17/08/2015

FULL STACK ENGINEERING; DIY DEVOPS - How to set up a CA signed SSL web application, LAMP stack and CentOS 7 AWS EC2 t2.micro instance in < 30 minutes.

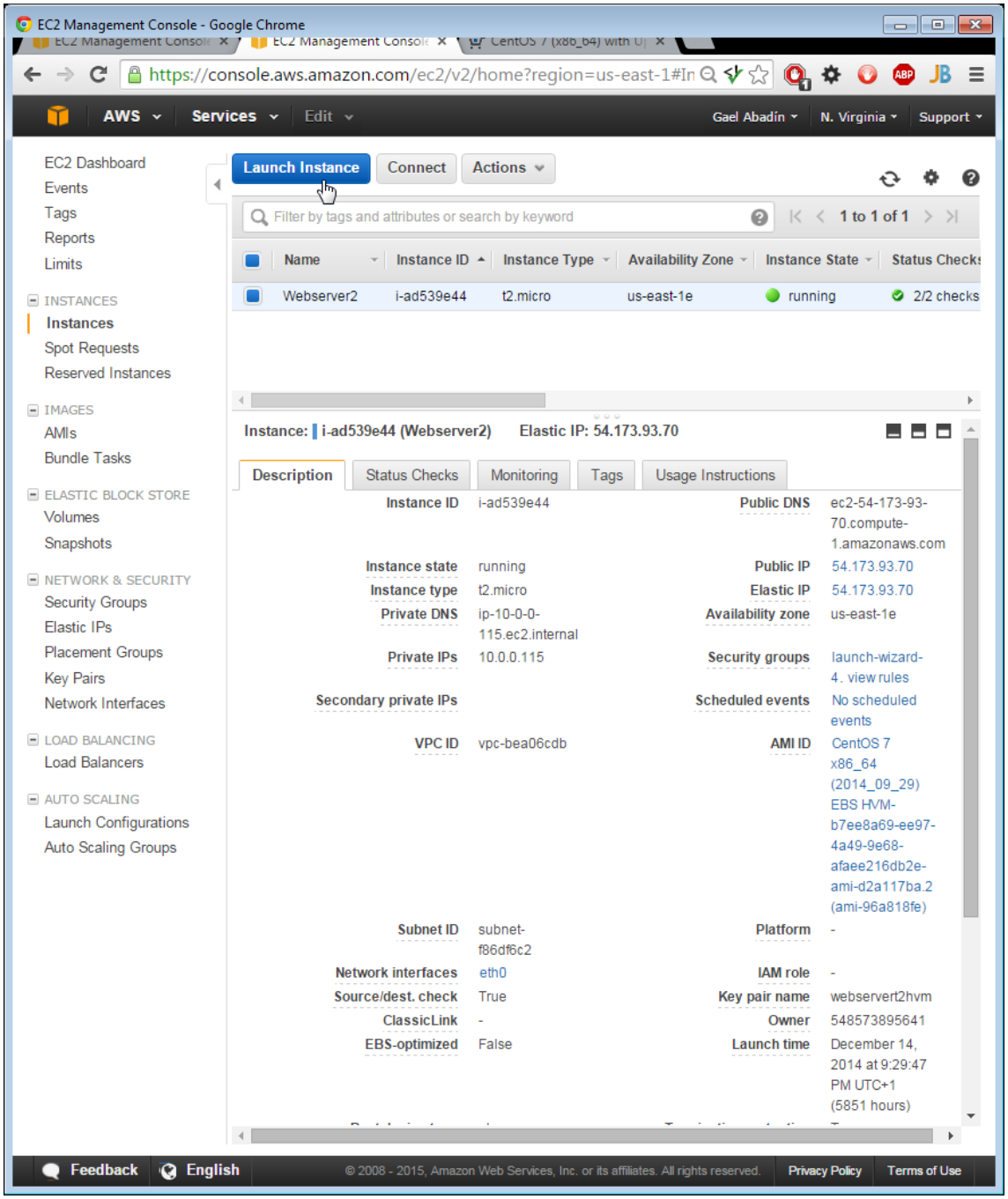
DISCLAIMER: This tutorial is about setting a reasonably safe and secured production environment while throwing out of the window some annoying industry standard good practices (particularly selinux policy enforcement and application based database permissions) which are time consuming and/or require a good deal of skill and effort to be properly put in place and maintained. This is a big NO if you have the resources and time to do otherwise, specially if your web app grows big enough or manages critical enough data to be considered anything else than a random target for a malicious hacker. Unless that is the case or you are contractually bound otherwise, I would recommend starting with a set up like this and build up carefully designed and implemented custom security measures and policies that make sense in the context they are going to be used. In other words: Before thinking on building on a hurry an online fortress with complex auditing and security constrains (which if you are not an expert you will probably misuse, leaving the doors open for anybody to enter), make sure that you know how to set the basic locks on a public server so you or anyone having a key + passphrase are the only ones able to do nasty stuff on it. Also, don't spend countless hours on having the most awesome, secure and efficient production environment when you still have less than 1000 users and your beta service is still full of bugs and lacks very much demanded functionality and UX improvements; that's the shortest path to sink. Instead, try to start with something sound and simple and keep making it better. That being said, I take no responsibility for the damage that any of my advice might cause to your company or reputation.

