

AWS/LAMP/Wordpress/Bootstrap Extravaganza

I am determined to make this shit fun.

Agenda

- What is AWS?
- Launch your instance
- Deploy LAMP stack
- Install Wordpress
- Bootstrap it
- Q&A



What is AWS?

- Amazon Web Services
- (It's awesome)

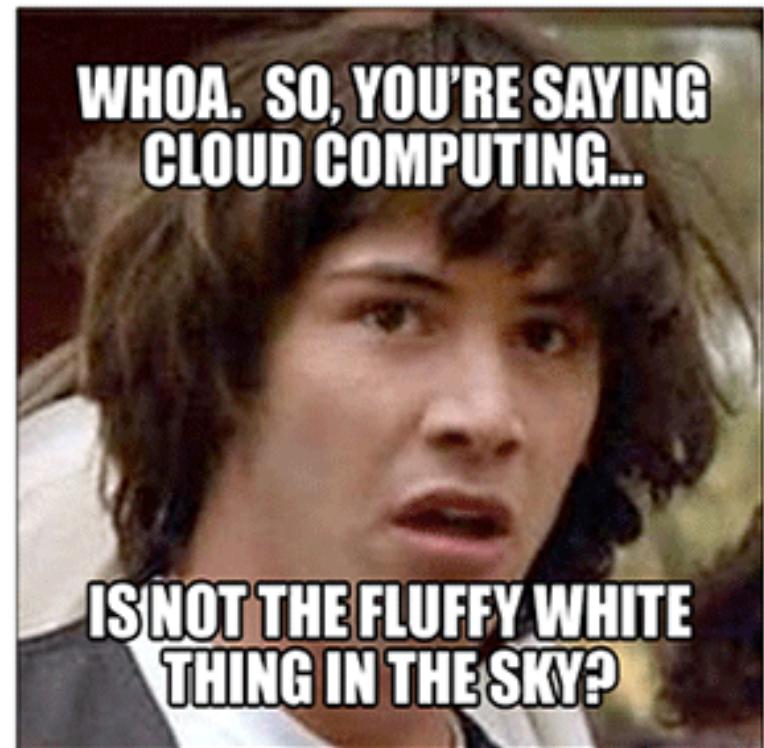


What service will you be using?

- Amazon EC2
 - Elastic Cloud Compute

*You will be building
your very own!*

Hooray!



Let's get started, yo

- Create your AWS account
 - <http://aws.amazon.com>
 - You will need:
 - Your phone
 - A credit card (but never fear!)
 - AWS EC2 Free Tier Usage Limitations
 - <http://aws.amazon.com/free/>

Time	
1	= 730.484
Month	Hour

Launch your instance

The screenshot shows the AWS Elastic Compute Cloud (EC2) console. The navigation bar at the top includes 'Services' and 'Edit' dropdowns. The left sidebar contains several sections: 'Events Dashboard', 'Tags', 'Reports', 'INSTANCES' (with sub-options 'Instances', 'Spot Requests', 'Reserved Instances'), 'IMAGES' (with sub-options 'AMIs', 'Bundle Tasks'), and 'ELASTIC BLOCK STORE' (with sub-options 'Volumes', 'Snapshots'). The main content area is titled 'Resources' and states: 'You are using the following Amazon EC2 resources in the US East (N. Virginia) region:'. It lists: '0 Running Instances', '0 Elastic IPs', '0 Volumes', '0 Snapshots', '0 Key Pairs', '0 Load Balancers', '0 Placement Groups', and '1 Security Group'. Below this is a callout box with the text: 'Optimize your resources' cost, performance and security with [AWS Trusted Advisor](#)'. A large red arrow points from the top left towards the 'Launch Instance' button. The 'Create Instance' section has a sub-section titled 'Start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.' It features a prominent blue 'Launch Instance' button. A note below it says: 'Note: Your instances will launch in the US East (N. Virginia) region'. To the right, there are two sections: 'Service Health' (which shows 'Service Status: US East (N. Virginia): This service is operating normally') and 'Scheduled Events' (which shows 'US East (N. Virginia): No events').

Pick your flavor

(*cough* RHEL based *cough*)

The screenshot shows the AWS Step 1: Choose an Amazon Machine Image (AMI) interface. The top navigation bar includes 'Services', 'Edit', and user information 'Whitney Champion' and 'N. Virg'. Below the navigation, a progress bar shows steps 1 through 6. The main section is titled 'Step 1: Choose an Amazon Machine Image (AMI)'. A large red arrow points from the top right towards the first item in the list.

