


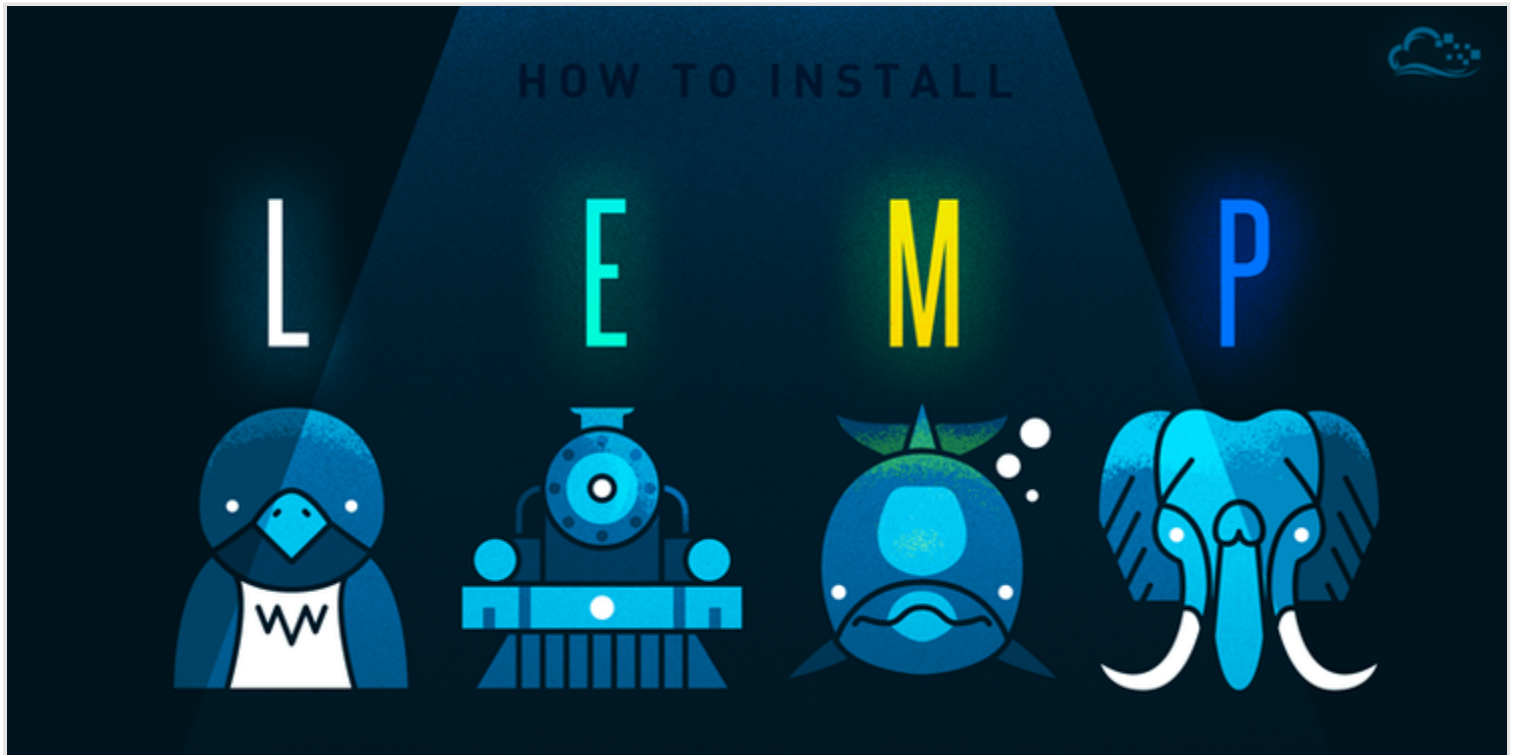
Introduction to Nginx and LEMP on Ubuntu 14.04 >

How To Install Linux, Nginx, MySQL... ▾

 Subscribe

 Share

 Contents ▾



How To Install Linux, Nginx, MySQL, PHP (LEMP) stack on Ubuntu 14.04


259

Posted April 25, 2014  729.4k

GETTING STARTED

NGINX

MYSQL

PHP

UBUNTU

By: Justin Ellingwood

Not using **Ubuntu 14.04**? Choose a different version:

Introduction

The LEMP software stack is a group of software that can be used to serve dynamic web pages and web applications. This is an acronym that describes a Linux operating system, with an Nginx web server. The backend data is stored in MySQL and the dynamic processing is handled by PHP.