1/5: DEPLOY AND ACCESS AN AWS EC2 CENTOS 7 INSTANCE

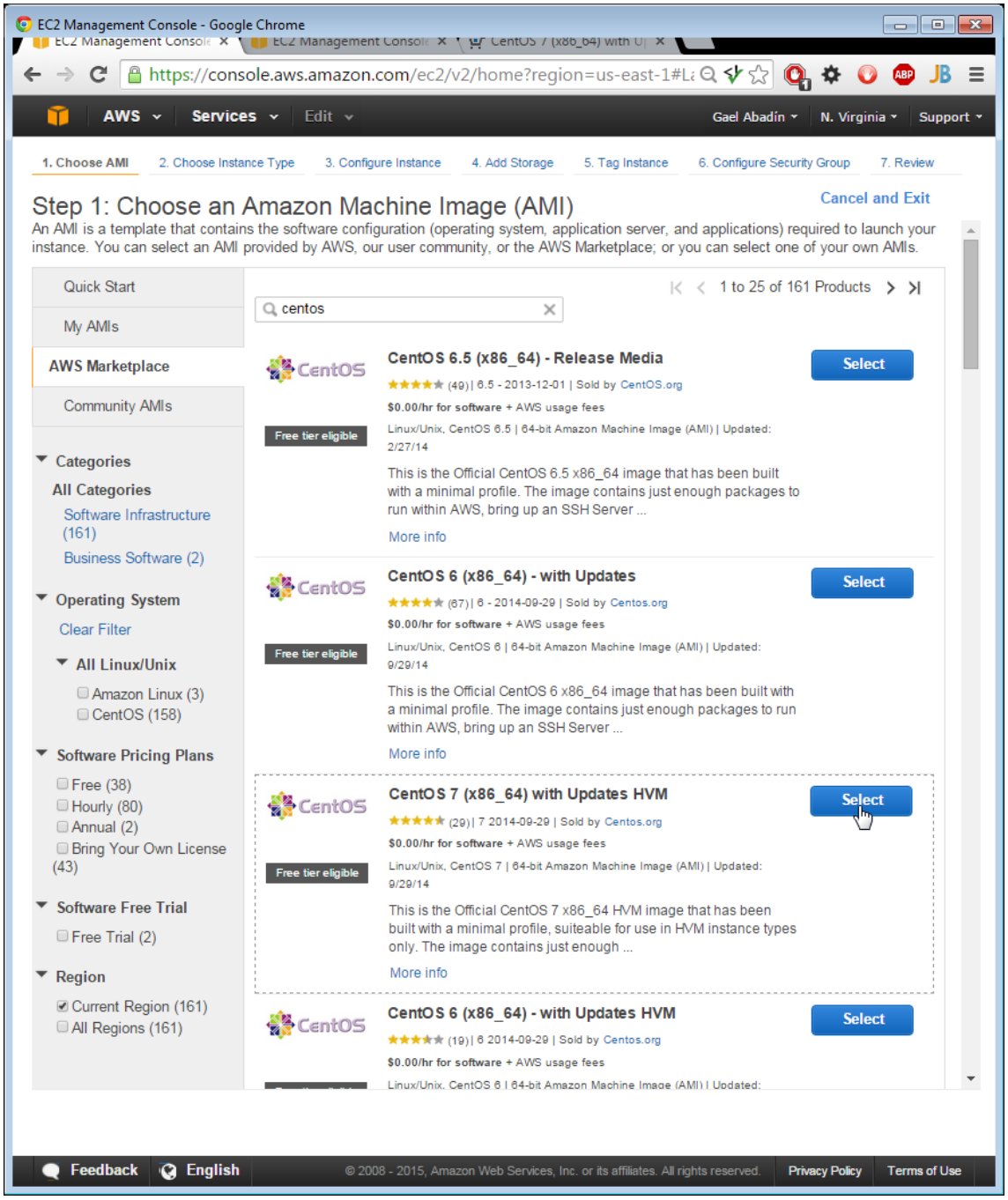
On this part I will show how to quickly deploy an affordable AWS EC2 LAMP instance for kick-starting a production environment. It won't handle boatloads of traffic but with the proper tuning you will be able to serve up more than 50 simultaneous average requests, like a WordPress user requesting a blog page, without your users noticing any performance impact; and without the need of setting up any kind of load balancing or dynamic content caching mechanism.