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, applications, and more) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Quick Start

- My AMIs
- AWS Marketplace Free tier eligible
- Community AMIs

Free tier only (i)

Image	Name	AMI ID	Architecture	Select
	Amazon Linux AMI 2013.09.2 - ami-bba18dd2 (64-bit) / ami-d7a18dbe (32-bit)	ami-bba18dd2	64-bit <input checked="" type="radio"/> 32-bit <input type="radio"/>	Select
	Red Hat Enterprise Linux 6.4 (PV) - ami-a25415cb (64-bit) / ami-7e175617 (32-bit)	ami-a25415cb	64-bit <input checked="" type="radio"/> 32-bit <input type="radio"/>	Select
	SUSE Linux Enterprise Server 11 sp3 (PV) - ami-e8084981 (64-bit) / ami-b60948df (32-bit)	ami-e8084981	64-bit <input checked="" type="radio"/> 32-bit <input type="radio"/>	Select

Pick a ~~size~~, Micro because it's free

Services ▾ Edit ▾ Whitney Champion ▾ N. Virginia ▾

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group

Step 2: Choose an Instance Type

Currently selected: t1.micro (up to 2 ECUs, 1 vCPUs, 0.613 GiB memory, EBS only)

All instance types	Micro instances						
Micro instances	Free tier eligible						
General purpose	Micro instances are a low-cost instance option, providing a small amount of CPU resources. They are suited for lower throughput applications, and websites that require additional compute cycles periodically, but are not appropriate for applications that require sustained CPU performance. Popular uses for micro instances include low traffic websites or blogs, small administrative applications, bastion hosts, and free trials to explore EC2 functionality.						
Memory optimized	Size	ECUs	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Netw
Storage optimized	t1.micro	up to 2	1	0.613	EBS only	-	Very
Compute optimized							

Micro instances are eligible for the AWS free usage tier. For the first 12 months following your AWS sign-up date, you get up to 750 hours of micro instances each month. When your free usage tier expires or if your usage exceeds the free tier restrictions, you pay standard, pay-as-you-go service rates.

[Learn more](#) about free usage tier eligibility and restrictions

Cancel Previous Review and Launch Next: Configure Instance Details

Configure details

Screenshot of the AWS Launch Wizard Step 3: Configure Instance Details page. A large red arrow points from the bottom right towards the 'Tenancy' section.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: 1

Purchasing option: Request Spot Instances

Network: vpc-5fc5353a (172.31.0.0/16) (default) | Create new VPC

Subnet: No preference (default subnet in any Availability Zone) | Create new subnet

Public IP: Automatically assign a public IP address to your instances

IAM role: None

Shutdown behavior: Stop

Enable termination protection: Protect against accidental termination

Monitoring: Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy: Shared tenancy (multi-tenant hardware)
Additional charges will apply for dedicated tenancy.

Cancel Previous Review and Launch Next: Add Storage Feedback

Add storage

The screenshot shows the AWS CloudFormation console interface for launching a new instance. The top navigation bar includes 'Services' (dropdown), 'Edit' (dropdown), and account information ('Whitney Champion', 'N. Virginia', 'Help'). Below the navigation is a progress bar with seven steps: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance, 4. Add Storage (highlighted in orange), 5. Tag Instance, 6. Configure Security Group, and 7. Review.

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more about storage options in Amazon EC2](#).

Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination
Root	/dev/sda1	snap-b4ef17a9	8	Standard	N/A	<input checked="" type="checkbox"/>

[Add New Volume](#)

Free tier eligible customers can get up to 30 GB of EBS storage. [Learn more](#) about free usage limits, availability and usage restrictions.

At the bottom are buttons for 'Cancel', 'Previous', 'Review and Launch' (highlighted in blue), and 'Next: Tag Instance'.

Name all the things!

Screenshot of the Amazon EC2 'Tag Instance' step interface. The top navigation bar shows 'Services' selected. Below the navigation, a progress bar indicates Step 5: Tag Instance is active. A large red arrow points down to the 'Value' input field, which contains the value 'Unicorns'. The 'Key' field is set to 'Name'. A 'Create Tag' button is visible. At the bottom, there are 'Cancel', 'Previous', 'Review and Launch' (which is highlighted in blue), and 'Next: Configure Security Group' buttons.

Whitney Champion | N. Virginia | Help

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 5: Tag Instance

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. [Learn more](#) about tagging your Amazon EC2 resources.

Key (127 characters maximum)	Value (255 characters maximum)
Name	Unicorns

Create Tag (Up to 10 tags maximum)

Cancel Previous **Review and Launch** Next: Configure Security Group

Configure security group

- But... what is a security group?
 - Says what traffic can get to your instance on what ports
- What ports should we have open?
 - 80 – Apache web server
 - 22 – SSH access

Configure security group

The screenshot shows the AWS EC2 instance creation wizard at Step 6: Configure Security Group. The top navigation bar includes 'Services', 'Edit', 'Whitney Champion', 'N. Virginia', and 'Help'. Below the navigation is a breadcrumb trail: 1. Choose AMI, 2. Choose Instance Type, 3. Configure Instance, 4. Add Storage, 5. Tag Instance, 6. Configure Security Group (which is underlined), and 7. Review.

