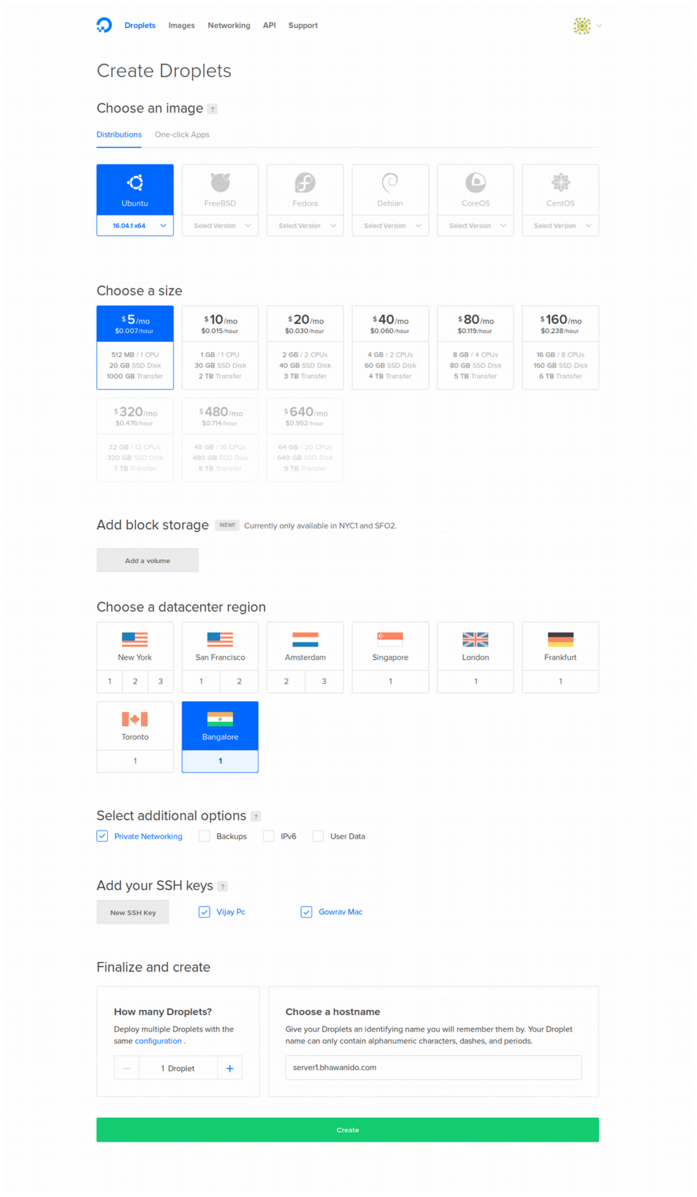
MariaDB Galera Cluster 10.1 Installation on DigitalOcean Ubuntu 16.04

Preparing the Server

follow the following Steps

1. Login to digitalocean.com & Create a Droplets



Droplets Images Networking API Support

Create Droplets

Choose an image

Distributions One-click Apps

Ubuntu	FreeBSD	Fedora	Debian	CoreOS	CentOS
16.04.1 x64	Select Version	Select Version	Select Version	Select Version	Select Version

Choose a size

$5/mo	$10/mo	$20/mo	$40/mo	$80/mo	$160/mo
$0.007/hour	$0.015/hour	$0.030/hour	$0.060/hour	$0.119/hour	$0.238/hour
512 MB / 1 CPU	1 GB / 1 CPU	2 GB / 2 CPUs	4 GB / 2 CPUs	8 GB / 4 CPUs	16 GB / 8 CPUs
20 GB SSD Disk	30 GB SSD Disk	40 GB SSD Disk	60 GB SSD Disk	80 GB SSD Disk	160 GB SSD Disk
1000 GB Transfer	2 TB Transfer	3 TB Transfer	4 TB Transfer	5 TB Transfer	6 TB Transfer

$320/mo	$480/mo	$640/mo
$0.476/hour	$0.714/hour	$0.952/hour
32 GB / 12 CPUs	48 GB / 16 CPUs	64 GB / 20 CPUs
320 GB SSD Disk	480 GB SSD Disk	640 GB SSD Disk
7 TB Transfer	8 TB Transfer	9 TB Transfer

Add block storage NEW! Currently only available in NYC1 and SFO2.

Add a volume

Choose a datacenter region

New York	San Francisco	Amsterdam	Singapore	London	Frankfurt	Toronto	Bangalore
1 2 3	1 2	2 3	1	1	1	1	1

Select additional options

[✓] Private Networking [ ] Backups [ ] IPv6 [ ] User Data

Add your SSH keys

New SSH Key [✓] Vijay Pc [✓] Gowrav Mac

Finalize and create

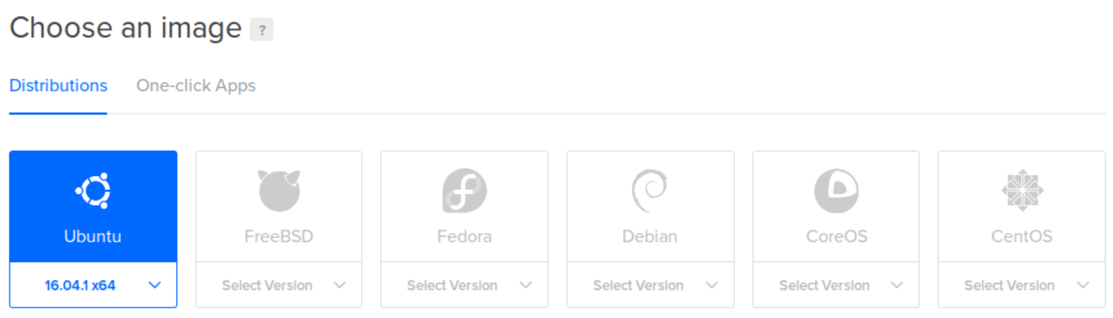
How many Droplets?
Deploy multiple Droplets with the same configuration .
1 Droplet

Choose a hostname
Give your Droplets an identifying name you will remember them by. Your Droplet name can only contain alphanumeric characters, dashes, and periods.
server1.bhawanido.com

Create

2. Create a Droplets to follow the instructions

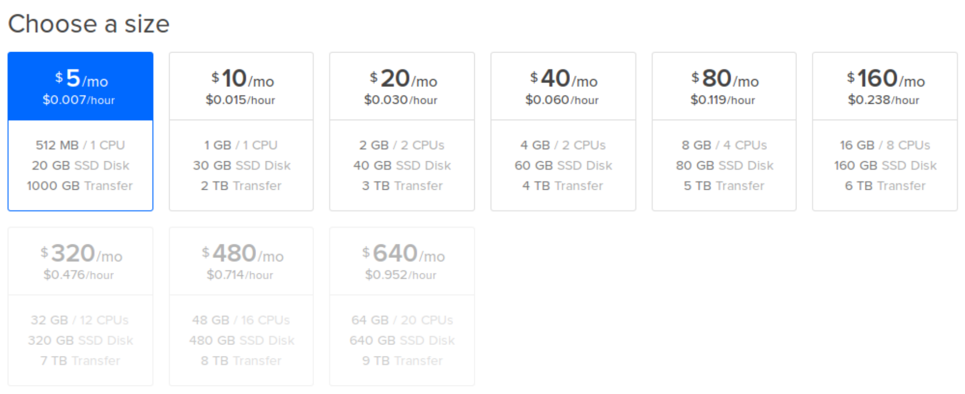
A. **Choose a droplets Image**


Choose an image ?

