Как установить и настроить phpMyAdmin в Ubuntu 20.04

Введение

Данная инструкция рассказывает о процессе установки инструмента phpMyAdmin в операционной системе Ubuntu 20.04. Установку выполняем в облачной инфраструктуре Selectel. Нам потребуется настроенный сервер LAMP. О том, как его развернуть, написали в статье.

PhpMyAdmin — бесплатный инструмент, созданный на языке php, для администрирования MySQL с использованием браузера.

В комплекте — огромный пул возможных операций с MySQL и MariaDB. На данный момент актуальная стабильная версия — phpMyAdmin 5.1.1.

Установка phpMyAdmin

Первый шаг — установка модуля расширения php-mbstring. Mbstring предоставляет функции для работы с многобайтными строками, которые облегчают обработку многобайтовых кодировок в php.

Команда установки:

```
sudo apt install php-mbstring -y
```

Вывод успешной установки:

```
vlan48@apachi:~$ sudo apt install php-mbstring -y
[sudo] password for vlan48:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libonig5 php7.4-mbstring
The following NEW packages will be installed:
  libonig5 php-mbstring php7.4-mbstring
0 upgraded, 3 newly installed, 0 to remove and 27 not upgraded.
Need to get 541 kB of archives.
After this operation, 1709 kB of additional disk space will be used.
Get:1 http://mirror.selectel.ru/ubuntu focal/universe amd64 libonig5 amd64 6.9.4-1 [142 kB]
Get:2 http://mirror.selectel.ru/ubuntu focal-updates/universe amd64 php7.4-mbstring amd64 7.4.3-4ubuntu2.6
[397 kB]
Get:3 http://mirror.selectel.ru/ubuntu focal/universe amd64 php-mbstring all 2:7.4+75 [2012 B]
Fetched 541 kB in 0s (12.1 MB/s)
Selecting previously unselected package libonig5:amd64.
(Reading database ... 45001 files and directories currently installed.)
Preparing to unpack .../libonig5_6.9.4-1_amd64.deb ...
Unpacking libonig5:amd64 (6.9.4-1) ...
Selecting previously unselected package php7.4-mbstring.
Preparing to unpack .../php7.4-mbstring_7.4.3-4ubuntu2.6_amd64.deb ...
Unpacking php7.4-mbstring (7.4.3-4ubuntu2.6) ...
Selecting previously unselected package php-mbstring.
Preparing to unpack .../php-mbstring_2%3a7.4+75_all.deb ...
Unpacking php-mbstring (2:7.4+75) ...
Setting up libonig5:amd64 (6.9.4-1) ...
Setting up php7.4-mbstring (7.4.3-4ubuntu2.6) ...

Creating config file /etc/php/7.4/mods-available/mbstring.ini with new version
Setting up php-mbstring (2:7.4+75) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Processing triggers for libapache2-mod-php7.4 (7.4.3-4ubuntu2.6) ...
Processing triggers for php7.4-cli (7.4.3-4ubuntu2.6) ...
```

Следующим этапом станет инсталляция phpMyAdmin в систему. Выполняется это командой:

```
sudo apt install phpmyadmin -y
```

Обратить внимание на Validate Password

Во время инсталляции LAMP мог быть активирован плагин Validate Password. Это чревато ошибкой при создании пароля для пользователя phpmyadmin. Для деактивации плагина необходимо выполнить следующие шаги:

Открыть консоль управления MySQL:

```
sudo mysql
```

Если активна аутентификация по паролю суперпользователя root, команда будет выглядеть так:

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

Теперь отправляем следующую команду:

```
UNINSTALL COMPONENT "file://component_validate_password";
```

Это действие произведет отключение плагина Validate Password.

Покинем консоль MySQL, команда:

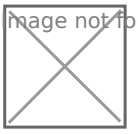
```
exit
```

Можно продолжить установку phpMyAdmin. Эти действия следует производить только в случае возникновения ошибки, во всех остальных случаях в этом нет необходимости.

Продолжим установку. После отправки команды в терминал появится окно установщика, в котором потребуется ответить на ряд вопросов. Навигация в установщике осуществляется посредством использования клавиш Up down, выбор пунктов кнопка «**Пробел**», переход ниже — **Tab.** Ввод выбранного ответа — **Enter**.

На первый вопрос про используемый web-сервер, необходимо ответить — *apache2.*

Следующий вопрос про инсталляцию и настройку служебной базы данных для самой программы phpMyAdmin, отвечаем положительно.



