случаях такие команды могут оказаться более предпочтительными. Например, если вы хотите очистить только системные журналы, команда flush-logs лучше, чем refresh.

Команда shutdown завершает работу сервера.

Команда processlist выводит информацию о задачах, выполняющихся на сервере. Команда kill уничтожает серверные потоки. Собственные потоки всегда можно просмотреть или уничтожить, но для отображения потоков, запущенных другими пользователями, нужна привилегия PROCESS,а для уничтожения потоков, запущенных другими пользователями, потребуется привилегия SUPER. Обратитесь к разделу See section 4.5.5 Синтаксис команды KILL.

В общем случае идея предоставлять привилегии только тем пользователям, которым они необходимы, хорошая, но к предоставлению некоторых из них следует относиться особенно внимательно:

Привилегия GRANT позволяет пользователям передавать свои привилегии другим пользователям. Два пользователя с неодинаковыми привилегиями, имея привилегию GRANT, способны объединить свои привилегии.

Привилегия ALTER может быть использована для переименования таблиц и разрушения таким образом всей системы привилегий.

Привилегия FILE может использоваться злонамеренно для считывания любого доступного файла, хранящегося на сервере, или любого файла в каталоге текущей базы данных в таблицу, к содержимому которой можно затем получить доступ с помощью команды SELECT.

Привилегия SHUTDOWN может использоваться злонамеренно для полного прекращения работы сервера и, таким образом, полного запрещения обслуживания других пользователей.

Привилегия PROCESS может быть использована для просмотра открытого текста запросов выполняющихся в данный момент, включая запросы на установку или изменение паролей.

Привилегии доступа к базе данных mysql могут быть использованы для изменения паролей и другой информации, относящейся к привилегиям доступа. (Пароли хранятся в зашифрованном виде, поэтому злоумышленник не сможет просто прочесть их, чтобы получить пароли в виде обычного текста). Получив доступ к столбцу паролей mysql.user, любой пользователь может войти на сервер MySQL под именем другого пользователя (имея достаточные привилегии, тот же самый пользователь может заменить пароль на другой).

Есть вещи, которые система привилегий MySQL делать не может:

Нельзя явно указать, что данному пользователю должен быть закрыт доступ. Т.е. вы не можете явно выбрать пользователя и затем отказать ему в подключении.

Нельзя указать, что некий пользователь имеет привилегии создавать или удалять таблицы в базе данных, но не имеет привилегий создавать или удалять саму базу данных.

2.3.Синтаксис команд GRANT и REVOKE

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)] ...]
   ON {tbl_name | * | *.* | db_name.*}
   TO user_name [IDENTIFIED BY [PASSWORD] 'password']
      [, user_name [IDENTIFIED BY 'password'] ...]
   [REQUIRE
         NONE |
      [{SSL| X509}]
      [CIPHER cipher [AND]]
      [ISSUER issuer [AND]]
      [SUBJECT subject]]
   [WITH [GRANT OPTION | MAX_QUERIES_PER_HOUR # |
                  MAX_UPDATES_PER_HOUR # |
                  MAX_CONNECTIONS_PER_HOUR #]]