Distributions | One-click Apps

Ubuntu	FreeBSD	Fedora	Debian	CoreOS	CentOS
[✓ 16.04.1 x64]	Select Version	Select Version	Select Version	Select Version	Select Version

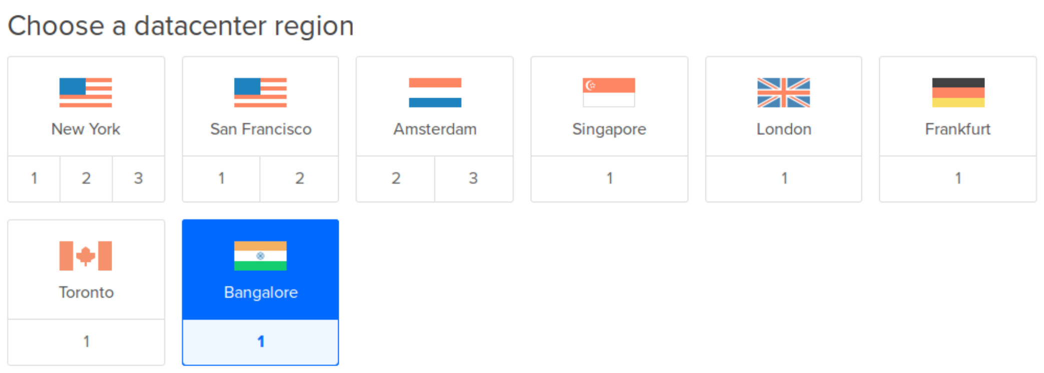
B. **Choose a size package**


Choose a size

[✓ $5/mo]	$10/mo	$20/mo	$40/mo	$80/mo	$160/mo
$0.007/hour	$0.015/hour	$0.030/hour	$0.060/hour	$0.119/hour	$0.238/hour
512 MB / 1 CPU	1 GB / 1 CPU	2 GB / 2 CPUs	4 GB / 2 CPUs	8 GB / 4 CPUs	16 GB / 8 CPUs
20 GB SSD Disk	30 GB SSD Disk	40 GB SSD Disk	60 GB SSD Disk	80 GB SSD Disk	160 GB SSD Disk
1000 GB Transfer	2 TB Transfer	3 TB Transfer	4 TB Transfer	5 TB Transfer	6 TB Transfer

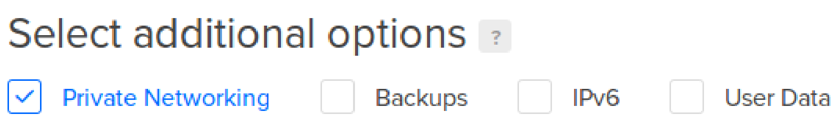
$320/mo	$480/mo	$640/mo
$0.476/hour	$0.714/hour	$0.952/hour
32 GB / 12 CPUs	48 GB / 16 CPUs	64 GB / 20 CPUs
320 GB SSD Disk	480 GB SSD Disk	640 GB SSD Disk
7 TB Transfer	8 TB Transfer	9 TB Transfer

C. Choose a Data Center Region


Choose a datacenter region

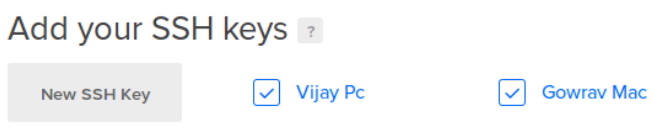
New York	San Francisco	Amsterdam	Singapore	London	Frankfurt	Toronto	[✓ Bangalore]
1 2 3	1 2	2 3	1	1	1	1	[✓ 1]

D. Select Additional Options


Select additional options ?

[✓ Private Networking] Backups IPv6 User Data

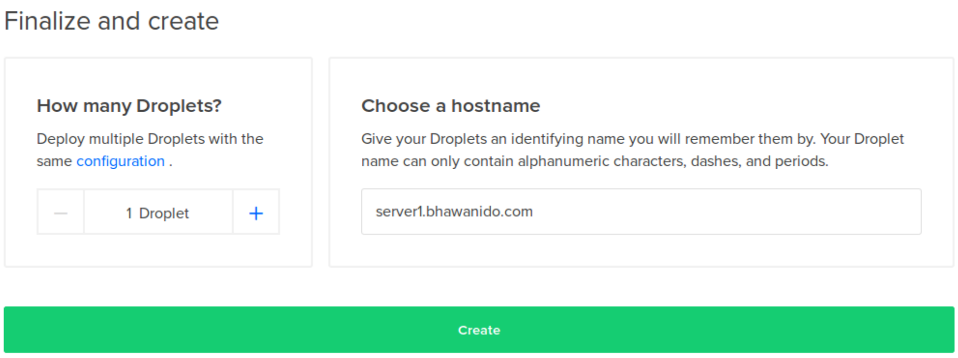
E. Add Your Ssh Key(optional)


Add your SSH keys ?

New SSH Key [✓ Vijay Pc] [✓ Gowrav Mac]

F. Finalize And Create Droplets

type your droplets name


Finalize and create

How many Droplets?

Deploy multiple Droplets with the same configuration .

Choose a hostname

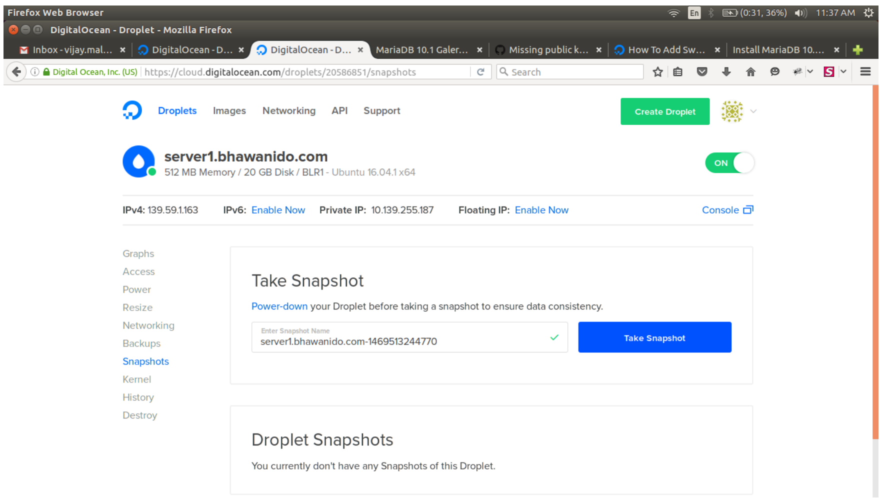
Give your Droplets an identifying name you will remember them by. Your Droplet name can only contain alphanumeric characters, dashes, and periods.

server1.bhawanido.com

G. Take a Snap Shot after ceate your Droplets

Click on droplets to open your droplets panel

click to snapshot & take a snap shot


Firefox Web Browser

DigitalOcean - Droplet - Mozilla Firefox

https://cloud.digitalocean.com/droplets/20586851/snapshots

server1.bhawanido.com

512 MB Memory / 20 GB Disk / BLR1 - Ubuntu 16.04.1 x64

IPv4: 139.59.1.163 IPv6: Enable Now Private IP: 10.139.255.187 Floating IP: Enable Now Console

Take Snapshot

Power-down your Droplet before taking a snapshot to ensure data consistency.