Следующим пунктом будет установлен пароль ранее созданной базы данных для пользователя phpmyadmin:

Подтвердим созданный ранее пароль:

Вывод успешной работы программы:

```
vlan48@apachi:~$ sudo apt -y install phpmyadmin
[sudo] password for vlan48:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  www-browser php-recode php-gd2 php-pragmarx-google2fa php-bacon-qr-code php-samyoul-u2f-php-server
Recommended packages:
```

```
  php-curl php-gd php-bz2 php-zip php-tcpdf
The following NEW packages will be installed:
  phpmyadmin
0 upgraded, 1 newly installed, 0 to remove and 28 not upgraded.
Need to get 4426 kB of archives.
After this operation, 27.2 MB of additional disk space will be used.
Get:1 http://mirror.selectel.ru/ubuntu focal/universe amd64 phpmyadmin all 4:4.9.5+dfsg1-2 [4426 kB]
Fetched 4426 kB in 0s (34.8 MB/s)
Preconfiguring packages ...
Determining localhost credentials from /etc/mysql/debian.cnf: succeeded.
Selecting previously unselected package phpmyadmin.
(Reading database ... 46341 files and directories currently installed.)
Preparing to unpack .../phpmyadmin_4%3a4.9.5+dfsg1-2_all.deb ...
Unpacking phpmyadmin (4:4.9.5+dfsg1-2) ...
Setting up phpmyadmin (4:4.9.5+dfsg1-2) ...
Determining localhost credentials from /etc/mysql/debian.cnf: succeeded.
dbconfig-common: writing config to /etc/dbconfig-common/phpmyadmin.conf

Creating config file /etc/dbconfig-common/phpmyadmin.conf with new version

Creating config file /etc/phpmyadmin/config-db.php with new version
checking privileges on database phpmyadmin for phpmyadmin@localhost: user creation needed.
granting access to database phpmyadmin for phpmyadmin@localhost: success.
verifying access for phpmyadmin@localhost: success.
creating database phpmyadmin: success.
verifying database phpmyadmin exists: success.
populating database via sql...  done.
dbconfig-common: flushing administrative password
apache2_invoke: Enable configuration phpmyadmin
```

Установка завершена. Прежде чем пойти далее, необходимо произвести проверку работоспособности на данном этапе, чтобы в случае возникновения проблем, их можно было решить с наименьшими трудозатратами.

Проверим. В любом браузере открываем phpMyAdmin по IP-адресу сервера:

```
ip_address/phpmyadmin
```

вход в систему под созданной учетной записью

Теперь можно войти в систему с использованием учетной записи, созданной во время установки. Проверяем:

вход в систему под учетной записью с паролем

панель phpMyAdmin после успешной авторизации

Для учетной записи root по умолчанию применяется доступ с использованием плагина auth_socket. Изменим это на аутентификацию с использованием пароля. В данном случае необходимо изменить тип аутентификации на аутентификацию с использованием пароля.

Откроем консоль MySQL:

```
sudo mysql
```

Произведем проверку таблицы пользователей, чтобы увидеть метод аутентификации для каждого пользователя:

```
SELECT user,plugin,host FROM mysql.user;
```

Вывод:

```
+------------------------+-----------------------------------+------------+
| user                   | plugin                            | host       |
+------------------------+-----------------------------------+------------+
| debian-sys-maint       | caching_sha2_password             | localhost  |
| mysql.infoschema       | caching_sha2_password             | localhost  |
| mysql.session          | caching_sha2_password             | localhost  |
| mysql.sys              | caching_sha2_password             | localhost  |
| phpmyadmin             | caching_sha2_password             | localhost  |
| root                   | auth_socket                       | localhost  |
+------------------------+-----------------------------------+------------+
7 rows in set (0.00 sec)
```

Исходя из данных таблицы следует, что аутентификация пользователя root происходит с использованием плагина auth_socket.

Для изменения отправим следующую команду:

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'password';
```

Password добавляем свой.

Вывод работы команды:

```
Query OK, 0 rows affected (0.03 sec)
```

В случае возникновения ошибки **Plugin caching_sha2_password in not loaded** следует обратиться к разделу «Возможные ошибки» в конце статьи.

Проверим результат.

