

Create Apache Server Instance via AWS of amazon.com

<http://ec2-52-23-162-119.compute-1.amazonaws.com/index.html>

A. Initial Setup for creating instance

Launch Instance: Create new instance

Step 1. Choose amazon Linux free tier Ubuntu 14.04 64 bit

Step 2. MicroInstance ,then Choose Next: Configure Instance Details

Step 3. Next: Configure Instance Details

- Number of Instances: 1
- Advanced Details: use Defaults for now

Step 4. Add Storage

- Volume Type: choose General Purpose (slowest ...sufficient enough for demo hopefully)
- DeleteOnTermination: click (Let's keep it clean)

Step 5. Next: Tag Instance

You can leave it alone or choose something simple

I tried the followings:

- Key: danbohelloworld
- Value: danboaws1

Step 6. Next: Configure Security Group

Choose "Create a new Security group" with value as the followings:

- Security group name: test
- List of Types: I kept and created the followings
 - Type-Protocol-Port-Range
 - SSH-TCP-22
 - HTTP-TCP-80

Step 7. (Almost there) : Review and Launch

- Review
- Launch (Let's create a pair of keys so we can use to ssh)

Create Apache Server Instance via AWS of amazon.com

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

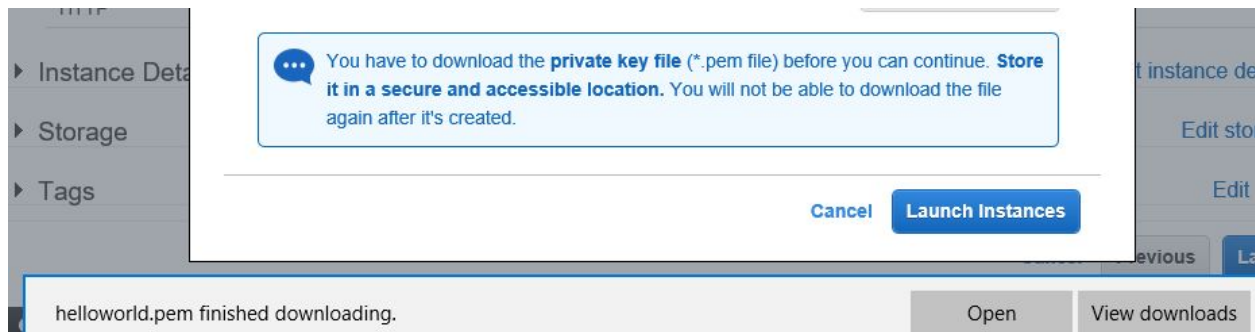
Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Create a new key pair

Key pair name

helloworld

Note: Use the above values, Click Download Key Pair and Save the private key file (helloworld.pem)



don't lose the helloworld.pem!

Note:

Step 8. (Last Step, Yeh!) Launch the instance

B. Install the Apache2 and PHP5

Step 1. Locate the instance we created from A via the tag

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
		i-00fb7b178e63c0ae4	t1.micro	us-east-1d	running	2/2 checks successful

Instance: i-00fb7b178e63c0ae4 Public DNS: ec2-52-23-162-119.compute-1.amazonaws.com

Description Status Checks Monitoring Tags

Add/Edit Tags

Key	Value
danbohelloworld	danboaws1

Create Apache Server Instance via AWS of amazon.com

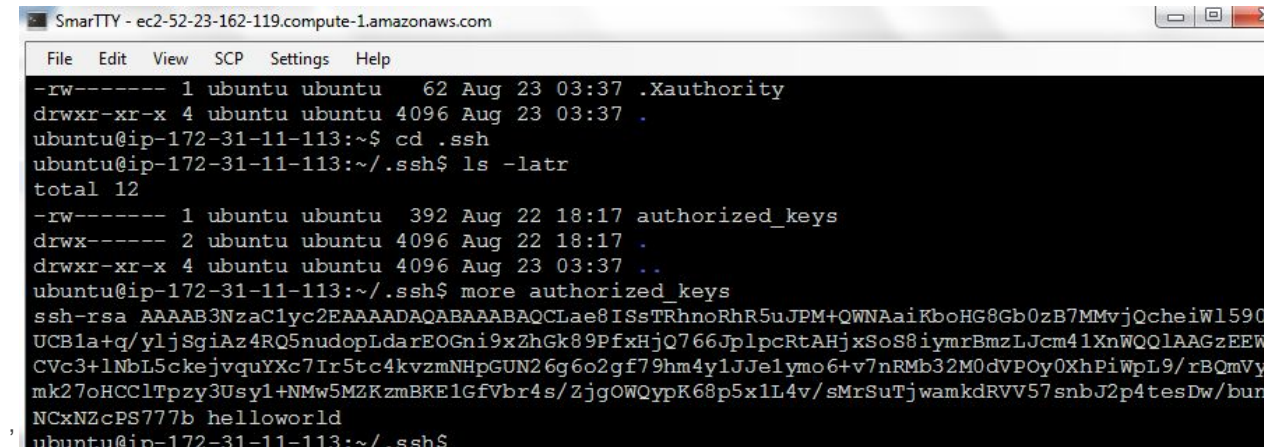
Step2. Ssh to the instance via using SmartTTY