In this guide, we will demonstrate how to install a LEMP stack on an Ubuntu 14.04 server. The Ubuntu operating system takes care of the first requirement. We will describe how to get the rest of the components up and running.

Note: The LEMP Stack can be installed automatically on your Droplet by adding [this script](#) to its User Data when launching it. Check out [this tutorial](#) to learn more about Droplet User Data.

Prerequisites

Before you complete this tutorial, you should have a regular, non-root user account on your server with `sudo` privileges. You can learn how to set up this type of account by completing steps 1-4 in our [Ubuntu 14.04 initial server setup](#).

Once you have your account available, sign into your server with that username. You are now ready to begin the steps outlined in this guide.

1. Install the Nginx Web Server

In order to display web pages to our site visitors, we are going to employ Nginx, a modern, efficient web server.

All of the software we will be getting for this procedure will come directly from Ubuntu's default package repositories. This means we can use the `apt` package management suite to complete the installation.

Since this is our first time using `apt` for this session, we should start off by updating our local package index. We can then install the server:

```
sudo apt-get update
sudo apt-get install nginx
```

In Ubuntu 14.04, Nginx is configured to start running upon installation.

You can test if the server is up and running by accessing your server's domain name or public IP address in your web browser.

If you do not have a domain name pointed at your server and you do not know your server's public IP address, you can find it by typing one of the following into your terminal:

```
ip addr show eth0 | grep inet | awk '{ print $2; }' | sed 's/\./.$$/'
```

```
111.111.111.111
fe80::601:17ff:fe61:9801
```

Or you could try using:

```
curl http://icanhazip.com
```

111.111.111.111

Try one of the lines that you receive in your web browser. It should take you to Nginx's default landing page:

http://server_domain_name_or_IP

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

If you see the above page, you have successfully installed Nginx.

2. Install MySQL to Manage Site Data

Now that we have a web server, we need to install MySQL, a database management system, to store and manage the data for our site.

You can install this easily by typing:

```
sudo apt-get install mysql-server
```

You will be asked to supply a root (administrative) password for use within the MySQL system.

The MySQL database software is now installed, but its configuration is not exactly complete yet.

First, we need to tell MySQL to generate the directory structure it needs to store its databases and information. We can do this by typing:

```
sudo mysql_install_db
```

Next, you'll want to run a simple security script that will prompt you to modify some insecure defaults. Begin the script by typing:

```
sudo mysql_secure_installation
```

You will need to enter the MySQL root password that you selected during installation.

Next, it will ask if you want to change that password. If you are happy with your MySQL root password, type "N" for no and hit "ENTER". Afterwards, you will be prompted to remove some test users and databases. You should just hit "ENTER" through these prompts to remove the unsafe default settings.

Once the script has been run, MySQL is ready to go.

3. Install PHP for Processing

Now we have Nginx installed to serve our pages and MySQL installed to store and manage our data, but we still need something to connect these two pieces and to generate dynamic content. We can use PHP for this.

Since Nginx does not contain native PHP processing like some other web servers, we will need to install `php5-fpm`, which stands for "fastCGI process manager". We will tell Nginx to pass PHP requests to this software for processing.

We can install this module and will also grab an additional helper package that will allow PHP to communicate with our database backend. The installation will pull in the necessary PHP core files. Do this by typing:

```
sudo apt-get install php5-fpm php5-mysql
```

Configure the PHP Processor

We now have our PHP components installed, but we need to make a slight configuration change to make our setup more secure.

Open the main `php5-fpm` configuration file with root privileges:

```
sudo nano /etc/php5/fpm/php.ini
```

What we are looking for in this file is the parameter that sets `cgi.fix_pathinfo`. This will be commented out with a semi-colon (;) and set to "1" by default.