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: Create a new security group
 Select an existing security group

Security group name: INTERWEBZ

Description: LAMP stack

Type	Protocol	Port Range	Source
SSH	TCP	22	My IP 64.20.30.66/32
HTTP	TCP	80	Anywhere 0.0.0.0/0

Add Rule

At the bottom are buttons for **Cancel**, **Previous**, **Review and Launch** (which is highlighted in blue), and **Feedback**.

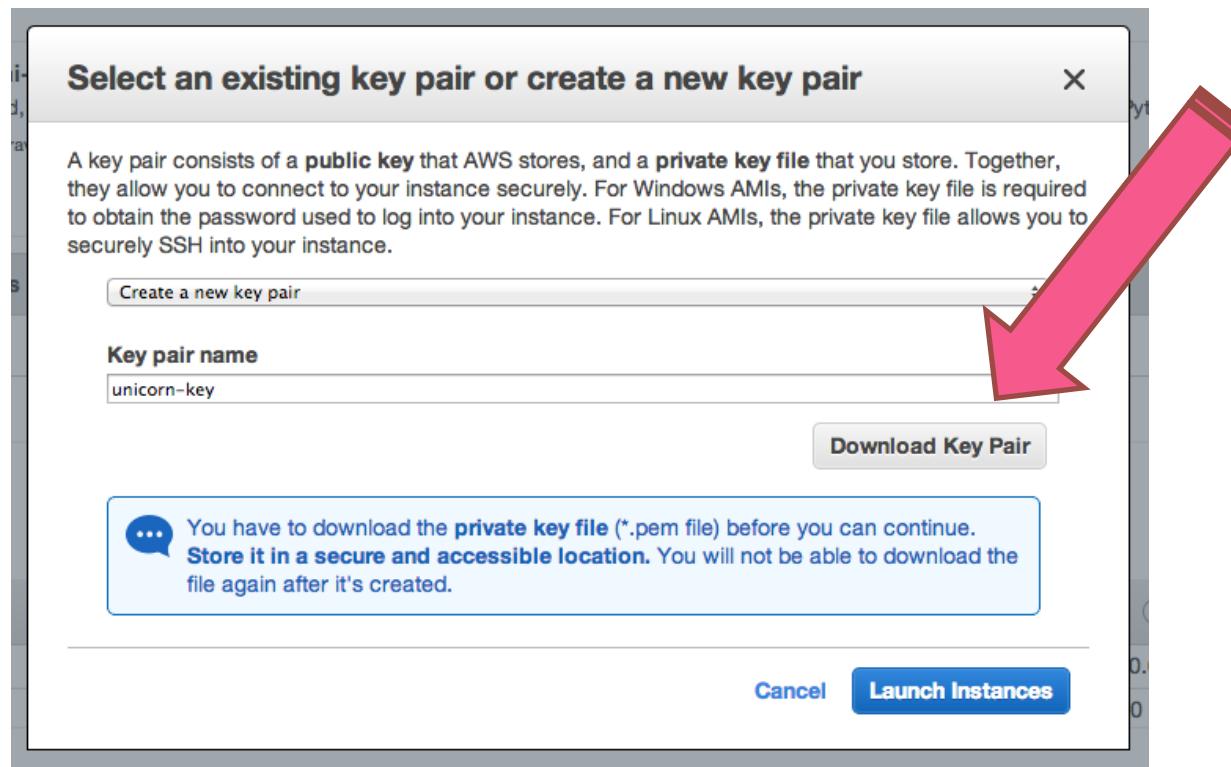
Large green arrow pointing towards the 'Review and Launch' button with the text 'SO CLOSE!' written along its path.

Review and...



Except not really!