Since I want to make this quick I will presume you already are somehow familiar with the terminology and the technology involved and have at least a notion of what you are about to do, so let's go:

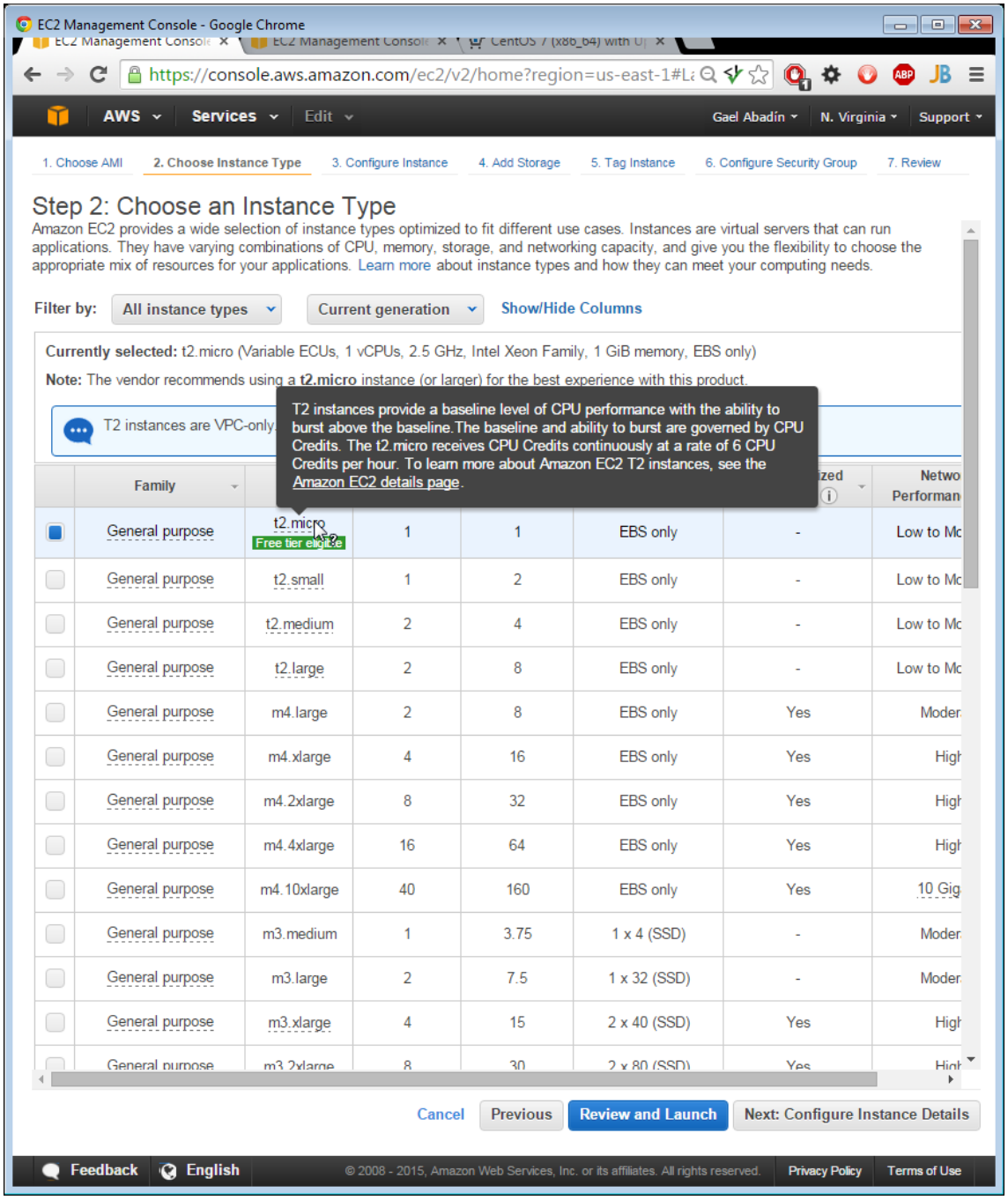