This is an extremely insecure setting because it tells PHP to attempt to execute the closest file it can find if a PHP file does not match exactly. This basically would allow users to craft PHP requests in a way that would allow them to execute scripts that they shouldn't be allowed to execute.

We will change both of these conditions by uncommenting the line and setting it to "0" like this:

```
cgi.fix_pathinfo=0
```

Save and close the file when you are finished.

Now, we just need to restart our PHP processor by typing:

```
sudo service php5-fpm restart
```

This will implement the change that we made.

4. Configure Nginx to Use our PHP Processor

Now, we have all of the required components installed. The only configuration change we still need to do is tell Nginx to use our PHP processor for dynamic content.

We do this on the server block level (server blocks are similar to Apache's virtual hosts). Open the default Nginx server block configuration file by typing:

```
sudo nano /etc/nginx/sites-available/default
```

Currently, with the comments removed, the Nginx default server block file looks like this:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;

    root /usr/share/nginx/html;
    index index.html index.htm;

    server_name localhost;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

We need to make some changes to this file for our site.

- First, we need to add an `index.php` option as the first value of our `index` directive to allow PHP index files to be served when a directory is requested.
- We also need to modify the `server_name` directive to point to our server's domain name or public IP address.

- The actual configuration file includes some commented out lines that define error processing routines. We will uncomment those to include that functionality.
- For the actual PHP processing, we will need to uncomment a portion of another section. We will also need to add a `try_files` directive to make sure Nginx doesn't pass bad requests to our PHP processor.

The changes that you need to make are in red in the text below:

```
server {  
    listen 80 default_server;  
    listen [::]:80 default_server ipv6only=on;  
  
    root /usr/share/nginx/html;  
    index index.php index.html index.htm;  
  
    server_name server_domain_name_or_IP;  
  
    location / {  
        try_files $uri $uri/ =404;  
    }  
  
    error_page 404 /404.html;  
    error_page 500 502 503 504 /50x.html;  
    location = /50x.html {  
        root /usr/share/nginx/html;  
    }  
  
    location ~ \.php$ {  
        try_files $uri =404;  
        fastcgi_split_path_info ^(.+\.php)(/.+)$;  
        fastcgi_pass unix:/var/run/php5-fpm.sock;  
        fastcgi_index index.php;  
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;  
  
        include fastcgi_params;  
    }  
}
```

When you've made the above changes, you can save and close the file.

Restart Nginx to make the necessary changes:

```
sudo service nginx restart
```

5. Create a PHP File to Test Configuration

Your LEMP stack should now be completely set up. We still should test to make sure that Nginx can correctly hand `.php` files off to our PHP processor.

We can do this by creating a test PHP file in our document root. Open a new file called `info.php` within your document root in your text editor:

```
sudo nano /usr/share/nginx/html/info.php
```

We can type this into the new file. This is valid PHP code that will return formatted information about our server:

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

When you are finished, save and close the file.

Now, you can visit this page in your web browser by visiting your server's domain name or public IP address followed by `/info.php`:

```
http://server_domain_name_or_IP/info.php
```

You should see a web page that has been generated by PHP with information about your server:



System	Linux lemp 3.13.0-24-generic #46-Ubuntu SMP Thu Apr 10 19:11:08 UTC 2014 x86_64
Build Date	Apr 9 2014 17:10:12
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/fpm
Loaded Configuration File	/etc/php5/fpm/php.ini
Scan this dir for additional .ini files	/etc/php5/fpm/conf.d
Additional .ini files parsed	/etc/php5/fpm/conf.d/05-opcache.ini, /etc/php5/fpm/conf.d/10-pdo.ini, /etc/php5/fpm/conf.d/20-json.ini, /etc/php5/fpm/conf.d/20-mysql.ini, /etc/php5/fpm/conf.d/20-mysqli.ini, /etc/php5/fpm/conf.d/20-pdo_mysql.ini
PHP API	20121113
PHP Extension	20121212
Zend Extension	220121212
Zend Extension Build	API220121212,NTS
PHP Extension Build	API20121212,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled

If you see a page that looks like this, you've set up PHP processing with Nginx successfully.

