RELATED

How To Install nginx on CentOS 6 with yum

View

How To Install nginx on Ubuntu 12.04 LTS (Precise Pangolin)

View

// Tutorial //

How To Set Up Nginx Load Balancing

By Etel Sverdlov

About Load Balancing

Loadbalancing is a useful mechanism to distribute incoming traffic around several capable Virtual Private

Setup

The steps in this tutorial require the user to have root privileges on your VPS. You can see how to set that up in the Users Tutorial.

Prior to setting up nginx loadbalancing, you should have nginx installed on your VPS. You can install it quickly with apt-get:

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

Upstream Module

In order to set up a round robin load balancer, we will need to use the nginx upstream module. We will incorporate the configuration into the nginx settings.

Go ahead and open up your website’s configuration (in my examples I will just work off of the generic default virtual host):

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

We need to add the load balancing configuration to the file.

First we need to include the upstream module which looks like this:

```
server { location / { proxy_pass http://backend; } }
```

Restart nginx:

```
sudo service nginx restart
```

As long as you have all of the virtual private servers in place you should now find that the load balancer will begin to distribute the visitors to the linked servers equally.

Directives

The previous section covered how to equally distribute load across several virtual servers. However, there are many reasons why this may not be the most efficient way to work with data. There are several directives that we can use to direct site visitors more effectively.

Weight

One way to begin to allocate users to servers with more precision is to allocate specific weight to certain machines. Nginx allows us to assign a number specifying the proportion of traffic that should be directed to each server.

A load balanced setup that included server weight could look like this:

```
upstream backend { server backend1.example.com weight=1; server backend2.example.com weight=2;
```

Hash

IP hash allows servers to respond to clients according to their IP address, sending visitors back to the same VPS each time they visit (unless that server is down). If a server is known to be inactive, it should be marked as down. All IPs that were supposed to routed to the down server are then directed to an alternate one.

The configuration below provides an example:

```
upstream backend { ip_hash; server backend1.example.com; server backend2.example.com; server
backend3.example.com down; }
```

Max Fails

According to the default round robin settings, nginx will continue to send data to the virtual private servers, even if the servers are not responding. Max fails can automatically prevent this by rendering unresponsive servers inoperative for a set amount of time.

There are two factors associated with the max fails: max_fails and fall_timeout. Max fails refers to the maximum number of failed attempts to connect to a server should occur before it is considered inactive. Fall_timeout specifies the length of that the server is considered inoperative. Once the time expires, new attempts to reach the server will start up again. The default timeout value is 10 seconds.

A sample configuration might look like this:

This has been a short overview of simple Round Robin load balancing. Additionally, there are other ways to speed and optimize a server:

- How to Configure Nginx as a Front End Proxy for Apache
- How to Install and Configure Varnish with Apache on Ubuntu 12.04
- How to Install and Use Memcache on Ubuntu 12.04

By Etel Sverdlov

About the authors

Etel Sverdlov Author

Still looking for an answer? Ask a question Search for more help

Was this helpful? Yes No

Comments

Leave a comment...

This textbox defaults to using `Markdown` to format your answer.

You can type `!ref` in this text area to quickly search our full set of tutorials, documentation & marketplace offerings and insert the link!

Sign In or Sign Up to Comment

ntuanphuc • December 30, 2015

I followed this tutorial but a blank page appears. Do I have to config in backend servers or make any other configurations. I test with only one backend server. Here is my nginx config in http context

upstream backend { server s1.my_domain.us:80; }

server { listen 80; server_name my_demo_domain.com; location / { proxy_pass http://backend; } }

Please give me some advice,

Thanks.

@marwan, this article is for nginx and not apache - to answer your questions, yes you do have to keep the same set of files on each of the server, and yes, a database (only) server sitting behind it

Show replies   Reply

C Cor • July 13, 2020

Is it possible combine least_connect and max fails?

For example:

upstream backend { leastconn; server backend1.example.com maxfails=3 fail_timeout=15s; server backend2.example.com ; server backend3.example.com ;

Reply

rs1882828 • October 1, 2019

Do we need to do same NGNIX Config on rest slave servers?

harveyjones282 • July 10, 2019

I had my doubts about if Nginx was going to be helpful… and I was having difficulties setting it up. You solved my probelem for setting it up and another helped me realize the benefits of using Nginx for load balancing.

Reply

neamati • June 11, 2019

Excellent Tutorial

Reply

鹏鹏 左 • February 28, 2019

> There are two factors associated with the max fails: max_fails and fall_timeout.

find a typo fall_timeout should be fail_timeout

2432612431302469634d4e43793258 • November 14, 2016

Hello @kamaln7, I've followed this tutorial thoroughly but am having trouble.