- j/k!
- You have to have a private key to connect to your instance



Now you can launch

- For real this time



Get an Elastic IP

- Public IPs change on reboot
- Elastic IPs do not



Monitoring! It's important

- Elastic load balancers double as port monitoring



Connect to your instance

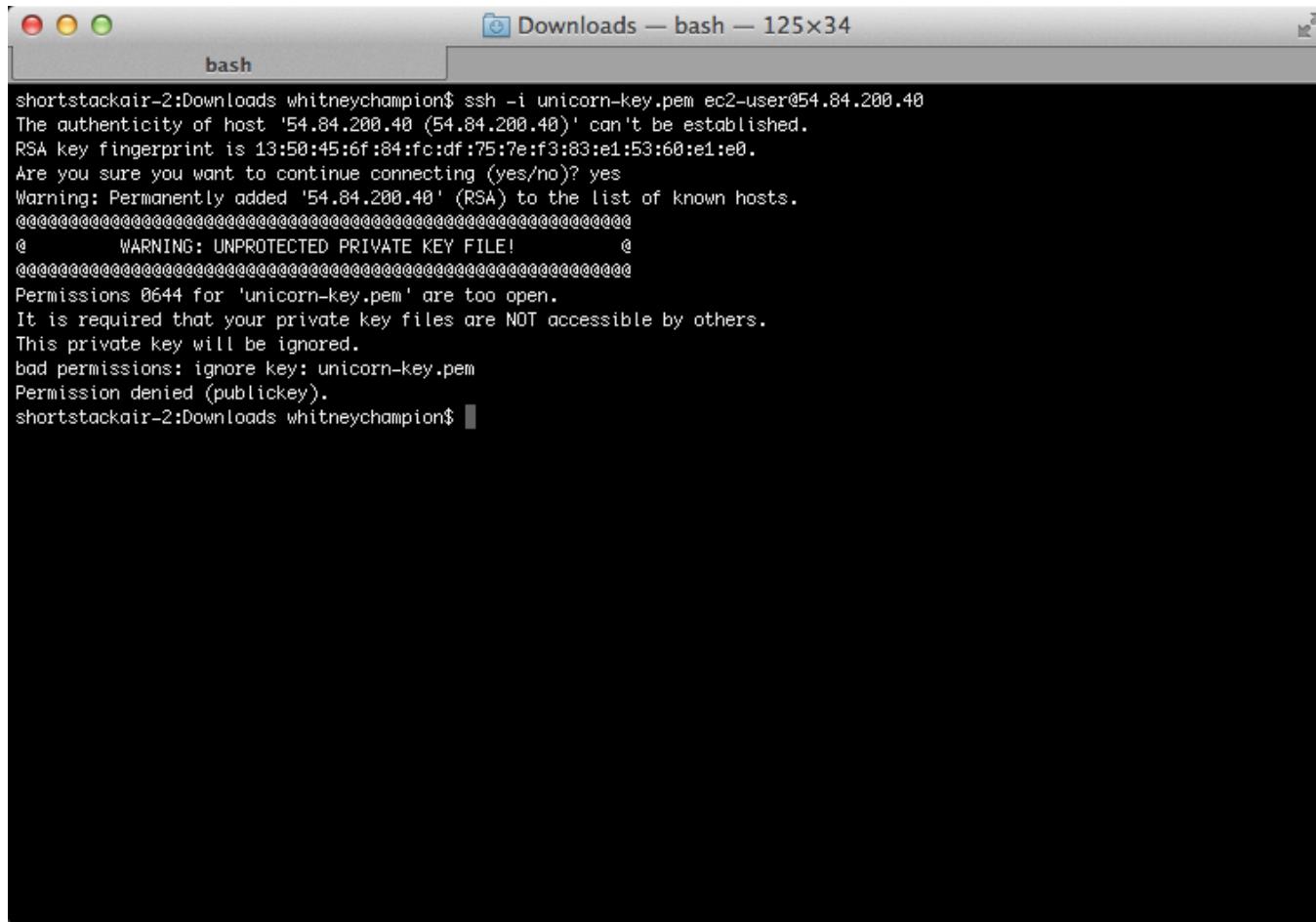
- Windows options
 - PuTTY
 - Cygwin
- Mac - Terminal

```
ssh -i /Users/{your-username}/Downloads/{your-private-key.pem} ec2-user@{your-public-ip}
```

- Linux- Terminal

```
ssh -i /home/{your-username}/Downloads/{your-private-key.pem} ec2-user@{your-public-ip}
```

Oh noes!



A screenshot of a Mac OS X terminal window titled "Downloads — bash — 125x34". The window has three colored window control buttons (red, yellow, green) at the top left. The title bar also shows the word "bash". The terminal output is as follows:

```
shortstackair-2:Downloads whitneychampion$ ssh -i unicorn-key.pem ec2-user@54.84.200.40
The authenticity of host '54.84.200.40 (54.84.200.40)' can't be established.
RSA key fingerprint is 13:50:45:6f:84:fc:df:75:7e:f3:83:e1:53:60:e1:e0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.84.200.40' (RSA) to the list of known hosts.
@@@@@@@WARNING: UNPROTECTED PRIVATE KEY FILE! @@@@_
Permissions 0644 for 'unicorn-key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: unicorn-key.pem
Permission denied (publickey).
shortstackair-2:Downloads whitneychampion$
```

chmod ftw

- Private keys need to have permissions of 600 or lower
- Based on 3 binary numbers

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
					r	w	x

	r	w	x	Total
Owner	4	2	0	6
Group	0	0	0	0
Other	0	0	0	0

```
chmod 600 {your-private-key.pem}
```