Enter Snapshot Name: server1.bhawanido.com-1469513244770

Droplet Snapshots

You currently don't have any Snapshots of this Droplet.

Add a Swap Space to Increase Your Droplets Capacities

Follow the command to add swap partition

1. Check the System for Swap Information

```
$ sudo swapon --show
```

If you don't get back any output, this means your system does not have swap space available currently.

You can verify that there is no active swap using the `free` utility:

```
$ free -h
```

Output

```
              total        used        free      shared  buff/cache   available
Mem:           488M         36M        104M        652K        348M        426M
Swap:            0B          0B          0B
```

As you can see in the "Swap" row of the output, no swap is active on the system.

Check Available Space on the Hard Drive Partition

```
$ df -h
```

Output

```
Filesystem      Size  Used Avail Use% Mounted on
udev            238M     0  238M   0% /dev
tmpfs            49M  624K   49M   2% /run
/dev/vda1        20G  1.1G   18G   6% /
tmpfs           245M     0  245M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           245M     0  245M   0% /sys/fs/cgroup
tmpfs            49M     0   49M   0% /run/user/1001
```

The device under /dev is our disk in this case. We have plenty of space available in this example (only 4.1G used). Your usage will probably be different

Create a Swap File

Now that we know our available hard drive space, we can go about creating a swap file within our filesystem. We will create a file of the swap size that we want called swapfile in our root (/) directory.

The best way of creating a swap file is with the `fallocate` program. This command creates a file of a preallocated size instantly.

Since the server in our example has 512MB of RAM, we will create a 4 Gigabyte file in this guide. Adjust this to meet the needs of your own server:

```
$ sudo fallocate -l 4G /swapfile
```

We can verify that the correct amount of space was reserved by typing:

```
$ ls -lh /swapfile
```

Output: -rw-r--r-- 1 root root 4.0G Apr 25 11:14 /swapfile

Our file has been created with the correct amount of space set aside.

Enabling the Swap File

```
$ sudo chmod 600 /swapfile
```

Verify the permissions change by typing:

```
$ ls -lh /swapfile
```

Output

```
-rw------- 1 root root 4.0G Apr 25 11:14 /swapfile
```

As you can see, only the root user has the read and write flags enabled.

We can now mark the file as swap space by typing:

```
$ sudo mkswap /swapfile
```

Output

```
Setting up swapspace version 1, size = 4 Gib (1073737728 bytes)
no label, UUID=6e965805-2ab9-450f-aed6-577e74089dbf
```

After marking the file, we can enable the swap file, allowing our system to start utilizing it:

```
$ sudo swapon /swapfile
```

We can verify that the swap is available by typing:

```
$ sudo swapon --show
```

Output

```
NAME      TYPE  SIZE USED PRIO
/swapfile file    4G   0B   -1
```

We can check the output of the free utility again to corroborate our findings:

```
$ free -h
```

Output

```
              total        used        free      shared  buff/cache   available
Mem:           488M         37M         96M        652K        354M        425M
Swap:          4.0G          0B        4.0G
```

Our swap has been set up successfully and our operating system will begin to use it as necessary.

More Details goto the link

https://www.digitalocean.com/community/tutorials/how-to-add-swap-space-on-ubuntu-16-04

server Setup

Install Apache with php and mysql

Install MariaDB 10.1 on Ubuntu14.04/15.10/16.04

I always like to install software packages from official repository (if there’s one) rather than from my Linux distribution repository. For one thing, I can install the latest stable version. For another, I don’t have to worry about distribution specific modifications. In other words, what I got is a bogstandard package no matter what Linux distribution I use.

This tutorial will guide you through the process of installing the latest stable version of MariaDB and that's MariaDB 10.1 on Ubuntu 14.04 and 15.10, 16.04. Ubuntu repository has MariaDB 10.0, but no MariaDB 10.1.

Step1: Install software-properties-common

```
$ sudo apt-get install software-properties-common
```

Step2: Fetch MariaDB signing key from Ubuntu's key server.

```
$ apt-key adv --keyserver ha.pool.sks-keyservers.net --recv-keys
F1656F24C74CD1D8
```

Step3: add MariaDB repository to your system.

ubuntu 14.04

```
sudo add-apt-repository 'deb [arch=amd64,i386]
http://sgp1.mirrors.digitalocean.com/mariadb/repo/10.1/ubuntu trusty main'
```

ubuntu 15.10

```
sudo add-apt-repository 'deb [arch=amd64,i386]
http://sgp1.mirrors.digitalocean.com/mariadb/repo/10.1/ubuntu wily main'
```

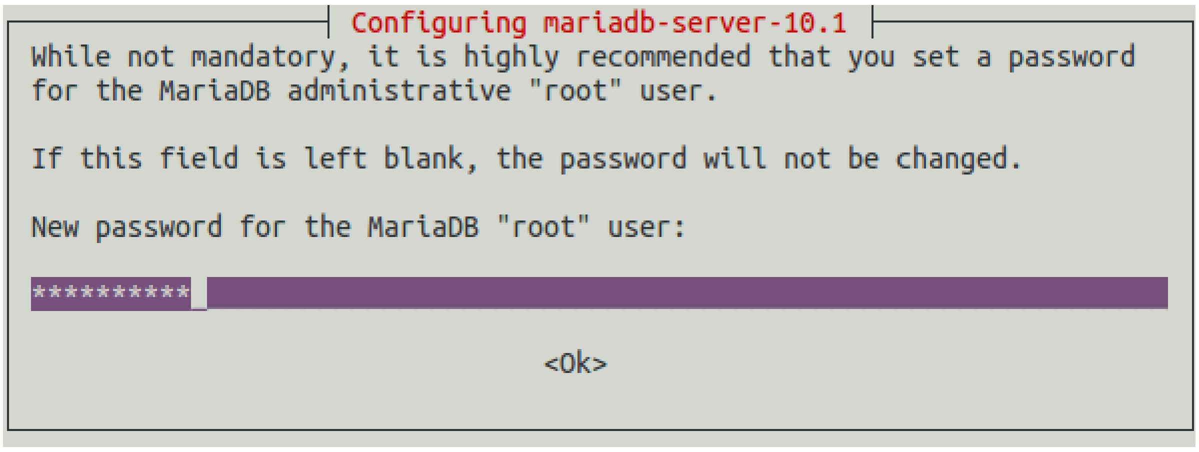
Ubuntu 16.04

```
$ sudo add-apt-repository 'deb [arch=amd64,i386]
http://sgp1.mirrors.digitalocean.com/mariadb/repo/10.1/ubuntu
xenial main'
```

```
wi$ sudo apt-get update

$ sudo apt-get install mariadb-server
```

You will be asked to set a password for the MariaDB root user.



```
Configuring mariadb-server-10.1
While not mandatory, it is highly recommended that you set a password
for the MariaDB administrative "root" user.

If this field is left blank, the password will not be changed.

New password for the MariaDB "root" user:

**********

                              <Ok>
```

After it’s installed, mysqld process will be automatically started.

