connect() failed (111: Connection refused) while connecting to upstream

I am hosting my Rails app on Rackspace with nginx webserver.

When calling any Rails API, I see this message in /var/log/nginx/error.log: *49 connect() failed (111: Connection refused) while connecting to upstream, client: 10.189.254.5, server: , request: "POST /api/v1/users/sign_in HTTP/1.1", upstream: "http://127.0.0.1:3001/api/v1/users/sign_in", host: "anthemapp.com"

1. What is the upstream block?
2. What is /etc/nginx/sites-available/default? Is this where I can configure this?
3. Why am I receiving the error above?

I spent several hours with 5-6 different Rackspace tech people (they didn't know how to resolve this). This all started when I took the server into rescue mode and followed the steps here: https://community.rackspace.com/products/f/25/t/69. Once I came out of rescue mode and rebooted the server, I started receiving the error I am writing about. Tnx!

ruby-on-rails ruby-on-rails-4 nginx rackspace-cloud rackspace

asked Feb 4 at 1:27

etayluz
1,193 11 20

1 Answer

Nginx is a *reverse proxy server* - its role on your server is to accept HTTP requests and proxy them to another process on the same host. The "upstream" that the error message is talking about is referring to the bit in nginx's configuration (part of which is the `/etc/nginx/sites-available/default` file) that tells it where to send incoming requests. The error message you're seeing indicates that nginx received a request, but couldn't send it to the other process it was supposed to.

When your server rebooted, the nginx process started back up, but your Rails process -- the one that's meant to be listening on port 3001 -- did not!

How you restart the Rails process depends on the way that you started it before and the way your server is configured. It may be as simple as `cd` 'ing into your Rails application's directory on the server and running:

```
rails server -b 127.0.0.1 -p 3001 -e production -d
```

...but, to prevent problems like this from happening in the future (and to improve the performance of your Rails app!), it would be better to use some kind of production-ready Rails application server. I would recommend using Phusion Passenger because it's the most turn-key solution -- their user's guide for nginx describes installation and configuration -- but there are *plenty* of alternatives. There's a great writeup of what your options are, what they all mean, and how they relate on the top answer of this StackOverflow question.

answered Feb 4 at 13:04

Ash Wilson
7,395 2 13 33

Thans Ash! I'm using Thin as my app server. How can I ensure that Thin is running? How do I debug the configuration between nginx and Thin? – etayluz Feb 4 at 20:05

Ah, good, you're using something other than WEBrick already! I haven't used Thin specifically myself, but a quick `ps -ef | grep thin` should show you if any thin processes are running. – Ash Wilson Feb 4 at 20:26

Also: it looks like `sudo thin install` will actually write you an `/etc/init.d` script that will make sure thin runs on boot. That's handy! Here's a blog post I found that talks about it a little: jordanhollinger.com/2011/04/22/how-to-use-thin-effectivly – Ash Wilson Feb 4 at 20:28