Now connect to your instance

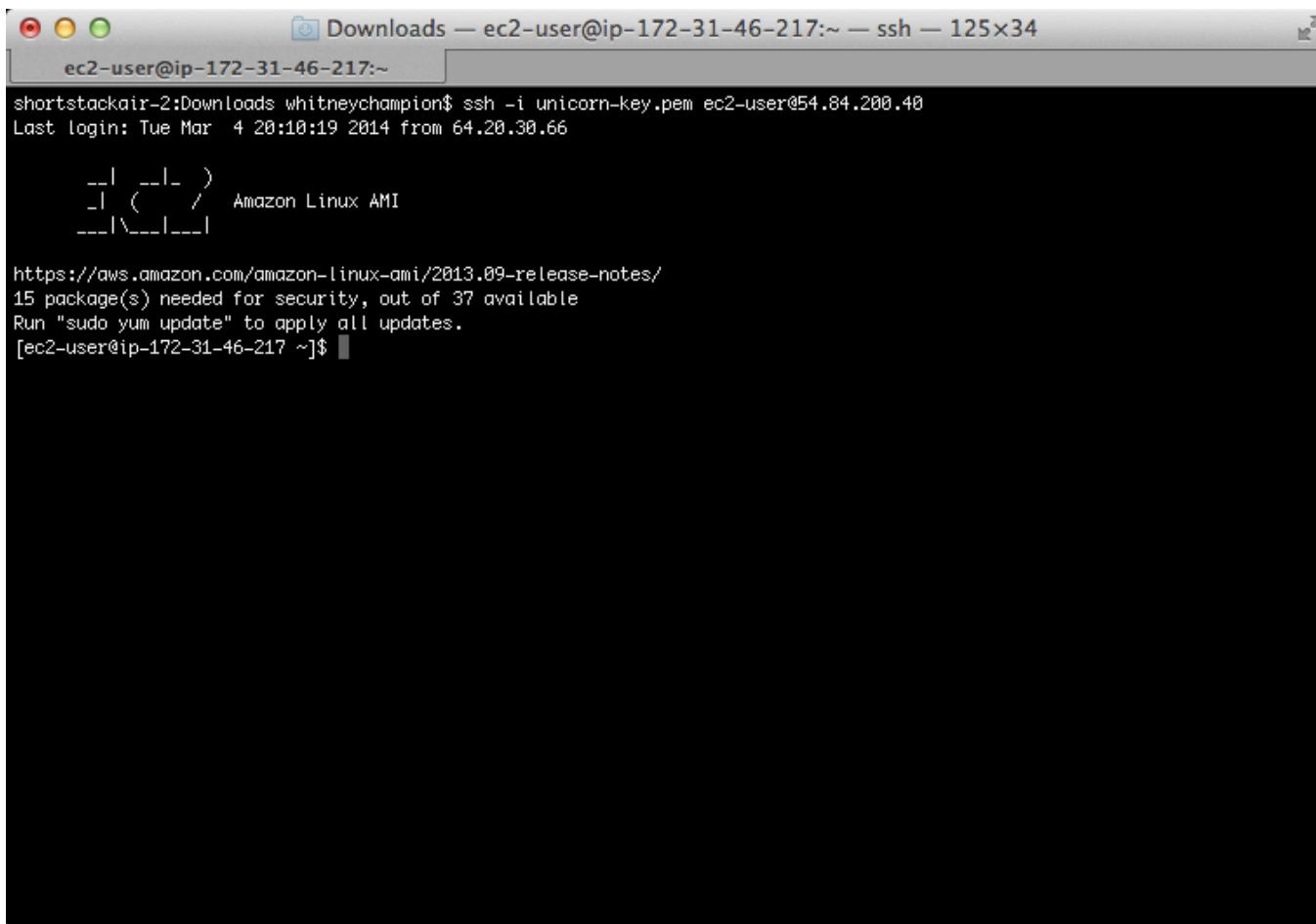
- Windows options
 - PuTTY
 - Cygwin
- Mac - Terminal

```
ssh -i /Users/{your-username}/Downloads/{your-private-key.pem} ec2-user@{your-public-ip}
```

- Linux- Terminal

```
ssh -i /home/{your-username}/Downloads/{your-private-key.pem} ec2-user@{your-public-ip}
```

Success!