After you test this, it's probably best to remove the file you created as it can actually give unauthorized users some hints about your configuration that may help them try to break in. You can always regenerate this file if you need it later.

For now, remove the file by typing:

```
sudo rm /usr/share/nginx/html/info.php
```

Conclusion

You should now have a LEMP stack configured on your Ubuntu 14.04 server. This gives you a very flexible foundation for serving web content to your visitors.

By Justin Ellingwood

Tutorial Series

Introduction to Nginx and LEMP on Ubuntu 14.04

This tutorial series helps sysadmins set up a new web server using the LEMP stack, focusing on Nginx setup with virtual blocks. This will let you serve multiple websites from one Droplet. You'll start by setting up your Ubuntu 14.04 server and end with multiple virtual blocks set up for your websites. An Nginx configuration guide is included at the end for reference.

[Show Tutorials](#)



We just made it easier for you to deploy faster.

[TRY FREE](#)

Related Tutorials

[How To Use Filezilla to Transfer and Manage Files Securely on your VPS](#)

[How to Get Started with FreeBSD](#)

[Initial Server Setup with Debian 9](#)

[How to Set Up SSH Keys on Debian 9](#)

[How to Set Up SSH Keys on Ubuntu 18.04](#)

133 Comments

Leave a comment...

Log In to Comment

^ [rich636856](#) April 25, 2014



0 Step Four:

The location block provided results in error:

nginx: [emerg] unknown directive "fast_cgi_split_path_info" in /etc/nginx/sites-enabled/default:56

Need to remove the underscore between fast and cgi:

fastcgi_split_path_info ^(.+\.php)(/.+)\$;

^ [rich636856](#) April 25, 2014



1 Step Two I installed mysql-server-5.6, when I run the mysql_install_db command I get the following error:

```
user@host:~$ sudo mysql_install_db
FATAL ERROR: Could not find my-default.cnf
```

If you compiled from source, you need to run 'make install' to copy the software into the correct location ready for operation.

If you are using a binary release, you must either be at the top level of the extracted archive, or pass the --basedir option pointing to that location.

Any idea what Dir I should be running this from or what --basedir option to use?

^ [arturkohut](#) November 23, 2014



1 I have the same issue when installing mysql-server-5.6 on fresh installed ubuntu 14.04.

^ [andrewdavidtanner87](#) April 27, 2014

o Brilliant tutorial, thanks Justin!

Really informative and I'm very happy you explained what every configuration means exactly.

Good work :)