Check version

```
$ mysql --version
```

```
root@server1:~# mysql --version
```

Output:

```
mysql  Ver 15.1 Distrib 10.1.16-MariaDB, for debian-linux-gnu
(x86_64) using readline 5.2

root@server1:~#
```

Finally we should execute the secure installation script to remove anonymous user, disable remote root login and remove test database for security reasons.

```
$ mysql_secure_installation
```

You will be asked these questions:

```
Enter current password for root (enter for none): <-- press enter
Set root password? [Y/n] <-- y
New password: <-- Enter the new MariaDB root password here
Re-enter new password: <-- Repeat the password
Remove anonymous users? [Y/n] <-- y
Disallow root login remotely? [Y/n] <-- y
Reload privilege tables now? [Y/n] <-- y
```

Test the login to MariaDB with the "mysql command"

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

Output:

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.

Your MariaDB connection id is 9

Server version: 10.1.16-MariaDB-1~xenial mariadb.org binary distribution

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

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

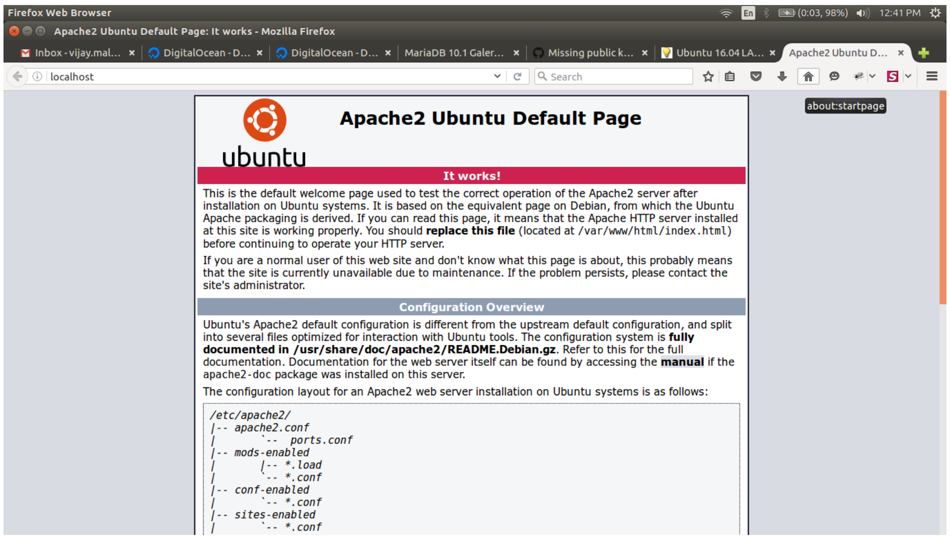
To leave the MariaDB shell, enter the command "quit" and press enter.

Install Apache 2.4

Apache 2 is available as an Ubuntu package, therefore we can install it like this:

$ sudo apt-get -y install apache2

Now direct your browser to http://192.168.1.100, and you should see the Apache2 default page (It works!):



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

The document root of the apache default vhost is /var/www/html on Ubuntu and the main configuration file is /etc/apache2/apache2.conf. The configuration system is fully documented in */usr/share/doc/apache2/README.Debian.gz*.

Install PHP 7

We can install PHP 7 and the Apache PHP module as follows:

$ sudo apt-get -y install php7.0 libapache2-mod-php7.0

Then restart Apache:

$ sudo systemctl restart apache2 or **sudo service apache2 restart**

Test PHP and get details about your PHP installation

The document root of the default web site is /var/www/html. We will now create a small PHP file (info.php) in that directory and call it in a browser. The file will display lots of useful details about our PHP installation, such as the installed PHP version.

$ php

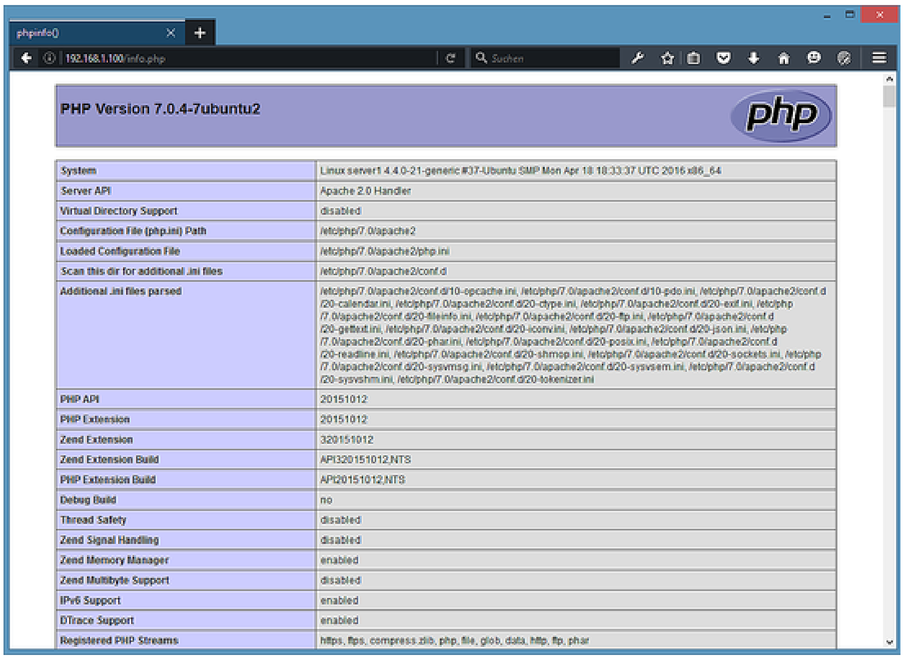
```
<?php
phpinfo();
?>
```

Clt+x, y ,enter

Then change the owner of the info.php file to the www-data user and group.

$ chown www-data:www-data /var/www/html/info.php

Now we call that file in a browser (e.g. http://ip_address/info.php):



PHP Version 7.0.4-7ubuntu2

System	Linux server1 4.4.0-21-generic #37-Ubuntu SMP Mon Apr 18 18:33:37 UTC 2016 x86_64
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.0/apache2
Loaded Configuration File	/etc/php/7.0/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.0/apache2/conf.d
Additional .ini files parsed	/etc/php/7.0/apache2/conf.d/10-opcache.ini, /etc/php/7.0/apache2/conf.d/10-pdo.ini, /etc/php/7.0/apache2/conf.d/20-calendar.ini, /etc/php/7.0/apache2/conf.d/20-ctype.ini, /etc/php/7.0/apache2/conf.d/20-exif.ini, /etc/php/7.0/apache2/conf.d/20-fileinfo.ini, /etc/php/7.0/apache2/conf.d/20-ftp.ini, /etc/php/7.0/apache2/conf.d/20-gettext.ini, /etc/php/7.0/apache2/conf.d/20-iconv.ini, /etc/php/7.0/apache2/conf.d/20-json.ini, /etc/php/7.0/apache2/conf.d/20-phar.ini, /etc/php/7.0/apache2/conf.d/20-posix.ini, /etc/php/7.0/apache2/conf.d/20-readline.ini, /etc/php/7.0/apache2/conf.d/20-shmop.ini, /etc/php/7.0/apache2/conf.d/20-sockets.ini, /etc/php/7.0/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.0/apache2/conf.d/20-sysvsem.ini, /etc/php/7.0/apache2/conf.d/20-sysvshm.ini, /etc/php/7.0/apache2/conf.d/20-tokenizer.ini
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012
Zend Extension Build	API320151012,NTS
PHP Extension Build	API20151012,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar

As you see, PHP 7.0 is working, and it's working through the Apache 2.0 Handler, as shown in the Server API line. If you scroll further down, you will see all modules that are already enabled in PHP5. MySQL is not listed there which means we don't have MySQL / MariaDB support in PHP yet.