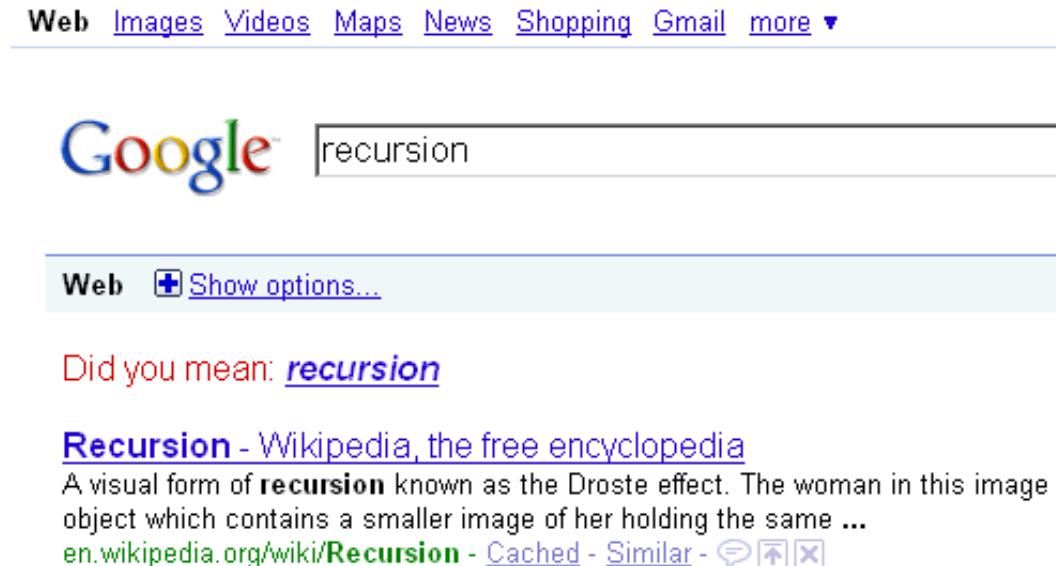
A screenshot of a terminal window titled "Downloads — ec2-user@ip-172-31-46-217:~ — ssh — 125x34". The window shows a successful SSH session to an Amazon Linux AMI instance. The session starts with the command "ssh -i unicorn-key.pem ec2-user@54.84.200.40", followed by the last login information: "Last login: Tue Mar 4 2010:19 2014 from 64.20.30.66". Below this, the Amazon Linux logo is displayed as a series of ASCII characters forming a stylized unicorn head. The next few lines show the user navigating to the "release-notes" directory on the Amazon Linux AMI website and checking for updates: "https://aws.amazon.com/amazon-linux-ami/2013.09-release-notes/", "15 package(s) needed for security, out of 37 available", and "Run \"sudo yum update\" to apply all updates.". The session ends with the prompt "[ec2-user@ip-172-31-46-217 ~]\$".

```
shortstackair-2:Downloads whitneychampion$ ssh -i unicorn-key.pem ec2-user@54.84.200.40
Last login: Tue Mar 4 2010:19 2014 from 64.20.30.66

 _\ _ \|_) 
 _| ( /   Amazon Linux AMI
 ___| \_\_|__|_|
https://aws.amazon.com/amazon-linux-ami/2013.09-release-notes/
15 package(s) needed for security, out of 37 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-46-217 ~]$
```

Updates and security patches

- What is yum?
 - Yellowdog Updater, Modified
- What is an RPM?
 - Red Hat Package Manager
 - RPM Package Manager



Web Images Videos Maps News Shopping Gmail more ▾

recursion

Web Show options...

Did you mean: [recursion](#)

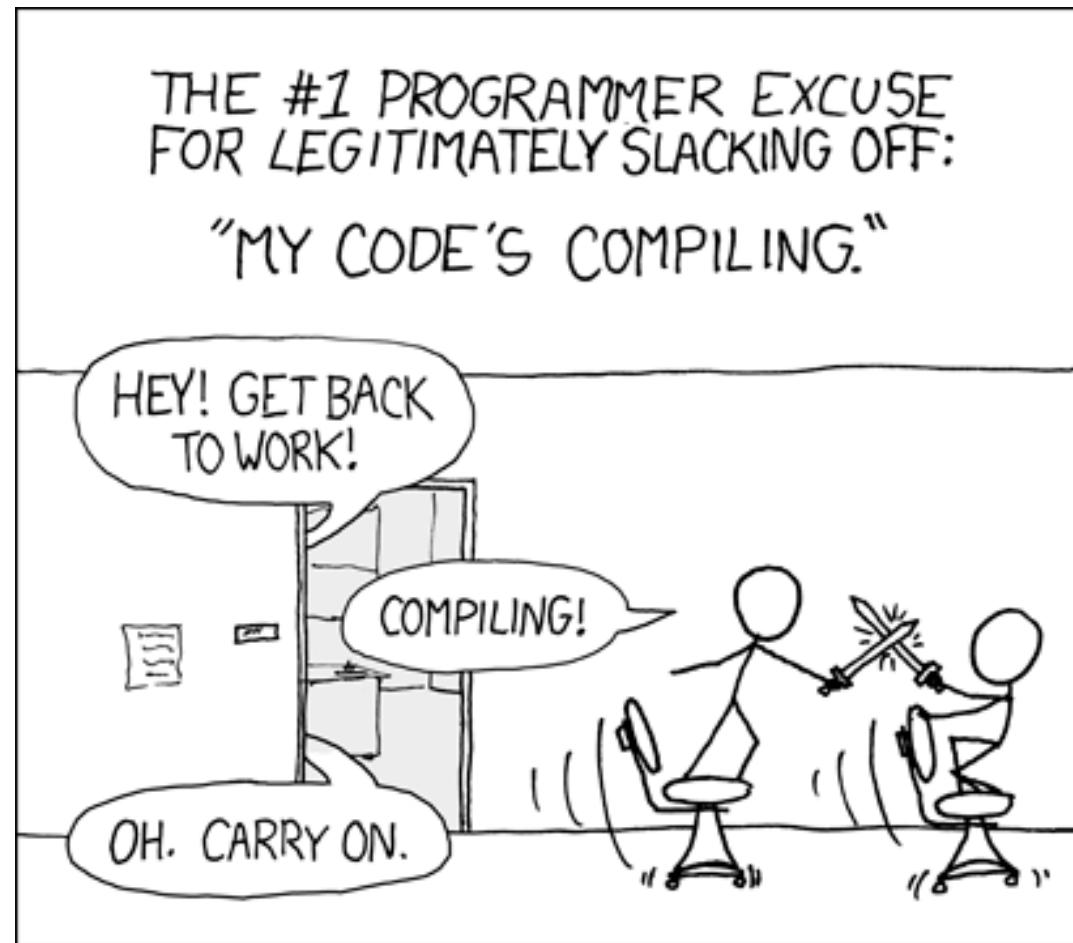
[Recursion - Wikipedia, the free encyclopedia](#)
A visual form of **recursion** known as the Droste effect. The woman in this image is object which contains a smaller image of her holding the same ...
en.wikipedia.org/wiki/Recursion - Cached - Similar -