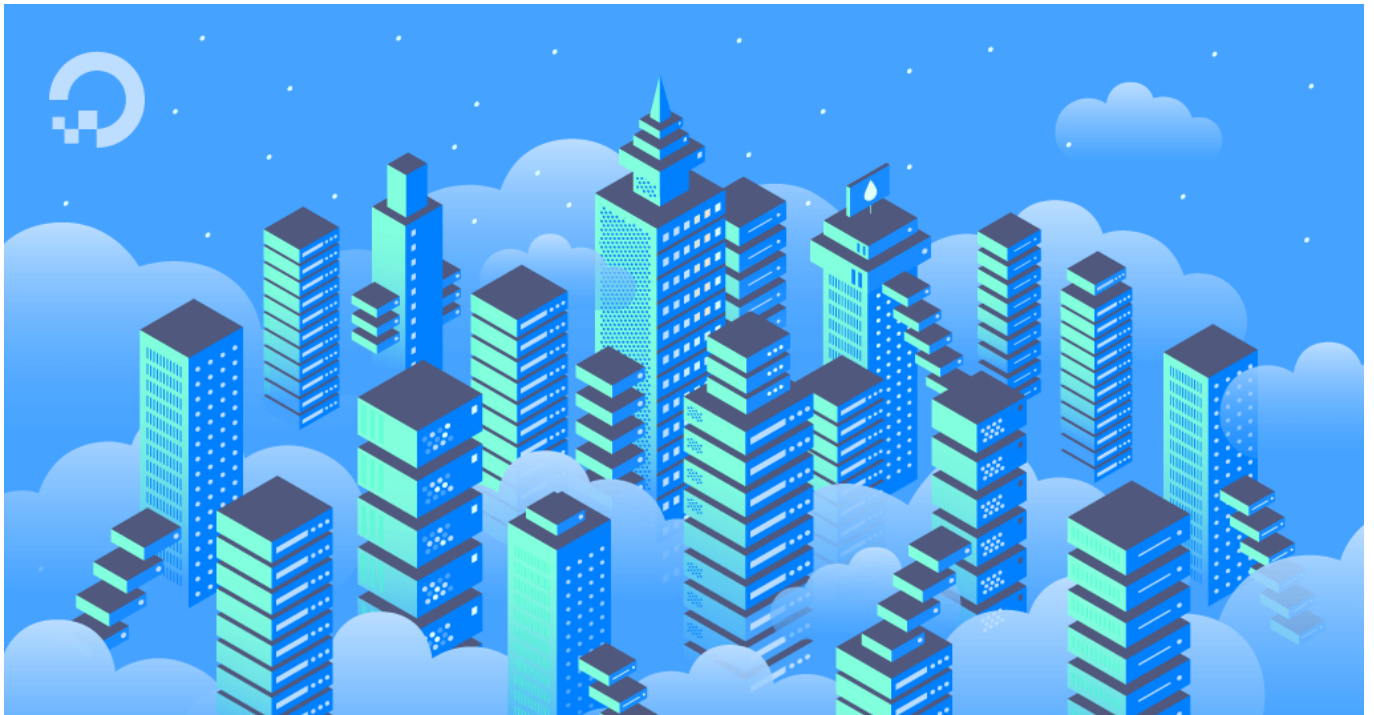
^ [davidprimadi](#) April 28, 2014

o Hii..

how to install wordpress with nginx on ubuntu 14.04?

any tutorial here?

can i use this tutorial: <https://www.digitalocean.com/community/articles/how-to-install-wordpress-with-nginx-on-ubuntu-12-04> ?



How To Install Wordpress with nginx on Ubuntu 12.04

by Etel Sverdlov

Wordpress is a free and open source website and blogging tool that uses php and MySQL. It was created in 2003 and has since then expanded to manage 22% of all the new websites created and has over 20,000 plugins to customize its functionality. This tutorial shows how to install Wordpress on a

^ [asb](#) MOD April 28, 2014

o @davidprimadi: It should be more or less the same on 14.04 as on 12.04. Let us know how it goes!

^ [davidprimadi](#) April 28, 2014

0 it's work!

^ [asb](#) MOD April 28, 2014

0 Great! Glad to hear it.

^ [brandon-](#) April 28, 2014

0 In step 5 of the 12.04 LEMP install tutorial, the instructions state to make the following change in the /etc/php5/fpm/pool.d/www.conf:

change: listen = 127.0.0.1:9000 to listen = /var/run/php5-fpm.sock

However in Step 3 this 14.04 tutorial we are not instructed to do so. Any reason why?

^ [asb](#) MOD April 28, 2014

1 @brandon:

In 14.04, php5-fpm defaults to using the socket file instead of communicating over 127.0.0.1:9000. So the reason you don't need to change it is because it should already read "listen = /var/run/php5-fpm.sock"

^ [ajgarcia](#) May 1, 2014

0 Great!! It's works!! Congrats!!

^ [donovan](#) May 4, 2014

0 Hi I get this error when I try start nginx any ideas?

2014/05/04 14:10:57 [emerg] 4272#0: "fastcgi_pass" directive is duplicate in /etc/nginx/sites-enabled/default:62

^ [netabare93](#) January 12, 2016

0 Please check your /etc/nginx/sites-enabled/default and locate to line 62 around, you will find something like

```
# With php5-cgi alone:
fastcgi_pass 127.0.0.1:9000;
# With php5-fpm:
fastcgi_pass unix:/var/run/php5-fpm.sock;
```

```
fastcgi_index index.php;  
include fastcgi_params;
```

Then comment the first case with php5-cgi alone since php5-fpm is already installed.
I came with the same problem. Hope this could help you.