Get MySQL / MariaDB support in PHP

To get MySQL support in PHP, we can install the php7.0-mysql package. It's a good idea to install some other PHP modules as well as you might need them for your applications. You can search for available PHP modules like this:

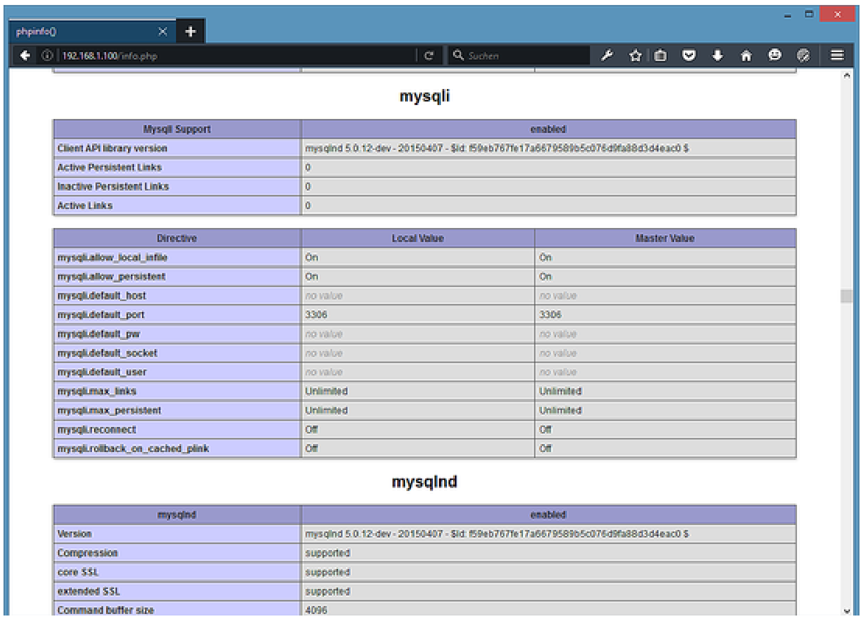
$ apt-cache search php7.0

Pick the ones you need and install them like this:

$ sudo apt-get -y install php7.0-mysql php7.0-curl php7.0-gd php7.0-intl php-pear php-imagick php7.0-imap php7.0-mcrypt php-memcache php7.0-pspell php7.0-recode php7.0-sqlite3 php7.0-tidy php7.0-xmlrpc php7.0-xsl php7.0-mbstring php-gettext

Now restart Apache2:

$ systemctl restart apache2



phpinfo()

192.168.1.100/info.php

mysqli

Mysqli Support	enabled
Client API library version	mysqlnd 5.0.12-dev - 20150407 - $Id: f59eb767fe17a6679589b5c076d9fa88d3d4eac0 $
Active Persistent Links	0
Inactive Persistent Links	0
Active Links	0

Directive	Local Value	Master Value
mysqli.allow_local_infile	On	On
mysqli.allow_persistent	On	On
mysqli.default_host	no value	no value
mysqli.default_port	3306	3306
mysqli.default_pw	no value	no value
mysqli.default_socket	no value	no value
mysqli.default_user	no value	no value
mysqli.max_links	Unlimited	Unlimited
mysqli.max_persistent	Unlimited	Unlimited
mysqli.reconnect	Off	Off
mysqli.rollback_on_cached_plink	Off	Off

mysqlnd

mysqlnd	enabled
Version	mysqlnd 5.0.12-dev - 20150407 - $Id: f59eb767fe17a6679589b5c076d9fa88d3d4eac0 $
Compression	supported
core SSL	supported
extended SSL	supported
Command buffer size	4096

PHP 7 has now MySQL / MariaDB support as shown in phpinfo() above.

Install the APCu PHP cache to speed up PHP

APCu is a free PHP opcode cacher for caching and optimizing PHP intermediate code. It is strongly recommended to have an Opcache installed to speed up your PHP page.

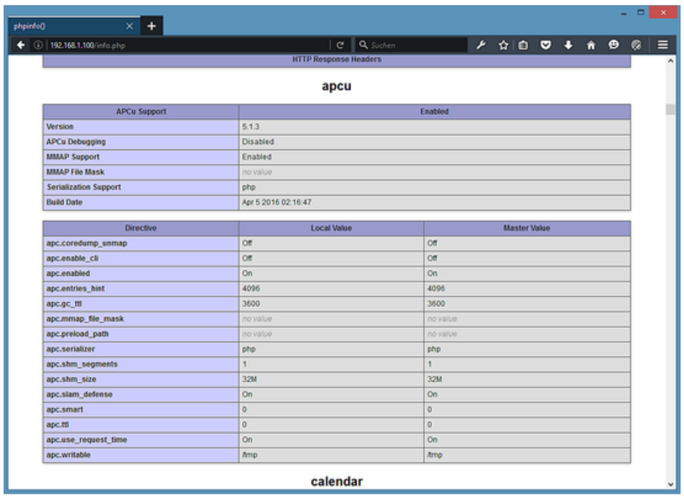
APCu can be installed as follows:

$ sudo apt-get -y install php-apcu

Now restart Apache:

$ systemctl restart apache2

Now reload http://192.168.1.100/info.php in your browser and scroll down to the modules section again. You should now find lots of new modules there:



phpinfo()

192.168.1.100/info.php

HTTP Response Headers

apcu

APCu Support	Enabled
Version	5.1.3
APCu Debugging	Disabled
MMAP Support	Enabled
MMAP File Mask	no value
Serialization Support	php
Build Date	Apr 5 2016 02:16:47

Directive	Local Value	Master Value
apc.coredump_unmap	Off	Off
apc.enable_cli	Off	Off
apc.enabled	On	On
apc.entries_hint	4096	4096
apc.gc_ttl	3600	3600
apc.mmap_file_mask	no value	no value
apc.preload_path	no value	no value
apc.serializer	php	php
apc.shm_segments	1	1
apc.shm_size	32M	32M
apc.slam_defense	On	On
apc.smart	0	0
apc.ttl	0	0
apc.use_request_time	On	On
apc.writable	/tmp	/tmp

calendar

Please don't forget to delete the info.php file when you don't need it anymore as it provides sensitive details of your server. Run the following command to delete the file.