Update your system

```
$ sudo yum update
```

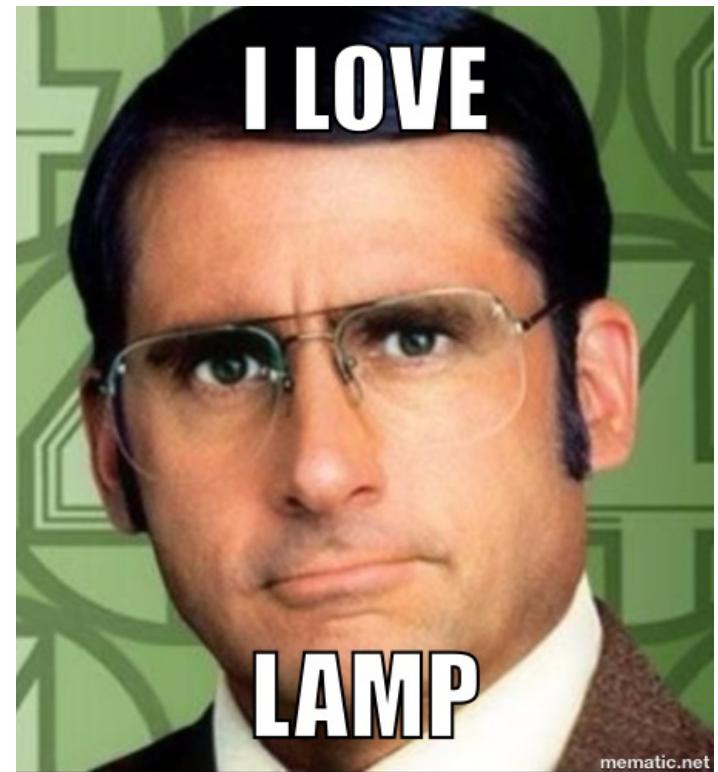
Easy, right? Right!

Now we wait



LAMP Stack

- ✓ Linux – Done!
- Apache – Let's do dis...
- MySQL
- PHP



Apache

```
$ sudo yum install httpd  
$ sudo service httpd start  
$ sudo chkconfig httpd on
```

- What did we just do?
 - Installed Apache
 - Started the web server
 - Made sure Apache was set to start on boot

Check 'er out

- Browser → <http://{your-public-ip}>

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

For information on Amazon Linux AMI , please visit the [Amazon AWS website](#).

If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:



Questions so far?



LAMP Stack

- ✓ Linux – Done!
- ✓ Apache – Done!
- MySQL – Onward!
- PHP – Let's knock this one out, too!

Let's first check our resources...