REVOKE priv_type [(column_list)] [, priv_type [(column_list)] ...]
   ON {tbl_name | * | *.* | db_name.*}
   FROM user_name [, user_name ...]
```

GRANT включен в MySQL начиная с версии 3.22.11 и выше. В более ранних версиях MySQL оператор GRANT ничего не выполняет.

Команды GRANT и REVOKE позволяют системным администраторам создавать пользователей MySQL, а также предоставлять права пользователям или лишать их прав на четырех уровнях привилегий:

Глобальный уровень

Глобальные привилегии применяются ко всем базам данных на указанном сервере. Эти привилегии хранятся в таблице mysql.user.

Уровень базы данных

Привилегии базы данных применяются ко всем таблицам указанной базы данных. Эти привилегии хранятся в таблицах mysql.db и mysql.host.

Уровень таблицы

Привилегии таблицы применяются ко всем столбцам указанной таблицы. Эти привилегии хранятся в таблице mysql.tables_priv.

Уровень столбца

Привилегии столбца применяются к отдельным столбцам указанной таблицы. Эти привилегии хранятся в таблице mysql.columns_priv.

Если привилегии предоставляются пользователю, которого не существует, то этот пользователь создается. Чтобы просмотреть примеры работы команды GRANT, см. раздел section 4.3.5 Добавление новых пользователей в MySQL.

В таблице приведен список возможных значений параметра priv_type для операторов GRANT и REVOKE:

ALL [PRIVILEGES]	Задает все простые привилегии, кроме WITH GRANT OPTION
ALTER	Разрешает использование ALTER TABLE
CREATE	Разрешает использование CREATE TABLE
CREATE TEMPORARY TABLES	Разрешает использование CREATE TEMPORARY TABLE
DELETE	Разрешает использование DELETE
DROP	Разрешает использование DROP TABLE
EXECUTE	Разрешает пользователю запускать хранимые процедуры (для MySQL 5.0)
FILE	Разрешает использование SELECT ... INTO OUTFILE и LOAD DATA INFILE
INDEX	Разрешает использование CREATE INDEX and DROP

	INDEX
INSERT	Разрешает использование INSERT
LOCK TABLES	Разрешает использование LOCK TABLES на таблицах, для которых есть привилегия SELECT
PROCESS	Разрешает использование SHOW FULL PROCESSLIST
REFERENCES	Зарезервировано для использования в будущем
RELOAD	Разрешает использование FLUSH
REPLICATION CLIENT	Предоставляет пользователю право запрашивать местонахождение головного и подчиненных серверов
REPLICATION SLAVE	Необходимо для подчиненных серверов при репликации (для чтения информации из бинарных журналов головного сервера)
SELECT	Разрешает использование SELECT
SHOW DATABASES	SHOW DATABASES выводит все базы данных
SHUTDOWN	Разрешает использование mysqladmin shutdown
SUPER	Позволяет установить одно соединение (один раз), даже если достигнуто значение max_connections, и запускать команды CHANGE MASTER, KILL thread, mysqladmin debug, PURGE MASTER LOGS и SET GLOBAL
UPDATE	Разрешает использование UPDATE
USAGE	Синоним для ``без привилегий''
GRANT OPTION	Синоним для WITH GRANT OPTION

Значение USAGE можно задавать, если необходимо создать пользователя без привилегий.

Привилегии CREATE TEMPORARY TABLES, EXECUTE, LOCK TABLES, REPLICATION ..., SHOW DATABASES и SUPER являются новыми для версии 4.0.2. Чтобы воспользоваться этими новыми привилегиями после

обновления до версии 4.0.2, необходимо запустить скрипт mysql_fix_privilege_tables.

В боле старых версиях MySQL привилегия PROCESS предоставляет такие же права, как и новая привилегия SUPER.

Чтобы лишить пользователя привилегий, предоставленных командой GRANT, воспользуйтесь значением priv_type в GRANT OPTION:

mysql> REVOKE GRANT OPTION ON ... FROM ...;

Для таблицы можно указать только следующие значения priv_type: SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, GRANT OPTION, INDEX и ALTER.

Для столбца можно указать только следующие значения priv_type (при использовании оператора column_list): SELECT, INSERT и UPDATE.

Глобальные привилегии можно задать, воспользовавшись синтаксисом ON *.*, а привилегии базы данных - при помощи синтаксиса ON db_name.*. Если указать ON * при открытой текущей базе данных, то привилегии будут заданы для этой базы данных. (**Предупреждение**: если указать ON * при **отсутствии** открытой текущей базы данных, это повлияет на глобальные привилегии!)