$ rm -f /var/www/html/info.php

Enable the SSL website in apache

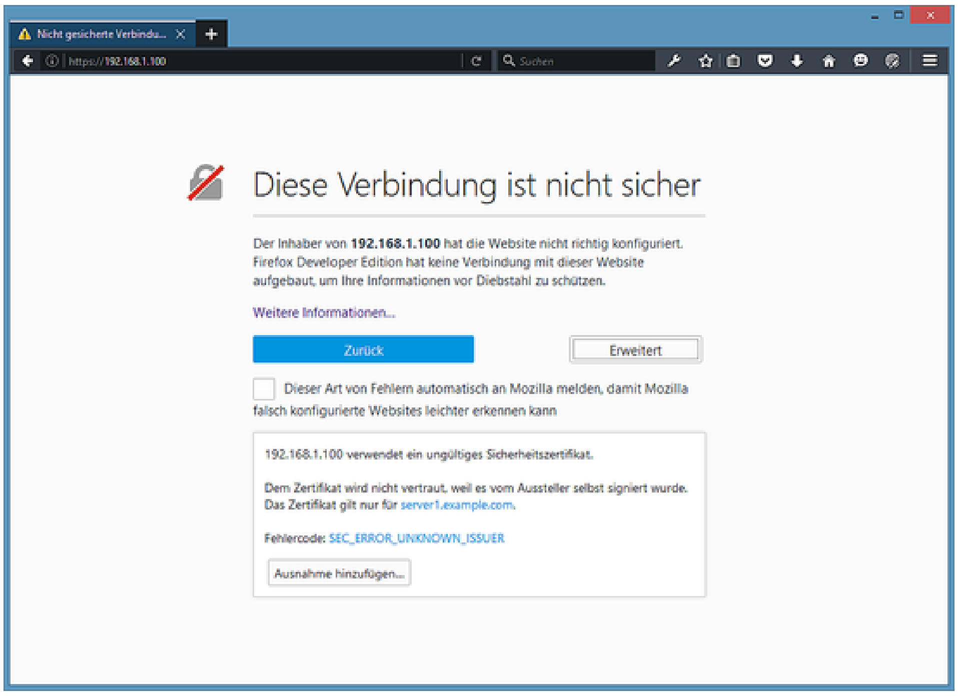
SSL/ TLS is a security layer to encrypt the connection between the web browser and your server. Execute the following commands on your server to enable https:// support. Run:

```
$ a2enmod ssl
$ a2ensite default-ssl
```

which enables the ssl module and adds a symlink in the /etc/apache2/sites-enabled folder to the file /etc/apache2/sites-available/default-ssl.conf to include it into the active apache configuration. Then restart apache to enable the new configuration:

```
$ systemctl restart apache2
```

Now test the SSL connection by opening https://ip_address in a web browser.



Diese Verbindung ist nicht sicher

Der Inhaber von **192.168.1.100** hat die Website nicht richtig konfiguriert. Firefox Developer Edition hat keine Verbindung mit dieser Website aufgebaut, um Ihre Informationen vor Diebstahl zu schützen.

Weitere Informationen...

Zurück   Erweitert

Dieser Art von Fehlern automatisch an Mozilla melden, damit Mozilla falsch konfigurierte Websites leichter erkennen kann

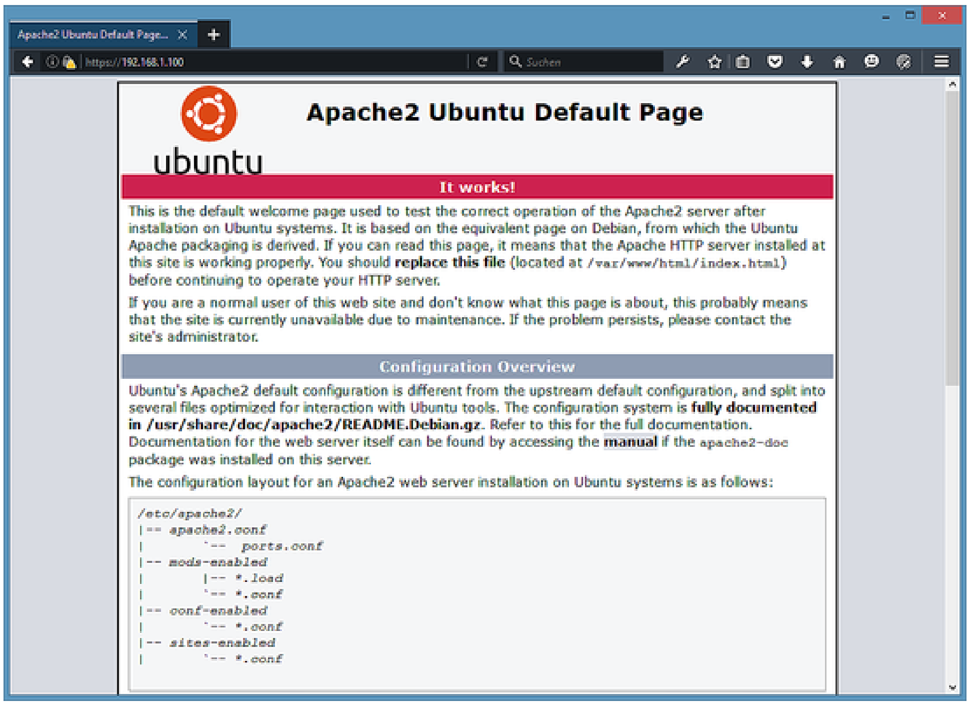
192.168.1.100 verwendet ein ungültiges Sicherheitszertifikat.

Dem Zertifikat wird nicht vertraut, weil es vom Aussteller selbst signiert wurde. Das Zertifikat gilt nur für server1.example.com.

Fehlercode: SEC_ERROR_UNKNOWN_ISSUER

Ausnahme hinzufügen...

You will receive an SSL warning as the SSL certificate of the server is a "self-signed" SSL certificate, this means that the browser does not trust this certificate by default and you have to accept the security warning first. After accepting the warning, you will see the apache default page.



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `-- ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

The closed "Green Lock" in front of the URL in the browser shows that the connection is encrypted. To get rid of the SSL warning, replace the self-signed SSL certificate /etc/ssl/certs/ssl-cert-snakeoil.pem with an officially signed SSL certificate from an SSL Authority.