```
SELECT user,plugin,host FROM mysql.user;
```

Вывод:

```
+----------------------+-----------------------------------+-----------+
| user                 | plugin                            | host      |
+----------------------+-----------------------------------+-----------+
| debian-sys-maint     | caching_sha2_password             | localhost |
| mysql.infoschema     | caching_sha2_password             | localhost |
| mysql.session        | caching_sha2_password             | localhost |
| mysql.sys            | caching_sha2_password             | localhost |
| phpmyadmin           | caching_sha2_password             | localhost |
| root                 | caching_sha2_password             | localhost |
+----------------------+-----------------------------------+-----------+
7 rows in set (0.00 sec)
```

Теперь у пользователя root тип аутентификации — caching_sha2_password.

Работа с пользователями

Основная задача в работе с пользователями phpMyAdmin — создание и настройка прав. В качестве примера создадим пользователя с максимальными привилегиями. Необходимо вернуться в терминал под пользователем с административными правами и отправить команду:

```
sudo mysql
```

Так мы запустим работу с базой данных от имени администратора. Потребуется ввод пароля.

Вывод:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 8.0.27-0ubuntu0.20.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Теперь необходимо добавить пользователя, а также его пароль. Выполнить это можно командой:

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
```

где **user** — новый пользователь, а **password** — пароль для этого пользователя.

В рабочем варианте отправленная команда выглядит так:

```
CREATE USER 'selectel'@'localhost' IDENTIFIED BY 's*****qq';
```

Вывод:

```
Query OK, 0 rows affected (0.02 sec)
```

Добавим необходимые привилегии для созданного пользователя. В данном случае это будут все привилегии для всех баз данных на сервере:

```
GRANT ALL PRIVILEGES ON *.* TO 'user'@'localhost';
```

Если пользователю необходимо разрешить создавать пользователей и назначать им привилегии, необходимо добавить опции:

```
GRANT ALL PRIVILEGES ON *.* TO 'user'@'localhost' WITH GRANT OPTION;
```

В данном примере в этом нет необходимости, поэтому будут только назначены полные привилегии для всех баз данных:

```
GRANT ALL PRIVILEGES ON *.* TO 'selectel'@'localhost';
```

Вывод:

```
Query OK, 0 rows affected (0.00 sec)
```

Остается только применить изменения в настройках привилегий для этого служит команда:

```
FLUSH PRIVILEGES;
```

Вывод:

```
Query OK, 0 rows affected (0.01 sec)
```

В некоторых случаях может потребоваться удаление ранее созданного пользователя.

Это выполняется с использованием команды:

```
DROP USER 'user'@'localhost';
```

Проверяем что пользователь существует:

```
SELECT user FROM mysql.user;
```

Отправленная команда осуществляет вывод списка текущих пользователей.

Вывод:

```
+--------------------------+
| user                     |
+--------------------------+
| debian-sys-maint         |
| mysql.infoschema         |
| mysql.session            |
| mysql.sys                |
| phpmyadmin               |
| root                     |
| selectel                 |
+--------------------------+
7 rows in set (0.01 sec)
```

Убедившись что пользователь ‘selectel’ активен, можно произвести удаление:

```
DROP USER 'selectel'@'localhost';
```

Вывод:

```
Query OK, 0 rows affected (0.01 sec)
```

Производим повторную проверку, убеждаясь в его отсутствии в списке.

Вывод:

```
+--------------------------+
| user                     |
+--------------------------+
| debian-sys-maint         |
| mysql.infoschema         |
| mysql.session            |
| mysql.sys                |
| phpmyadmin               |
| root                     |
+--------------------------+
7 rows in set (0.00 sec)
```

На этом настройка пользователей завершена. Для выхода из mysql необходимо отправить команду **exit**.

Обеспечение безопасности phpMyAdmin

Если к серверу есть хоть какой-то доступ из интернета или ненадежной сети, необходимо обеспечить безопасность, добавив авторизацию.

Требуется создать файл ***.htaccess,*** который является конфигурационным файлом web-сервера Apache. Он дает возможность управлять web-сервером и настройками web-приложения с помощью директив, без изменения основного файла конфигурации web-сервера. В данной инструкции будет использован текстовый редактор **nano**.

Создаем файл ***.htaccess*** в директории **/usr/share/phpmyadmin/** и вносим в него следующие директивы:

```
AuthType Basic
Authname "Restricted Content"
AuthUserFile /etc/phpmyadmin/.htpasswd
Require valid-user
```

Производим команду:

```
sudo nano /usr/share/phpmyadmin/.htaccess
```

AuthType Basic — авторизация по паролю;

Authname «Restricted Content» — сообщение для окна авторизации;

AuthUserFile /etc/phpmyadmin/.htpasswd — путь к файлу пароля, который будет использован для авторизации;

Require valid-user — директива указывает, что только авторизованные пользователи получат доступ к ресурсу.