Hostname : ec2-52-23-162-119.compute-1.amazonaws.com

Username : ubuntu

OpenSSH keyfile : helloworld.pem

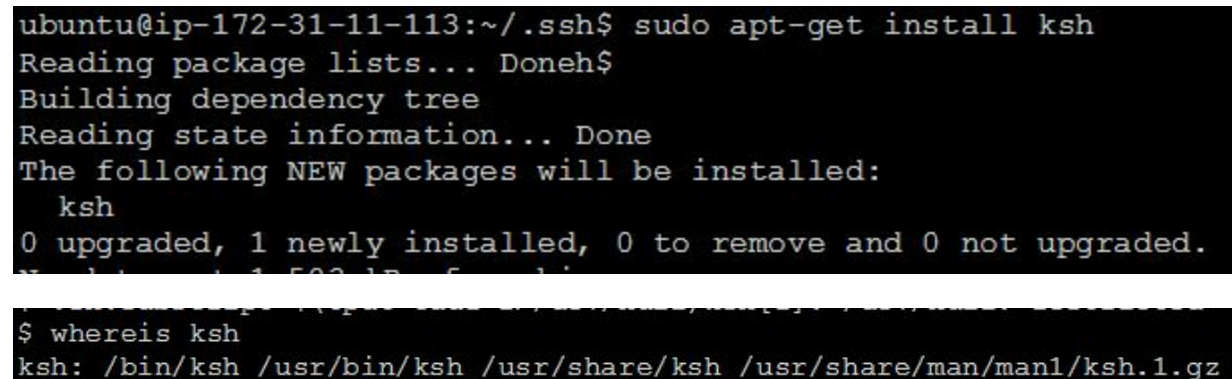
Or if having ssh terminal client, then use this cmd "ssh -I helloworld.pem ubuntu@52.23.162.119"

A screenshot of a SmartTTY terminal window. The title bar reads "SmartTTY - ec2-52-23-162-119.compute-1.amazonaws.com". The terminal shows the following commands and output:

```
File Edit View SCP Settings Help
-rw----- 1 ubuntu ubuntu 62 Aug 23 03:37 .Xauthority
drwxr-xr-x 4 ubuntu ubuntu 4096 Aug 23 03:37 .
ubuntu@ip-172-31-11-113:~$ cd .ssh
ubuntu@ip-172-31-11-113:~/.ssh$ ls -latr
total 12
-rw----- 1 ubuntu ubuntu 392 Aug 22 18:17 authorized_keys
drwx----- 2 ubuntu ubuntu 4096 Aug 22 18:17 .
drwxr-xr-x 4 ubuntu ubuntu 4096 Aug 23 03:37 ..
ubuntu@ip-172-31-11-113:~/.ssh$ more authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCLae8ISsTRhnoRhR5uJPM+QWNAaiKboHG8Gb0zB7MMvjQcheiWl590
UCB1a+q/yljSgiAz4RQ5nudopLdarEOGni9xZhGk89PfxHjQ766JplpcRtAHjxSoS8iymrBmzLJcm41XnWQQ1AAGzEEW
CVc3+1NbL5ckejvquYXc7Ir5tc4kvzmNHpGUN26g6o2gf79hm4y1JJelymo6+v7nRMb32M0dVPOy0XhPiWpL9/rBQmVy
mk27oHCClTpzy3Usy1+NMw5MZKzmBKE1GfVbr4s/ZjgOWQypK68p5x1L4v/sMrSuTjwamkdRVV57snbJ2p4tesDw/bun
NCxNZcPS777b helloworld
ubuntu@ip-172-31-11-113:~/.ssh$
```

Step3. After ssh to the AWS instance virtual host, Let's get the Korn Shell installed!