Заметьте: шаблонные символы `_' и `%' не допускаются в определении имени баз данных в операторе GRANT. Это означает, что если вы хотите использовать, скажем, символ `_' в имени базы данных, то вы должны указать его как `_' в GRANT, чтобы пользователь не имел возможности получить доступ к другим базам данных, соответствующих шаблону: GRANT ... ON `foo\_bar`.* TO ....

С тем, чтобы можно было определять права пользователям с конкретных компьютеров, в MySQL обеспечивается возможность указывать имя пользователя (user_name) в форме user@host. Если необходимо указать строку user, в которой содержатся специальные символы (такие как `-') или строку host, в которой содержатся специальные или групповые символы

(такие как `%'), можно заключить имя удаленного компьютера или пользователя в кавычки (например, 'test-user'@'test-hostname').

В имени удаленного компьютера также можно указывать групповые символы. Например, user@'%.loc.gov' относится к user всех удаленных компьютеров домена loc.gov, а user@'144.155.166.%' относится к user всех удаленных компьютеров подсети 144.155.166 класс C.

Простая форма user является синонимом для user@"%".

В MySQL не поддерживаются групповые символы в именах пользователей. Анонимные пользователи определяются вставкой записей User='' в таблицу mysql.user или созданием пользователя с пустым именем при помощи команды GRANT.

Примечание: если анонимным пользователям разрешается подсоединяться к серверу MySQL, необходимо также предоставить привилегии всем локальным пользователям как user@localhost, поскольку в противном случае при попытке пользователя зайти в MySQL с локального компьютера в таблице mysql.user будет использоваться вход для анонимного пользователя!

Чтобы проверить, происходит ли подобное на вашем компьютере, выполните следующий запрос:

mysql> SELECT Host,User FROM mysql.user WHERE User='';

На данный момент команда GRANT поддерживает имена удаленных компьютеров, таблиц, баз данных и столбцов длиной не более 60 символов. Имя пользователя должно содержать не более 16 символов.

Привилегии для таблицы или столбца формируются при помощи логического оператора OR из привилегий каждого из четырех уровней. Например, если в таблице mysql.user указано, что у пользователя есть глобальная привилегия SELECT, эта привилегия не отменяется на уровне базы данных, таблицы или столбца.

Привилегии для столбца могут быть вычислены следующим образом:

глобальные привилегии

OR (привилегии базы данных AND привилегии удаленного компьютера)

OR привилегии таблицы

OR привилегии столбца

В большинстве случаев права пользователя определяются только на одном уровне привилегий, поэтому обычно эта процедура не настолько сложна, как описано выше. Подробная информация о последовательности действий проверки привилегий представлена в разделе section 4.2 Общие проблемы безопасности и система привилегий доступа MySQL.

Если привилегии предоставляются сочетанию пользователь/удаленный компьютер, которое отсутствует в таблице mysql.user, то в последнюю добавляется запись, которая остается в таблице до тех пор, пока не будет удалена при помощи команды DELETE. Иначе говоря, команда GRANT может создавать записи user в таблице, но команда REVOKE не может их удалить. Это необходимо делать при помощи команды DELETE.

Если в MySQL версий 3.22.12 и выше создан новый пользователь или предоставлены глобальные привилегии, пароль пользователя будет назначаться оператором IDENTIFIED BY, если он указан. Если у пользователя уже есть пароль, то этот пароль будет заменен новым.

Если вы не хотите отправлять пароль открытым текстом, можно воспользоваться параметром PASSWORD с зашифрованным паролем, полученным при помощи функции SQL PASSWORD() или функции C API make_scrambled_password(char *to, const char *password).

Предупреждение: если при создании нового пользователя не указать оператор IDENTIFIED BY, будет создан пользователь без пароля. Это ненадежно с точки зрения безопасности.

Пароли также можно задавать при помощи команды SET PASSWORD. See section 6.2.3.4 Тип множества SET.

Если у вас привилегии для базы данных, то при необходимости в таблице mysql.db создается запись. Данная запись удаляется после удаления всех привилегий для этой базы данных командой REVOKE.