Теперь необходимо установить пароль учетной записи.

```
sudo htpasswd -c /etc/phpmyadmin/.htpasswd user
```

где **user** — учетная запись.

Отправляем команду:

```
sudo htpasswd -c /etc/phpmyadmin/.htpasswd selectel
```

и дважды вводим пароль.

Вывод:

```
Adding password for user selectel
```

Далее необходимо включить использование файлов .htaccess для директории /usr/share. Для этого откроем для редактирования файл apache2.conf и внесем изменения директивы для директории.

```
<Directory /usr/share>
AllowOverride All
Require all granted
</Directory>
```

По умолчанию файл выглядит так:

```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>

<Directory /usr/share>
        AllowOverride None
        Require all granted
</Directory>

<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride None
```

```
        Require all granted
</Directory>

#<Directory /srv/>
#       Options Indexes FollowSymLinks
#       AllowOverride None
#       Require all granted
#</Directory>
```

Вносим изменения в необходимую директиву:

```
<Directory /usr/share>
AllowOverride All
Require all granted
</Directory>
```

Вывод:

```
# This is the main Apache server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See http://httpd.apache.org/docs/2.4/ for detailed information about
# the directives and /usr/share/doc/apache2/README.Debian about Debian specific
# hints.
#
#
# Summary of how the Apache 2 configuration works in Debian:
# The Apache 2 web server configuration in Debian is quite different to
# upstream's suggested way to configure the web server. This is because Debian's
# default Apache2 installation attempts to make adding and removing modules,
# virtual hosts, and extra configuration directives as flexible as possible, in
# order to make automating the changes and administering the server as easy as
# possible.

# It is split into several files forming the configuration hierarchy outlined
# below, all located in the /etc/apache2/ directory:
#
```

```
#       /etc/apache2/
#       |-- apache2.conf
#       |       `--  ports.conf
#       |-- mods-enabled
#       |       |-- *.load
#       |       `-- *.conf
#       |-- conf-enabled
#       |       `-- *.conf
#       `-- sites-enabled
#               `-- *.conf
#
#
# * apache2.conf is the main configuration file (this file). It puts the pieces
#   together by including all remaining configuration files when starting up the
#   web server.
#
# * ports.conf is always included from the main configuration file. It is
#   supposed to determine listening ports for incoming connections which can be
#   customized anytime.
#
# * Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/
#   directories contain particular configuration snippets which manage modules,
#   global configuration fragments, or virtual host configurations,
#   respectively.
#
#   They are activated by symlinking available configuration files from their
#   respective *-available/ counterparts. These should be managed by using our
#   helpers a2enmod/a2dismod, a2ensite/a2dissite and a2enconf/a2disconf. See
#   their respective man pages for detailed information.
#
# * The binary is called apache2. Due to the use of environment variables, in
#   the default configuration, apache2 needs to be started/stopped with
#   /etc/init.d/apache2 or apache2ctl. Calling /usr/bin/apache2 directly will not
#   work with the default configuration.
# Global configuration
#


#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
```

```
#
# NOTE!  If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the Mutex documentation (available
# at <URL:http://httpd.apache.org/docs/2.4/mod/core.html#mutex>);
# you will save yourself a lot of trouble.
#
# Do NOT add a slash at the end of the directory path.
#
#ServerRoot "/etc/apache2"

#
# The accept serialization lock file MUST BE STORED ON A LOCAL DISK.
#
#Mutex file:${APACHE_LOCK_DIR} default

#
# The directory where shm and other runtime files will be stored.
#

DefaultRuntimeDir ${APACHE_RUN_DIR}

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
# This needs to be set in /etc/apache2/envvars
#
PidFile ${APACHE_PID_FILE}

#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 300

#
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
```

```
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 100

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 5


# These need to be set in /etc/apache2/envvars
User ${APACHE_RUN_USER}
Group ${APACHE_RUN_GROUP}

#
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off

# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog ${APACHE_LOG_DIR}/error.log

#
# LogLevel: Control the severity of messages logged to the error_log.
# Available values: trace8, ..., trace1, debug, info, notice, warn,
# error, crit, alert, emerg.
# It is also possible to configure the log level for particular modules, e.g.
```

```
# "LogLevel info ssl:warn"
#
LogLevel warn

# Include module configuration:
IncludeOptional mods-enabled/*.load
IncludeOptional mods-enabled/*.conf

# Include list of ports to listen on
Include ports.conf
# Sets the default security model of the Apache2 HTTPD server. It does
# not allow access to the root filesystem outside of /usr/share and /var/www.
# The former is used by web applications packaged in Debian,
# the latter may be used for local directories served by the web server. If
# your system is serving content from a sub-directory in /srv you must allow
# access here, or in any related virtual host.
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>

<Directory /usr/share>
        AllowOverride All
        Require all granted
</Directory>

<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>

#<Directory /srv/>
#       Options Indexes FollowSymLinks
#       AllowOverride None
#       Require all granted
#</Directory>
```

```
# AccessFileName: The name of the file to look for in each directory
# for additional configuration directives.  See also the AllowOverride
# directive.
#
AccessFileName .htaccess

#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<FilesMatch "^\.ht">
        Require all denied
</FilesMatch>


#
# The following directives define some format nicknames for use with
# a CustomLog directive.
#
# These deviate from the Common Log Format definitions in that they use %O
# (the actual bytes sent including headers) instead of %b (the size of the
# requested file), because the latter makes it impossible to detect partial
# requests.
#
# Note that the use of %{X-Forwarded-For}i instead of %h is not recommended.
# Use mod_remoteip instead.
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf
```

```
# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf
```