CMD: sudo apt-get install ksh

A screenshot of a terminal window showing the installation of ksh. The terminal output is as follows:

```
ubuntu@ip-172-31-11-113:~/.ssh$ sudo apt-get install ksh
Reading package lists... Doneh$
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  ksh
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
$ whereis ksh
ksh: /bin/ksh /usr/bin/ksh /usr/share/ksh /usr/share/man/man1/ksh.1.gz
```


Step4. Once we have ksh, let's put some scripts together

Step4.1 Let's make our eyes comfortable by adjusting smartTTY's editor's settings.

Create Apache Server Instance via AWS of amazon.com

File & Keyboard Visual Settings Editor Settings

XMing settings

XMing executable: (use pre-packaged XMing provided with SmartTTY) 

XMing arguments: -multiwindow

Visual settings


Vertical scrollbar: Always visible


Horizontal scrollbar: Hover mouse to show

Page header position: Bottom

Session header position: Left

Terminal bell sound: None

Terminal font (monospace): Courier New, Size = 14.25 

Default text color: 

Step 4.2. Let's prepare a script to install & bounce apache2 server

```
$ more installApache.ksh
#!/bin/ksh
echo "let us install apache2 and php5"
sudo apt-get update
sudo apt-get install php5
sudo apt-get update
sudo apt-get install apache2
sudo apt-get update
sudo apt-get install libapache2-mod-php5
sudo apt-get update
echo "finish installing apache2 and php5"
exit
```

```
$ ls -latr *.ksh
-rwxr-xr-x 1 ubuntu ubuntu 71 Aug 23 04:06 startApache.ksh
-rwxr-xr-x 1 ubuntu ubuntu 282 Aug 23 04:10 installApache.ksh
-rwxr-xr-x 1 ubuntu ubuntu 69 Aug 23 04:13 stopApache.ksh
```

Create Apache Server Instance via AWS of amazon.com

```
./stopApache.ksh
let's stop apache2
* Stopping web server apache2
*
$ ./startApache.ksh
let's start apache2
* Starting web server apache2
*
$ ps -eaf |grep apache2
root      4781      1  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4784    4781  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4785    4781  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4786    4781  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4787    4781  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
www-data  4788    4781  0  04:13 ?        00:00:00 /usr/sbin/apache2 -k start
ubuntu    4793   2248  0  04:13 pts/0    00:00:00 grep apache2
```

Create Apache Server Instance via AWS of amazon.com

C. Prepare the helloWorld page

```
$ pwd
/var/www
$ sudo chmod -R 777 html
$ cd html
$ ls -latr
total 20
drwxr-xr-x 3 root root 4096 Aug 23 04:11 ..
-rwxrwxrwx 1 root root 11510 Aug 23 04:11 index.html
drwxrwxrwx 2 root root 4096 Aug 23 04:11 .
```

<http://ec2-52-23-162-119.compute-1.amazonaws.com/index.html>

← → ↻  ec2-52-23-162-119.compute-1.amazonaws.com/index.html

From the Earth: Hello World!

From the Martians: Hey Human on the Earth !

Create Apache Server Instance via AWS of amazon.com

D. Now let's clone the instance and set up ssl on the cloned instance.

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)
danbohelloworldssl	danboaws2

Create Tag (Up to 50 tags maximum)

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: testssl
Description: aws2 instance with ssl

Type	Protocol	Port Range	Source
SSH	TCP	22	Anywhere 0.0.0.0/0
HTTP	TCP	80	Anywhere 0.0.0.0/0
HTTPS	TCP	443	Anywhere 0.0.0.0/0

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

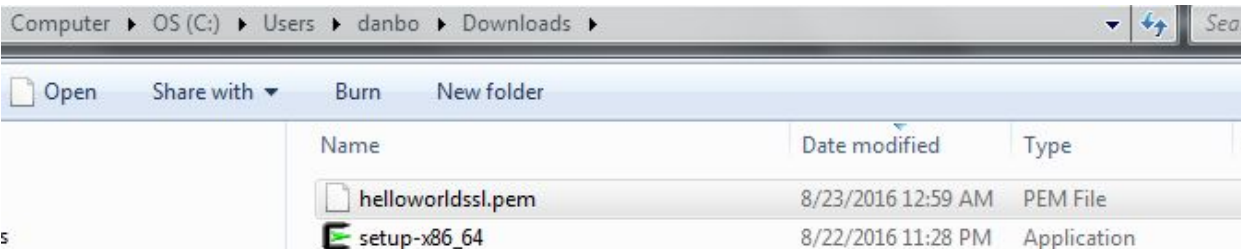
Note: The selected key pair will be added to the set of keys authorized for this instance. [Learn more](#) about [removing existing key pairs from a public AMI](#).