Если у пользователя нет никаких привилегий для таблицы, то таблица не отображается, когда пользователь запрашивает список таблиц (например, при помощи оператора SHOW TABLES).

Оператор WITH GRANT OPTION предоставляет пользователю возможность наделять других пользователей любыми привилегиями, которые он сам имеет на указанном уровне привилегий. При предоставлении привилегии GRANT необходимо проявлять осмотрительность, так как два пользователя с разными привилегиями могут объединить свои привилегии!

Параметры MAX_QUERIES_PER_HOUR #, MAX_UPDATES_PER_HOUR # и MAX_CONNECTIONS_PER_HOUR # являются новыми в MySQL версии 4.0.2. Эти параметры ограничивают количество запросов, обновлений и входов, которые пользователь может осуществить в течение одного часа. Если установлено значение 0 (принято по умолчанию), то это означает, что для данного пользователя нет ограничений. See section 4.3.6 Ограничение ресурсов пользователя.

Внимание: чтобы указать любую из этих опция для существующего пользователя, но не давать никаких дополнительных привилегий, используйте GRANT USAGE ... WITH MAX_....

Нельзя предоставить другому пользователю привилегию, которой нет у вас самого. Привилегия GRANT позволяет предоставлять только те привилегии, которыми вы обладаете.

Учтите, что если пользователю назначена привилегия GRANT на определенном уровне привилегий, то все привилегии, которыми этот пользователь уже обладает (или которые будут ему назначены в будущем!)

на этом уровне, также могут назначаться этим пользователем. Предположим, пользователю назначена привилегия INSERT в базе данных. Если потом в базе данных назначить привилегию SELECT и указать WITH GRANT OPTION, пользователь сможет назначать не только привилегию SELECT, но также и INSERT. Если затем в базе данных предоставить пользователю привилегию UPDATE, пользователь сможет после этого назначать INSERT, SELECT и UPDATE.

Не следует назначать привилегии ALTER обычным пользователям. Это дает пользователю возможность разрушить систему привилегий путем переименования таблиц!

Обратите внимание на то, что если используются привилегии для таблицы или столбца даже для одного пользователя, сервер проверяет привилегии таблиц и столбцов для всех пользователей, и это несколько замедляет работу MySQL.

При запуске mysqld все привилегии считываются в память. Привилегии базы данных, таблицы и столбца вступают в силу немедленно, а привилегии уровня пользователя - при следующем подсоединении пользователя. Изменения в таблицах назначения привилегий, которые осуществляются при помощи команд GRANT и REVOKE, обрабатываются сервером немедленно. Если изменять таблицы назначения привилегий вручную (используя команды INSERT, UPDATE и т.д.), необходимо запустить оператор FLUSH PRIVILEGES или mysqladmin flush-privileges, чтобы указать серверу на необходимость перезагрузки таблиц назначения привилегий. See section 4.3.3 Когда изменения в привилегиях вступают в силу.

Наиболее значительные отличия команды GRANT версий ANSI SQL и MySQL следующие:

В MySQL привилегии назначаются для сочетания имя пользователя + удаленный компьютер, а не только для имени пользователя.

В ANSI SQL отсутствуют глобальные привилегии и привилегии уровня базы данных, и ANSI SQL поддерживает не все типы привилегий MySQL. В свою очередь, в MySQL отсутствует поддержка привилегий ANSI SQL TRIGGER, UNDER.

Структура привилегий ANSI SQL является иерархической. Если удалить пользователя, то все назначенные этому пользователелю привилегии будут отменены. В MySQL назначенные привилегии не отменяются автоматически, их при необходимости требуется удалять самостоятельно.

В MySQL пользователь может применять к таблице оператор INSERT при наличии у него привилегии INSERT только для нескольких столбцов в этой таблице. Столбцы, для которых отсутствует привилегия INSERT, будут установлены в свои значения, принятые по умолчанию. В ANSI SQL требуется наличие привилегии INSERT для всех столбцов.

При удалении таблицы в ANSI SQL все привилегии для этой таблицы будут отменены. Если отменить привилегию в ANSI SQL, то все привилегии, которые были назначены на основе этой привилегии, также будут отменены. В MySQL привилегии могут удаляться только при помощи команды REVOKE или путем изменения таблиц назначения привилегий MySQL.