Сохраняем, перечитаем конфигурацию Apache2, отправкой команды:

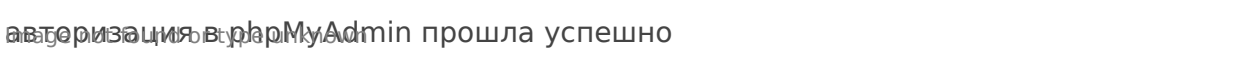
```
sudo systemctl reload apache2
```

Если сервер пока не в активной работе можно перезапустить демона командой:

```
sudo systemctl restart apache2
```

При попытке попасть в phpMyAdmin теперь потребуется дополнительная авторизация.

После ввода верных учетных данных авторизация проходит успешно.

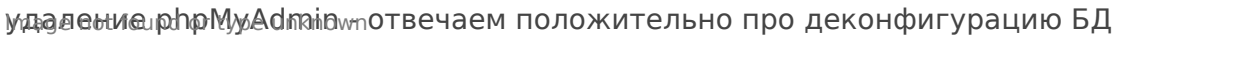


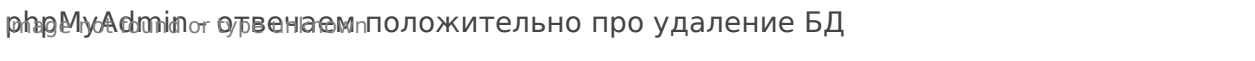
Удаление phpMyAdmin

Удаление происходит с использованием этих команд:

```
sudo apt remove phpmyadmin -y
```

Отвечаем на все вопросы положительно.





Вывод:

```
 Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  dbconfig-common dbconfig-mysql libjs-jquery libjs-openlayers libjs-sphinxdoc libjs-underscore libxslt1.1 php
php-google-recaptcha php-phpmyadmin-motranslator
  php-phpmyadmin-shapefile php-phpmyadmin-sql-parser php-phpseclib php-psr-cache php-psr-container php-
psr-log php-symfony-cache php-symfony-cache-contracts
  php-symfony-expression-language php-symfony-service-contracts php-symfony-var-exporter php-twig php-
twig-extensions php-xml php7.4 php7.4-xml
Use 'sudo apt autoremove' to remove them.
The following packages will be REMOVED:
  phpmyadmin
0 upgraded, 0 newly installed, 1 to remove and 41 not upgraded.
After this operation, 27.2 MB disk space will be freed.
(Reading database ... 54693 files and directories currently installed.)
Removing phpmyadmin (4:4.9.5+dfsg1-2) ...
Determining localhost credentials from /etc/mysql/debian.cnf: succeeded.
dbconfig-common: dumping mysql database phpmyadmin to /var/tmp/phpmyadmin.phpmyadmin.2021-11-16-
07.04.mysql.rTp4t6.
dbconfig-common: dropping mysql database phpmyadmin.
dropping database phpmyadmin: success.
verifying database phpmyadmin was dropped: success.
dbconfig-common: revoking privileges for user phpmyadmin on phpmyadmin.
revoking access to database phpmyadmin from phpmyadmin@localhost: success.
Conf phpmyadmin disabled.
apache2_invoke postrm:Disable configuration phpmyadmin
```