- First you log into your AWS account, go to EC2, confirm you are in your region of choice (in my case north virginia, as you can see in the top right of the screen capture) and then click on the launch instance button that shows up when you select the instances tab



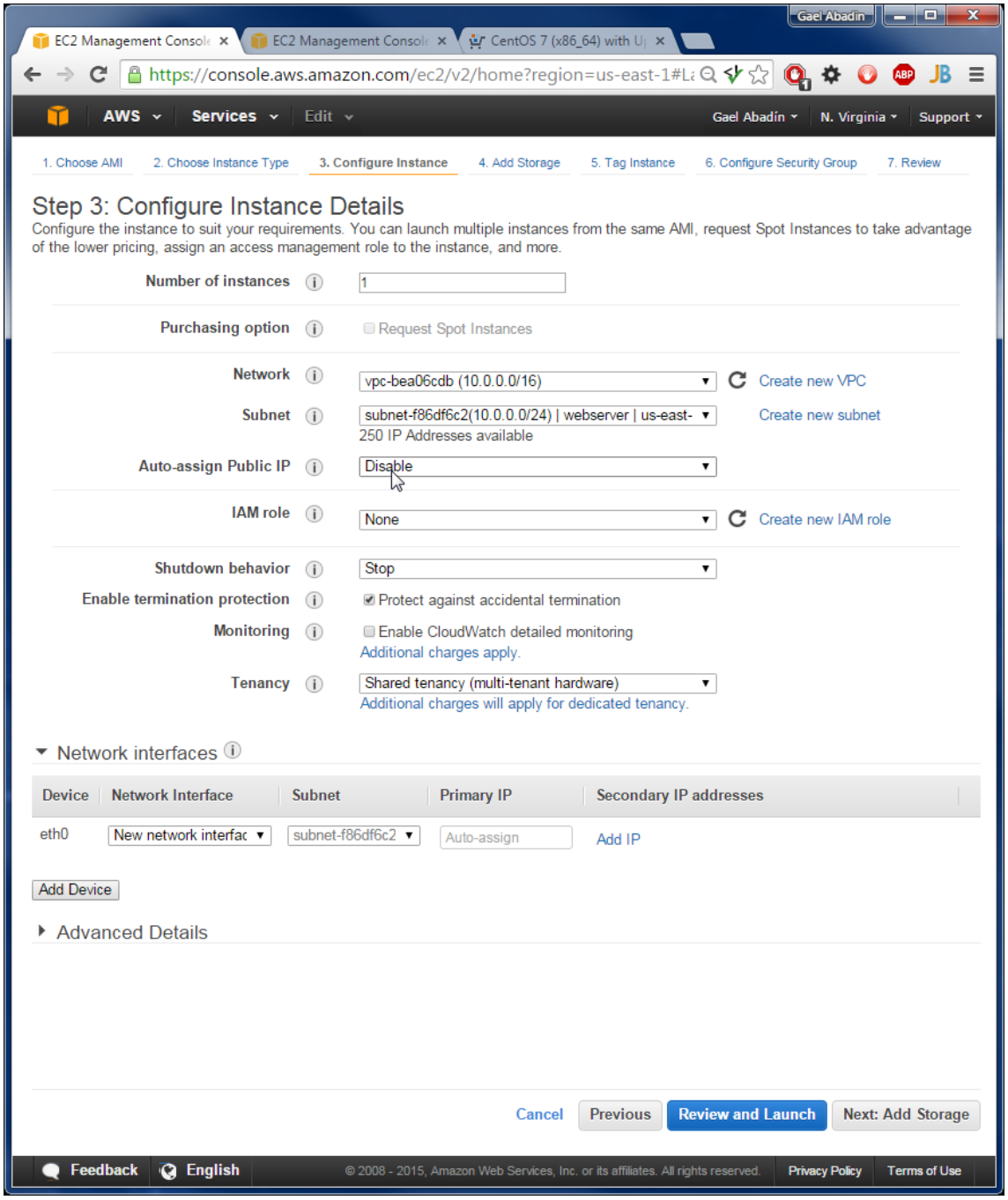