Чтобы ознакомиться с описанием использования REQUIRE, см. раздел See section 4.3.9 Использование безопасных соединений.

2.4.Имена пользователей MySQL и пароли

Между MySQL и Unix или Windows существует несколько различий в использовании имен пользователей и паролей:

Имена пользователей, которые применяются в MySQL для авторизации, не имеют ничего общего с именами пользователей Unix (аккаунты Unix) или именами пользователей Windows. Большинство клиентов MySQL по умолчанию пытаются войти в систему, используя текущее имя пользователя Unix в качестве имени пользователя MySQL, но это сделано только для удобства. Программы клиентов позволяют указывать

различные имена при помощи параметров -u или --user. Это означает, что невозможно обеспечить безопасность базы данных, если не все имена пользователей MySQL снабжены паролями: ведь можно попытаться подсоединиться к серверу, используя любое имя, а если воспользоваться именем, которому не назначен пароль, то удастся войти в систему.

Имена пользователей MySQL могут содержать до 16 символов. Имена пользователей Unix обычно ограничены 8 символами.

Пароли MySQL не имеют никакого отношения к паролям Unix. Не существует связи между паролем, который используется для входа в Unix, и паролем, необходимым для доступа к базе данных.

MySQL шифрует пароли при помощи своего алгоритма, который отличается от алгоритма Unix, используемого во время входа в систему. Описание функций PASSWORD() и ENCRYPT() можно найти в разделе See section 6.3.6.2 Разные функции. Обратите внимание: даже если ваш пароль хранится в 'зашифрованном виде', то знания этого 'зашифрованного' пароля будет достаточно, чтобы подсоединиться к серверу MySQL!

Пользователи MySQL и их привилегии обычно создаются при помощи команды GRANT. See section 4.3.1 Синтаксис команд GRANT и REVOKE.

Если подсоединение к серверу MySQL осуществляется с клиента командной строки, необходимо указать пароль при помощи параметра --password=your-password. See section 4.2.8 Соединение с сервером MySQL.

mysql --user=monty --password=guess database_name

Если необходимо, чтобы клиент запрашивал пароль, то следует указать --password без каких-либо аргументов

mysql --user=monty --password database_name

или сокращенный вариант этого параметра:

mysql -u monty -p database_name

Обратите внимание на то, что в последнем примере database_name не является паролем.

Заключение

Для управления доступом к MySQL служат команды GRANT и REVOKE. Предоставляйте ровно столько прав, сколько необходимо, и не больше. Никогда не предоставляйте права всем хостам. Полезно проводить следующие контрольные проверки. Не доверяйте никаким данным, которые вводят пользователи. Возможны попытки перехитрить вашу программу путем ввода последовательностей специальных или экранированных символов в веб-формы, URL-ы или любое приложение, созданное вами. Убедитесь, что защита вашего приложения не будет нарушена, если пользователь введет что-нибудь типа "'; DROP DATABASE mysql;". Это крайний случай, но действия хакеров, использующих подобную технологию, могут привести к потере информации и появлению брешей в системе безопасности, если вы не готовы к ним. Не следует также забывать о необходимости проверки цифровых данных (распространенной ошибкой является защита только строк). В общем случае идея предоставлять привилегии только тем пользователям, которым они необходимы, хорошая, но к предоставлению некоторых из них следует относиться особенно внимательно.Привилегия GRANT позволяет пользователям передавать свои привилегии другим пользователям. Два пользователя с неодинаковыми привилегиями, имея привилегию GRANT, способны объединить свои привилегии.

Список литературы

1. Hinz S. , DuBois P. , Stephens J. “ MySQL 5.5 Reference Manual ”.

2. Haines R. “ The SELinux Notebook — The Foundations. 2nd Edition ”.

3. Smalley S. “ Configuring the SELinux Policy ”.

4. Loscocco P. , Smalley S. “ Meeting Critical Security Objectives with Security-Enhanced Linux ” .

5. Использованные сайты :

1. **Russia-diplom.com**
2. **Google.uz**
3. **Ziyo.net**
4. **Librarytuit.uz**
5. **Referat.uz**