^ [donovan](#) *May 4, 2014*



o nevermind got it

^ [maisdotsolutions](#) *May 10, 2014*



o Hi,

Excuse me for my ignorance, but I have a problem. I am using Ubuntu 14.04 and just create it.

In "Step Two - Install MySQL Data to Manage Site" , appears this Warning:

140510 18:00:33 [Warning] Using unique option prefix key_buffer instead of key_buffer_size is deprecated and will be removed in a future release. Please use the full name instead.

Soon after, in step "sudo mysql_install_db" appears:

Installing MySQL system tables...

140510 18:01:58 [Warning] Using unique option prefix key_buffer instead of key_buffer_size is deprecated and will be removed in a future release. Please use the full name instead.

OK

Filling help tables...

140510 18:01:58 [Warning] Using unique option prefix key_buffer instead of key_buffer_size is deprecated and will be removed in a future release. Please use the full name instead.

OK

To start mysqld at boot time you have to copy

support-files/mysql.server to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !

To do so, start the server, then issue the following commands:

```
/usr/bin/mysqladmin -u root password 'new-password'
```

```
/usr/bin/mysqladmin -u root -h MaisSolutions password 'new-password'
```

Alternatively you can run:

```
/usr/bin/mysql_secure_installation
```

which will also give you the option of removing the test databases and anonymous user created by default. This is strongly recommended for production servers.

See the manual for more instructions.

You can start the MySQL daemon with:

```
cd /usr ; /usr/bin/mysqld_safe &
```

You can test the MySQL daemon with mysql-test-run.pl

```
cd /usr/mysql-test ; perl mysql-test-run.pl
```

Please report any problems at <http://bugs.mysql.com/>

What can be? Do you have a solution? Thank you very much!

Best regards,
Leandro Naves

^ kamaln7 MOD May 12, 2014



- o @maisdotsolutions: That's fine, it's just telling you that `/etc/mysql/my.cnf` has a deprecated option, "key_buffer", which will be removed in a later version of MySQL. Just to be safe, edit `/etc/mysql/my.cnf` and replace `key_buffer` with `key_buffer_size`:

```
sudo nano /etc/mysql/my.cnf
```

^ adsf May 12, 2014



- o installation of mysql produced the following error
[Warning] Using unique option prefix key_buffer instead of key_buffer_size is deprecated and will be removed in a future release. Please use the full name instead.

what should i do?

^ adsf May 12, 2014



- o In another tutorial <https://www.digitalocean.com/community/articles/how-to-install-nginx-on-ubuntu-14-04-lts> you say

We can make sure that our web server will restart automatically when the server is rebooted by typing:

```
sudo update-rc.d nginx defaults
```

So is this step needed or not?



How To Install Nginx on Ubuntu 14.04 LTS

by Justin Ellingwood

Nginx is one of the most popular web servers in the world. It is extremely flexible and powerful and can be used to efficiently host sites and applications small or large sites and applications. In this guide we'll cover how to install and set this up on an Ubuntu 14.04 server.

^ kamaln7 MOD May 13, 2014

o @asdf:

installation of mysql produced the following error [Warning] Using unique option prefix key_buffer instead of key_buffer_size is deprecated and will be removed in a future release. Please use the full name instead.

Take a look at my response to maisdotsolutions right above your comment.

So is this step needed or not?

You don't *have* to run it, but you can run it to *make sure* nginx starts on boot, which it should do by default.

^ jasonmbaumgartner May 20, 2014

o This is a really nice and concise guide to getting up and running with a basic LEMP stack. There is one suggestion I would like to make, though.

When making changes to the nginx configuration file, it is not necessary to restart the nginx service. Instead of running the command "sudo service nginx restart", you can simply reload the configuration with this command:

```
nginx -s reload
```

It's a small minor point, but some people may find it helpful if they don't want to have the service

stopped even if it is temporarily.

Great guide!

^ [kradllit](#) May 26, 2014



0 Awesome Tutorial! Tnx!

^ [servidigitalneiva](#) June 2, 2014



0 I follow the guide, but I go to phpmyadmin throws a 404, as it would solve.
thanks

Load More Comments



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)