\$ free



Create a swap file

```
$ sudo dd if=/dev/zero of=/swapfile bs=1M count=1024
$ sudo chmod 600 /swapfile
$ sudo mkswap /swapfile
$ sudo swapon /swapfile
$ sudo vim /etc/fstab
    /swapfile swap swap defaults 0 0
$ sudo mount -a
```

- What did we just do?
 - Made a 1G file full of 0's
 - Made it accessible only to root
 - Turned that file into swap space
 - Enabled our new swap space
 - Enabled on reboot
 - Tested our changes (always... always do this)

Install MySQL & PHP

```
$ sudo yum install mysql mysql-server php php-xml php-mysql  
$ sudo service mysqld start  
$ sudo chkconfig mysqld on  
$ sudo service httpd restart
```

- What did we just do?
 - Installed MySQL command line tools
 - Installed MySQL server
 - Installed PHP and PHP extensions
 - Started MySQL server, and set it to start on boot
 - Restarted Apache, because it needs to know PHP is now installed and available

Configure MySQL

```
$ sudo /usr/bin/mysql_secure_installation
```

Enter current password for root (enter for none):

Set root password? [Y/n] Y

Remove anonymous users? [Y/n] Y

Disallow root login remotely? [Y/n] Y

Remove test database and access to it? [Y/n] Y

Reload privilege tables now? [Y/n] Y

*****MAKE A NOTE OF ROOT PASSWORD*****

Configure Wordpress database

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

```
mysql> show databases;
```

```
mysql> create database wordpress;
```

```
mysql> grant usage on *.* to wpadmin@localhost  
identified by 'googlerules';
```

```
mysql> grant all privileges on wordpress.* to  
wpadmin@localhost;
```

MAKE A NOTE OF YOUR CREDENTIALS

Install phpMyAdmin

- What is phpMyAdmin?
 - A MySQL administration tool

```
$ wget http://packages.sw.be/rpmforge-release/rpmforge-
      release-0.5.2-2.el5.rf.i386.rpm
$ sudo rpm -Uvh rpmforge-release-0.5.2-2.el5.rf.i386.rpm
$ sudo yum install phpmyadmin
```

Configure phpMyAdmin

```
$ sudo vim /etc/httpd/conf.d/phpmyadmin.conf  
Allow from 64.20.30.66
```

```
$ sudo vim /etc/httpd/conf/httpd.conf  
<Directory /var/www/html>  
    AllowOverride All
```

```
$ sudo vim /usr/share/phpmyadmin/config.inc.php  
$cfg['blowfish_secret'] = 'magical-unicorns';
```

```
$ sudo service httpd reload
```

Check 'er out

- Browser →
<http://{your-public-ip}/phpMyAdmin>
- Log in with your Wordpress MySQL username
and password

LAMP Stack

- ✓ Linux – Done!
 - ✓ Apache – Done!
 - ✓ MySQL – Done!
 - ✓ PHP – Done!
-
- ✓ *Plus* you're ready for Wordpress and have a sweet MySQL GUI

Let's re-cap...

So far, you have built...

- An EC2 instance in the AWS cloud with:
 - Amazon's flavor of Linux
 - An Apache web server
 - A swap partition
 - A MySQL server with a Wordpress-ready database

Onward!



Let's revisit port monitoring



Time for Wordpress

```
$ cd /var/www/html  
$ sudo wget http://wordpress.org/latest.tar.gz  
$ sudo tar -xvf latest.tar.gz --strip-components 1  
$ sudo chown -R apache:apache /var/www/html
```

- What did we just do?
 - Downloaded Wordpress
 - Unzipped it to the proper directory so it will show up on our web server
 - Made sure all Wordpress files are owned by Apache

Install Wordpress

- Browser → <http://{your-public-ip}/wp-admin/install.php>

The screenshot shows the WordPress installation setup page. At the top is the classic WordPress logo. Below it, a message encourages users to enter their database connection details if they're not sure, to contact their host. The form contains five input fields with accompanying descriptions:

Database Name	<input type="text" value="wordpress"/>	The name of the database you want to run WP in.
User Name	<input type="text" value="username"/>	Your MySQL username
Password	<input type="text" value="password"/>	...and MySQL password.
Database Host	<input type="text" value="localhost"/>	You should be able to get this info from your web host, if localhost does not work.
Table Prefix	<input type="text" value="wp_"/>	If you want to run multiple WordPress installations in a single database, change this.

At the bottom left is a prominent "Submit" button.

Twitter Bootstrap

- What is bootstrap?



Get some plugins

- Browser → <http://{your-public-ip}/wp-admin>
- Plugins → Add New
 - Bootstrap Shortcodes
 - CPT Bootstrap Carousel
- Install and activate
- Shiny plug-and-play goodness awaits

Let's make it pretty!

- Appearance → Themes → Add New



Tinker

- Make a carousel and embed it in a page
 - [image-carousel]
- Make your homepage a static page instead of blog posts
- Add a few pages and menu items
- Test out the shortcodes
- Add some homepage widgets
 - Hint: You can use the sweet bootstrap shortcodes inside a “Text” widget
- Play with the CSS inside of the theme editor

Now you have...

- A server
- A website
- A responsive website
- **AND IT'S FREE FOR A YEAR.**
- **AND YOU BUILT IT.**
- **HELLO FREE TRAINING FOR A YEAR.**

Profit!



Here's to choosing the hard way :)

Questions?