In short, when I pass my ip address directly into "proxy_pass", the proxy works:

```
server {
        location / {
                proxy_pass http://01.02.03.04;
        }
}
```

When I visit my proxy computer, I can see the content from the proxy ip... but when I use an upstream directive, it doesn't

```
upstream backend {
         server 01.02.03.04;
}

server {
        location / {
                proxy_pass http://backend;
        }
}
```

When I visit my proxy computer, I am greeted with the default Nginx server page

Show replies Reply

2432612431302469634d4e43793258 • November 14, 2016

This comment has been deleted

Reply

averysmithproductions • November 14, 2016

This comment has been deleted

Reply

Load More Comments

Popular Topics

Ubuntu

Linux Basics

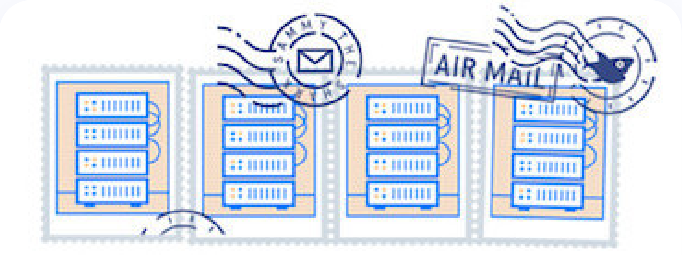
JavaScript

Python

MySQL

All tutorials →

Free Managed Hosting →



Get our biweekly newsletter

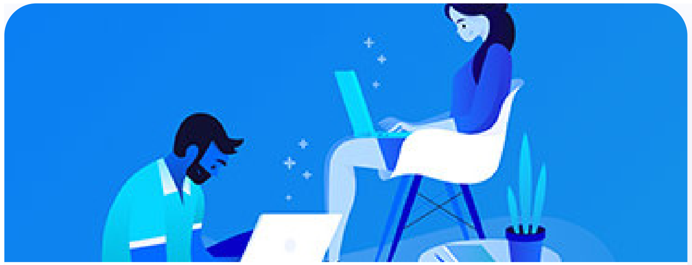
Sign up for Infrastructure as a Newsletter.

Sign up →

Hollie's Hub for Good

Working on improving health and education, reducing inequality, and spurring economic growth? We'd like to help.

Learn more →



Become a contributor

You get paid; we donate to tech nonprofits.

Learn more →

Featured on Community

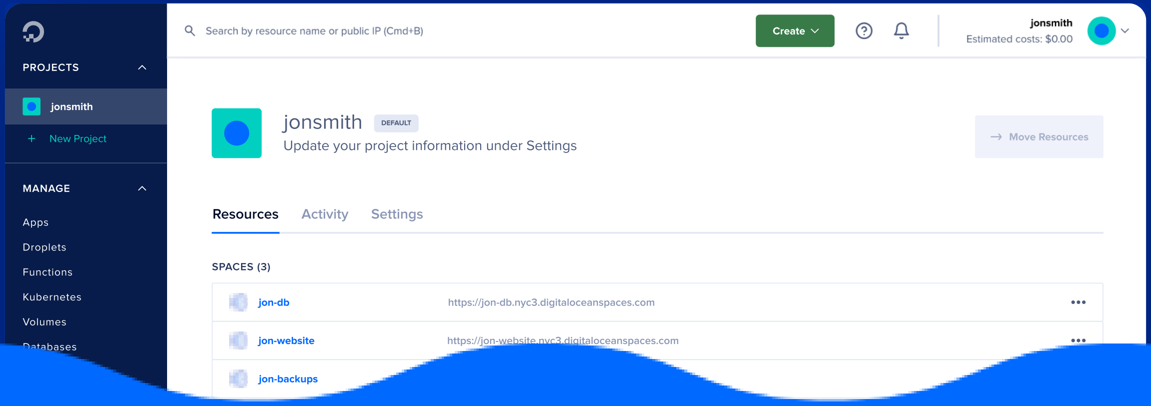
Kubernetes Course   Learn Python 3   Machine Learning in Python   Getting started with Go   Intro to Kubernetes

DigitalOcean Products

Welcome to the developer cloud

DigitalOcean makes it simple to launch in the cloud and scale up as you grow – whether you're running one virtual machine or ten thousand.

Learn more →



jonsmith

Update your project information under Settings

SPACES (3)

jon-db	https://jon-db.nyc3.digitaloceanspaces.com
jon-website	https://jon-website.nyc3.digitaloceanspaces.com
jon-backups	

Company

About

Products

Products Overview

Community

Tutorials

Solutions

Website Hosting

Contact

Support

This site uses cookies and related technologies, as described in our privacy policy, for purposes that may include site operation, analytics, enhanced user experience, or advertising. You may choose to consent to our use of these technologies, or manage your own preferences.

Partners

Channel Partners

Referral Program

Affiliate Program

Press

Legal

Security

Investor Relations

DO Impact

Cloudways

Managed Databases

Spaces

Marketplace

Load Balancers

Block Storage

Tools & Integrations

API

Pricing

Documentation

Release Notes

Uptime

Hatch Startup Program

deploy by DigitalOcean

Shop Swag

Research Program

Open Source

Code of Conduct

Newsletter Signup

Meetups

VPN

SaaS Platforms

Cloud Hosting for Blockchain

Startup Resources