Install phpMyAdmin

phpMyAdmin is a web interface through which you can manage your MySQL databases. It's a good idea to install it:

```
$ sudo apt-get -y install phpmyadmin
```

You will see the following questions:

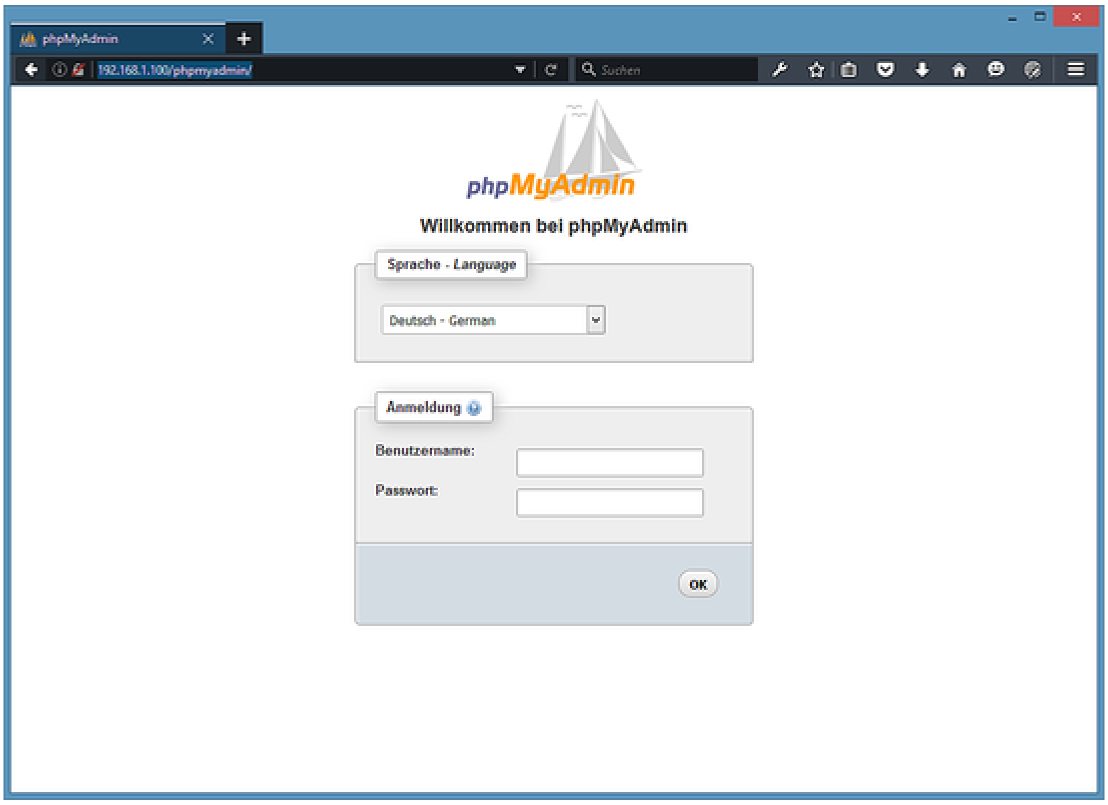
Web server to configure automatically: <-- **Select the option**: **apache2**
Configure database for phpmyadmin with dbconfig-common? <-- **Yes**
MySQL application password for phpmyadmin: <-- **Press enter, apt will create a random password automatically**.

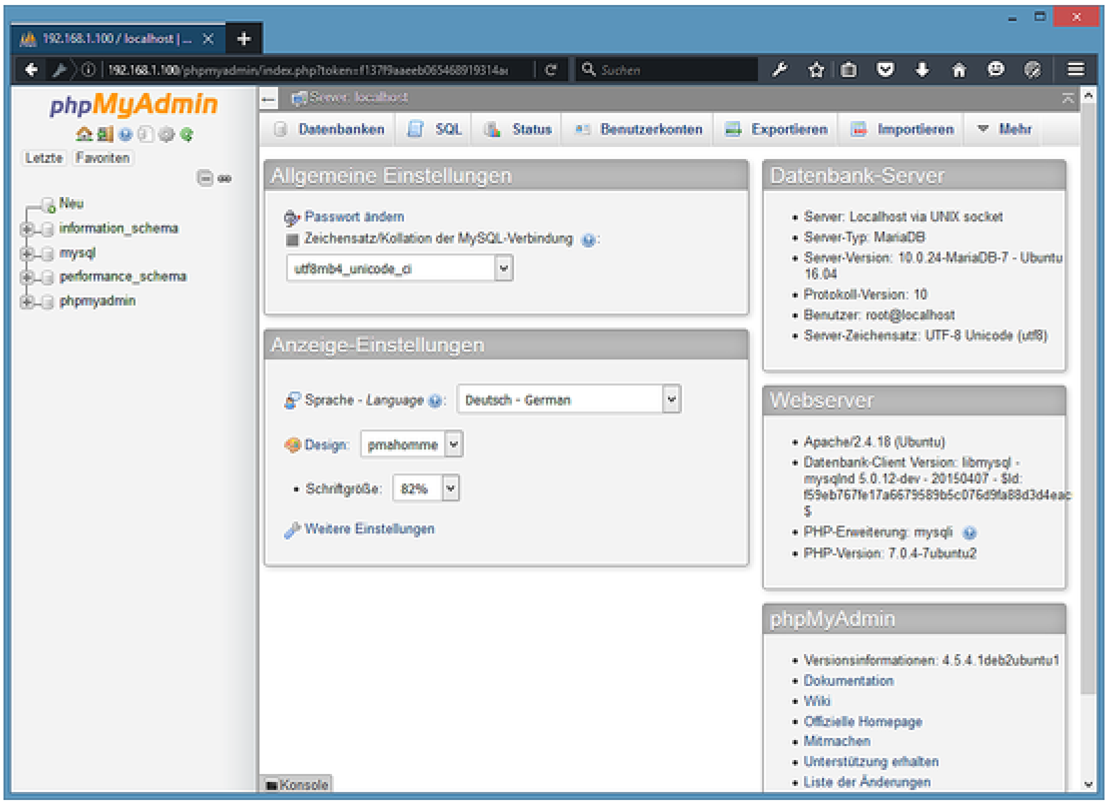
MariaDB enables a plugin called "unix_socket" for the root user by default, this plugin prevents that the root user can log in to PHPMyAdmin and that TCP connections to MySQL are working for the root user. Therefore, I'll deactivate that plugin with the following command:

```
$ echo "update user set plugin='' where User='root'; flush privileges;" | mysql -u root -p mysql
```

Enter the MariaDB root password, when requested by the mysql command.

Afterward, you can access phpMyAdmin under http://192.168.1.100/phpmyadmin/:





Server: localhost

Allgemeine Einstellungen
- Passwort ändern
- Zeichensatz/Kollation der MySQL-Verbindung: utf8mb4_unicode_ci

Anzeige-Einstellungen
- Sprache - Language: Deutsch - German
- Design: pmahomme
- Schriftgröße: 82%
- Weitere Einstellungen

Datenbank-Server
- Server: Localhost via UNIX socket
- Server-Typ: MariaDB
- Server-Version: 10.0.24-MariaDB-7 - Ubuntu 16.04
- Protokoll-Version: 10
- Benutzer: root@localhost
- Server-Zeichensatz: UTF-8 Unicode (utf8)