Для очистки неиспользуемых зависимостей применяем:

```
sudo apt-get autoremove
```

Вывод:

```
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  dbconfig-common dbconfig-mysql libjs-jquery libjs-openlayers libjs-sphinxdoc libjs-underscore libxslt1.1 php
php-google-recaptcha php-phpmyadmin-motranslator
  php-phpmyadmin-shapefile php-phpmyadmin-sql-parser php-phpseclib php-psr-cache php-psr-container php-
psr-log php-symfony-cache php-symfony-cache-contracts
  php-symfony-expression-language php-symfony-service-contracts php-symfony-var-exporter php-twig php-
twig-extensions php-xml php7.4 php7.4-xml
0 upgraded, 0 newly installed, 26 to remove and 41 not upgraded.
After this operation, 20.5 MB disk space will be freed.
Do you want to continue? [Y/n] y
(Reading database ... 52976 files and directories currently installed.)
Removing dbconfig-mysql (2.0.13) ...
Removing dbconfig-common (2.0.13) ...
Removing libjs-sphinxdoc (1.8.5-7ubuntu3) ...
Removing libjs-jquery (3.3.1~dfsg-3) ...
Removing libjs-openlayers (2.13.1+ds2-7) ...
Removing libjs-underscore (1.9.1~dfsg-1ubuntu0.20.04.1) ...
Removing php-xml (2:7.4+75) ...
Removing php7.4-xml (7.4.3-4ubuntu2.7) ...
Removing libxslt1.1:amd64 (1.1.34-4) ...
Removing php (2:7.4+75) ...
Removing php-google-recaptcha (1.2.3-1) ...
Removing php-phpmyadmin-motranslator (5.0.0-1) ...
Removing php-phpmyadmin-shapefile (2.1-3) ...
Removing php-phpmyadmin-sql-parser (4.6.1-2) ...
Removing php-phpseclib (2.0.23-2) ...
Removing php-symfony-expression-language (4.3.8+dfsg-1ubuntu1) ...
Removing php-symfony-cache (4.3.8+dfsg-1ubuntu1) ...
Removing php-symfony-cache-contracts (1.1.8-1) ...
Removing php-psr-cache (1.0.1-2) ...
Removing php-symfony-service-contracts (1.1.8-1) ...
Removing php-psr-container (1.0.0-2) ...
Removing php-psr-log (1.1.2-1) ...
Removing php-symfony-var-exporter (4.3.8+dfsg-1ubuntu1) ...
Removing php-twig-extensions (1.5.4-1) ...
Removing php-twig (2.12.5-1) ...
Removing php7.4 (7.4.3-4ubuntu2.7) ...
```

```
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libapache2-mod-php7.4 (7.4.3-4ubuntu2.7) ...
```

Перезапускаем демон apache2:

```
sudo service apache2 restart
```

PhpMyAdmin удален с сервера.

Возможные ошибки

В процессе изменения типа аутентификации учетной записи root:

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'password';
```

Возможно возникновение ошибки:

```
Plugin caching_sha2_password in not loaded
```

Это вызвано тем, что в свежих версиях mysql тип аутентификации по умолчанию:

```
caching_sha2_password
```

Это не позволит произвести удаленное подключение к mysql и вызовет ошибку плагина:

```
caching_sha2_password
```

Возможным вариантом решения является установка типа аутентификации:

```
mysql_native_password
```

Команда выглядит так:

```
ALTER USER 'username'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
```

Теперь возможен доступ пользователя к mysql с localhost.

Для подключения с нескольких хостов необходимо изменить команду:

```
ALTER USER 'username'@'%' IDENTIFIED WITH mysql_native_password BY 'password';
```

Для применения изменений необходимо использовать:

```
FLUSH PRIVILEGES;
```

Заключение

В этом мануале мы рассказали о настройке сервера phpMyAdmin на операционной системе Ubuntu 20.04 на облачном сервере от Selectel. Мы разобрали основные моменты, которых достаточно для того, чтобы быстро и качественно установить данный инструмент на сервер, а также рассмотрели настройки безопасности.

Revision #3
Created 23 March 2024 17:34:43 by ANelidov
Updated 24 May 2024 06:05:47 by VBusurin