Create a new key pair

Key pair name

helloworldssl

Download Key Pair



Create Apache Server Instance via AWS of amazon.com

helloworldssl	i-0e4c1037e8d978e94	t1....	us-east-1d	running	Initializing	None	ec2-54-197-185-184.co...	54.
Instance ID	i-0e4c1037e8d978e94			Public DNS		ec2-54-197-185-184.compute-1.amazonaws.com		
Instance state	running			Public IP		54.197.185.184		
Instance type	t1.micro			Elastic IPs				
Private DNS	ip-172-31-8-192.ec2.internal			Availability zone		us-east-1d		
Private IPs	172.31.8.192			Security groups		testssl, view rules		
Secondary private IPs				Scheduled events		No scheduled events		
VPC ID	vpc-23661c44			AMI ID		ubuntu/images/ebs-ssd/ubuntu-trusty-14.04-amd64-server-20160627 (ami-c93a83de)		
Subnet ID	subnet-6ed81b27			Platform		-		
Network interfaces	eth0			IAM role		-		
Source/dest. check	True			Key pair name		helloworldssl		

E. Let's work on helloworldssl instance

E.1 ssh to helloworldssl instance

Hostname : **ec2-54-197-185-184.compute-1.amazonaws.com**

Username : ubuntu

OpenSSH keyfile : helloworldssl.pem

E.2 use FileZilla to transfer files so we can use the scripts created for helloworld instance

Select Entry:

My Sites

danbovirtual

helloworld

helloworldssl

General

Advanced

Transfer Settings

Charset

Host:

7-185-184.compute-1.amazonaws.com

Port:

22

Protocol:

SFTP - SSH File Transfer Protocol

Logon Type:

Key file

User:

ubuntu

Key file:

C:\Users\danbo\Downloads\keys\hellc

Browse...

Now we copied helloworld instance's script dir to helloworldssl's instance

E.3 install the apache2 and clone the index.html

Create Apache Server Instance via AWS of amazon.com

```
ubuntu@ip-172-31-8-192:~/scripts$ ls -latr
total 24
-rw-rw-r-- 1 ubuntu ubuntu 74 Aug 23 05:19 bounceApache.ksh
-rw-rw-r-- 1 ubuntu ubuntu 282 Aug 23 05:19 installApache.ksh
-rw-rw-r-- 1 ubuntu ubuntu 71 Aug 23 05:19 startApache.ksh
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 23 05:19 .
-rw-rw-r-- 1 ubuntu ubuntu 69 Aug 23 05:19 stopApache.ksh
drwxr-xr-x 5 ubuntu ubuntu 4096 Aug 23 05:19 ..
ubuntu@ip-172-31-8-192:~/scripts$ sudo chmod -R 775 *.*
ubuntu@ip-172-31-8-192:~/scripts$ date
Tue Aug 23 05:22:28 UTC 2016
ubuntu@ip-172-31-8-192:~/scripts$ ls -latr
total 24
-rwxrwxr-x 1 ubuntu ubuntu 74 Aug 23 05:19 bounceApache.ksh
-rwxrwxr-x 1 ubuntu ubuntu 282 Aug 23 05:19 installApache.ksh
-rwxrwxr-x 1 ubuntu ubuntu 71 Aug 23 05:19 startApache.ksh
drwxrwxr-x 2 ubuntu ubuntu 4096 Aug 23 05:19 .
-rwxrwxr-x 1 ubuntu ubuntu 69 Aug 23 05:19 stopApache.ksh
drwxr-xr-x 5 ubuntu ubuntu 4096 Aug 23 05:19 ..
```

```
ubuntu@ip-172-31-8-192:~/scripts$ ./installApache.ksh
let us install apache2 and php5
Ign http://us-east-1.ec2.archive.ubuntu.com trusty InRelease
```

Issue cmd: `sudo chmod -R 777 /var/www/html` so we can copy the index.html

```
ubuntu@ip-172-31-8-192:/var/www$ sudo chmod -R 777 html
ubuntu@ip-172-31-8-192:/var/www$ ls -latr
total 12
drwxr-xr-x 13 root root 4096 Aug 23 05:25 ..
drwxr-xr-x 3 root root 4096 Aug 23 05:25 .
drwxrwxrwx 2 root root 4096 Aug 23 05:25 html
```