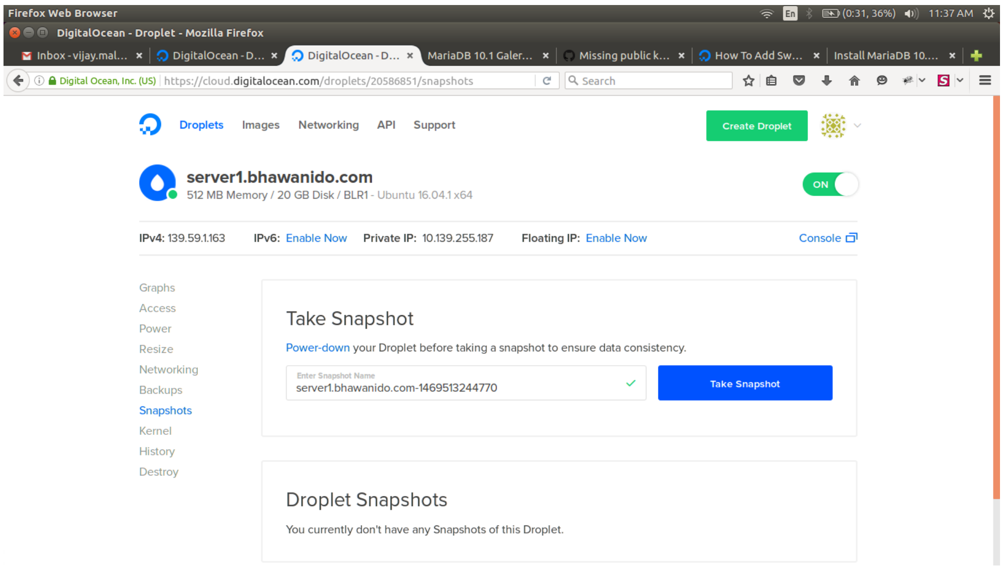
Webserver
- Apache/2.4.18 (Ubuntu)
- Datenbank-Client Version: libmysql - mysqlnd 5.0.12-dev - 20150407 - $Id: f59eb767fe17a6679589b5c076d9fa88d3d4eac $
- PHP-Erweiterung: mysqli
- PHP-Version: 7.0.4-7ubuntu2

phpMyAdmin
- Versionsinformationen: 4.5.4.1deb2ubuntu1
- Dokumentation
- Wiki
- Offizielle Homepage
- Mitmachen
- Unterstützung erhalten
- Liste der Änderungen

Take a Snap Shot Configuration server Setup

Click on droplets to open your droplets panel

```
click to snapshot & take a snap shot
```



server1.bhawanido.com
512 MB Memory / 20 GB Disk / BLR1 - Ubuntu 16.04.1 x64

IPv4: 139.59.1.163 IPv6: Enable Now Private IP: 10.139.255.187 Floating IP: Enable Now

Take Snapshot

Power-down your Droplet before taking a snapshot to ensure data consistency.

Enter Snapshot Name: server1.bhawanido.com-1469513244770

Droplet Snapshots

You currently don't have any Snapshots of this Droplet.

This is a Howto about installing MariaDB Galera Cluster

What we need

```
In our setup we assume 3 nodes (cluster01, cluster02, cluster03) with one
interface each. We assume following IP addresses: 139.59.1.163, 139.59.16.234,
and 139.59.4.203.
```

Configuring Galera

So we have to do some configuration next. There is a MariaDB configuration part and one part to configure Galera (starting with wsrep_). As we do the most basic and simple installation in this Howto, it is sufficient you just change the IP's (Remember: 10.0.99.31, 10.0.99.32, 10.0.99.33) with your IP's. In our example, we have set hostnames on each node (cluster01, cluster02, cluster03) so we do not need the IP addresses of the hosts.
This will be needed to define the wsrep_cluster_address Variable (the list of nodes a starting mysqld contacts to join the cluster).

The following configuration file has to be distributed on all nodes. We use a separate configuration file (create a new file) /etc/mysql/conf.d/galera.cnf with the following settings:

$ sudo /etc/mysql/conf.d

$ nano galera.cnf

```
[mysqld]
#mysql settings
binlog_format=ROW
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
innodb_doublewrite=1
query_cache_size=0
query_cache_type=0
bind-address=0.0.0.0

#galera settings
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
wsrep_cluster_name="test_cluster"
wsrep_cluster_address=gcomm://cluster01,cluster02,cluster03
wsrep_sst_method=rsync
```

FYI: The shared library for wsrep_provider is provided by the installed galera package.

We could also change the cluster name by changing the value of wserp_cluster_name to fit our style. This setting also works as a shared secret to control the access to the cluster.

With wsrep_cluster_address you see the hostnames of our setup. wsrep_cluster_address could also be gcomm://10.0.99.31,10.0.99.32,10.0.99.33. Multiple IP's or hostnames have to be comma seperated.

The wsrep_sst_method tells what method to use to synchronise the nodes. While there are also mysqldump and xtrabackup available, I prefer rsync because it is easy to configure (i.e. it does not need any credentials set on the nodes). If you are considering using the xtrabackup method, don't forget to install xtrabackup.

Now stop mysqld on all nodes:

```
cluster01# systemctl stop mysql
cluster02# systemctl stop mysql
cluster03# systemctl stop mysql
```

Debian/Ubuntu uses a special user (‚debian-sys-maint'@'localhost') in their init script and the

credentials for that user are stored in /etc/mysql/debian.cnf. This user is used to make some checks starting MySQL. Checks I don't think belong into a service script anyway. Because of the unique password for the user on each system, mysqld will throw some errors while starting and stopping. We could simply ignore it, but the user is also used to shutdown mysqld. This is also not required, as a SIGTERM is sufficient to shutdown the mysqld :/
So we've got to fix it, by copying /etc/mysql/debian.cnf from the first node (cluster01) to all other nodes. So the data and configuration files have the same data.

Starting the Galera Cluster

The configuration file (galera.cnf) is already distributed to all nodes, so we next start the first mysqld on cluster01. This node initializes/starts the cluster (creates a GTID).

```
cluster01# galera_new_cluster

cluster01# mysql -u root -p -e 'SELECT VARIABLE_VALUE as "cluster size" FROM
INFORMATION_SCHEMA.GLOBAL_STATUS WHERE VARIABLE_NAME="wsrep_cluster_size"'

+--------------+
| cluster size |
+--------------+
| 1            |
+--------------+
```

If you see the above, great! That's what we would expect. Now that the Cluster already exists, we let the next nodes just start and join the cluster.

```
cluster02# systemctl start mysql
```

Let's pause here and do a quick check. As we are running a cluster it is not important if we execute the following on cluster01 or cluster02.

```
$ mysql -u root -p -e 'SELECT VARIABLE_VALUE as "cluster size" FROM
INFORMATION_SCHEMA.GLOBAL_STATUS WHERE VARIABLE_NAME="wsrep_cluster_size"'

+--------------+
| cluster size |
+--------------+
| 2            |
+--------------+
```

If you see the above, very nice! Now let's start the third node:

```
cluster03# systemctl start mysql

$ mysql -u root -p -e 'SELECT VARIABLE_VALUE as "cluster size" FROM
INFORMATION_SCHEMA.GLOBAL_STATUS WHERE VARIABLE_NAME="wsrep_cluster_size"'

+--------------+
| cluster size |
+--------------+
| 3            |
```

Ok we are finished. We have a running MariaDB Galera Cluster \o/

Restarting the whole Galera Cluster

If the whole galera cluster has to be restarted (due to poweroutage for example) we have to complete the following steps by hand:

1. Identify the node with the most advanced node state ID.
2. Start the most advanced node as the first node of the cluster.
3. Start the rest of the node as usual.

Identify the node with the most advanced node state ID.

```
$ cat /var/lib/mysql/grastate.dat
```

The node with the highest seqno is your new first host.

Start the most advanced node as the first node of the cluster.

On the new first host, the one with the highes squence number, bootstrap the galera cluster again:

```
$ galera_new_cluster
```

Start the rest of the node as usual.

On each other node start mysqld as usual:

```
$ systemctl start mysql
```