Get the index.html ready for the apache2 on helloworldssl instance

```
ubuntu@ip-172-31-8-192:/var/www/html$ ls -latr
total 24
drwxr-xr-x 3 root root 4096 Aug 23 05:25 ..
-rwxrwxrwx 1 root root 11510 Aug 23 05:25 index.html
drwxrwxrwx 2 root root 4096 Aug 23 05:34 .
-rw-rw-r-- 1 ubuntu ubuntu 567 Aug 23 05:34 indexssl.html
ubuntu@ip-172-31-8-192:/var/www/html$ cp index.html index.html.orig
ubuntu@ip-172-31-8-192:/var/www/html$ mv indexssl.html index.html
```

Create Apache Server Instance via AWS of amazon.com

Now the initial page is UP

<http://ec2-54-197-185-184.compute-1.amazonaws.com/index.html>



F. Create the self signed certificate

F.1 Generate a Private Key

```
openssl genrsa -des3 -out server.key 1024
```

F.2 Generate a CSR (Certificate Signing Request)

```
openssl req -new -key server.key -out server.csr
```

F3: Remove Passphrase from Key

```
cp server.key server.key.org
```

```
openssl rsa -in server.key.org -out server.key
```

```
ubuntu@ip-172-31-8-192:~/ssl$ ls -latr
total 20
drwxr-xr-x 6 ubuntu ubuntu 4096 Aug 23 06:35 ..
-rwxrwxrwx 1 root    root    708 Aug 23 06:39 server.csr
-rwxrwxrwx 1 ubuntu ubuntu  963 Aug 23 06:41 server.key.org
drwxrwxrwx 2 ubuntu ubuntu 4096 Aug 23 06:41 .
-rwxrwxrwx 1 root    root    887 Aug 23 06:43 server.key
```

F.4 Generate a Self-Signed Certificate

```
sudo openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

```
ubuntu@ip-172-31-8-192:~/ssl$ sudo openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
Signature ok
subject=/C=us/ST=pa/L=malvern/O=aws/OU=aws/CN=ec2-54-197-185-184.compute-1.amazonaws.com/emailAddress=dw
Getting Private key
```

Create Apache Server Instance via AWS of amazon.com

```
drwxr-xr-x 6 ubuntu ubuntu 4096 Aug 23 06:35 ..
-rwxrwxrwx 1 root root 708 Aug 23 06:39 server.csr
-rwxrwxrwx 1 ubuntu ubuntu 963 Aug 23 06:41 server.key.org
-rwxrwxrwx 1 root root 887 Aug 23 06:43 server.key
drwxrwxrwx 2 ubuntu ubuntu 4096 Aug 23 06:46 .
-rw-r--r-- 1 root root 960 Aug 23 06:46 server.crt
```

F.5 Enable mod_ssl

```
sudo a2enmod ssl
```

```
sudo service apache2 restart
```

F.6. We got an issue:

```
https://ec2-54-197-185-184.compute-1.amazonaws.com:443/index.html
```

This site can't provide a secure connection

54.197.185.184 sent an invalid response.

ERR_SSL_PROTOCOL_ERROR

F.7. Resolving the issue.

. Adjusted the default-ssl.conf

```
ubuntu@ip-172-31-8-192:/etc/apache2/sites-available$ egrep -i "servername|sslcertificate" default*.conf|grep -v "#"
ServerName ec2-54-197-185-184.compute-1.amazonaws.com
SSLCertificateFile /etc/apache2/ssl.crt/server.crt
SSLCertificateKeyFile /etc/apache2/ssl.key/server.key
```

```
sudo a2ensite default-ssl.conf
```

. Added Redirect in 000-default.conf

```
<VirtualHost *:80>
# The ServerName directive sets the request scheme, hostname and port that
# the server uses to identify itself. This is used when creating
# redirection URLs. In the context of virtual hosts, the ServerName
# specifies what hostname must appear in the request's Host: header to
# match this virtual host. For the default virtual host (this file) this
# value is not decisive as it is used as a last resort host regardless.
# However, you must set it for any further virtual host explicitly.
#ServerName www.example.com

ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
Redirect permanent / https://ec2-54-197-185-184.compute-1.amazonaws.com/
```

. Reload apache2 sudo service apache2 reload

Create Apache Server Instance via AWS of amazon.com

G. Now we can access this page via https (well, it doesn't like the certificate)

<https://ec2-54-197-185-184.compute-1.amazonaws.com/index.html>

and it will redirect the request with port 80 to https

<http://ec2-54-197-185-184.compute-1.amazonaws.com:80/index.html>

verified the certificate via this link

<https://www.sslshopper.com/ssl-checker.html#hostname=ec2-54-197-185-184.compute-1.amazonaws.com>