- Now pick your AMI from the AWS marketplace tab: Centos 7 with updates on HVM virtualization



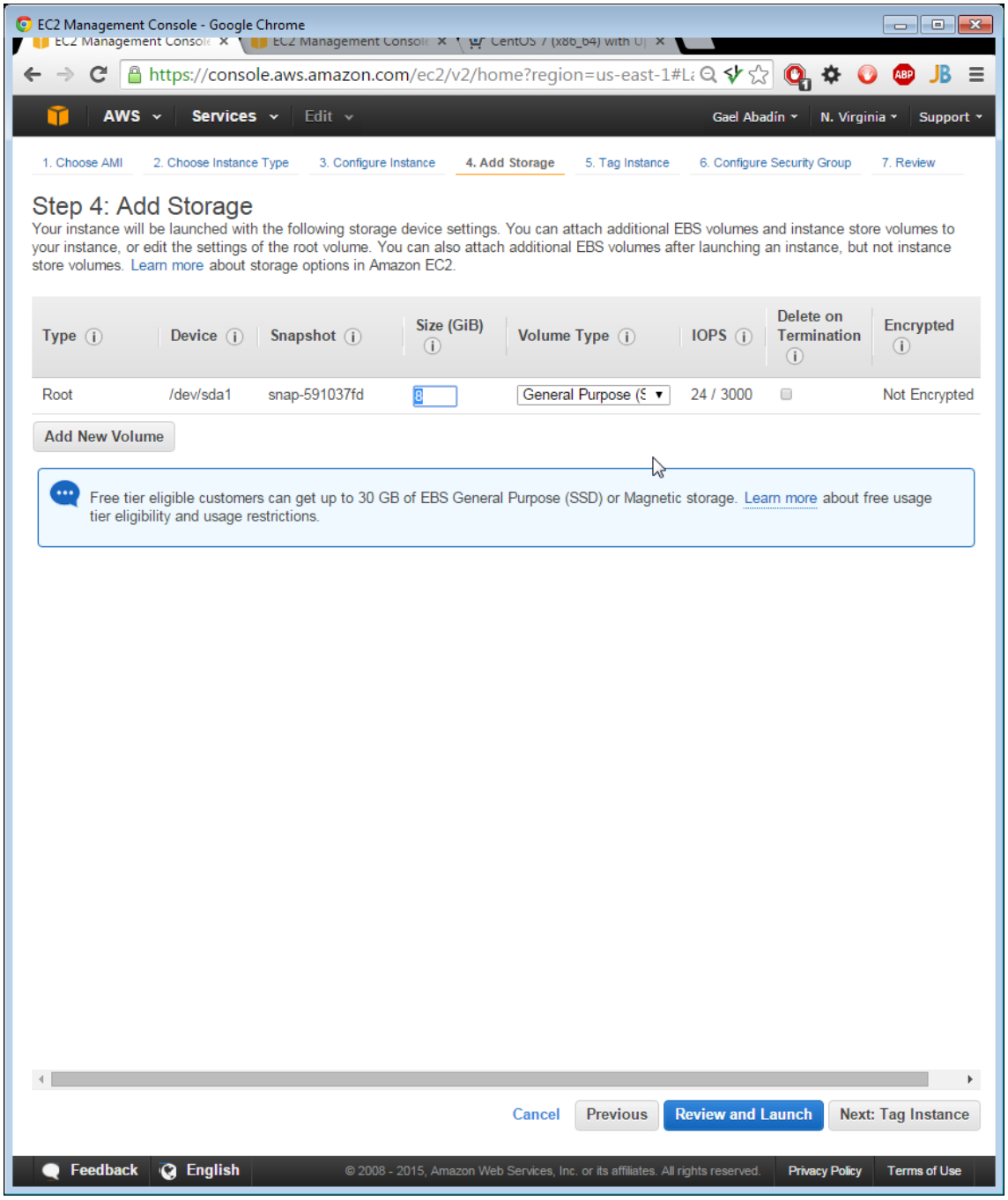
- Pick the instance type: A general purpose t2.micro, with 2.5GHz and 1GB of RAM, is a good place to start



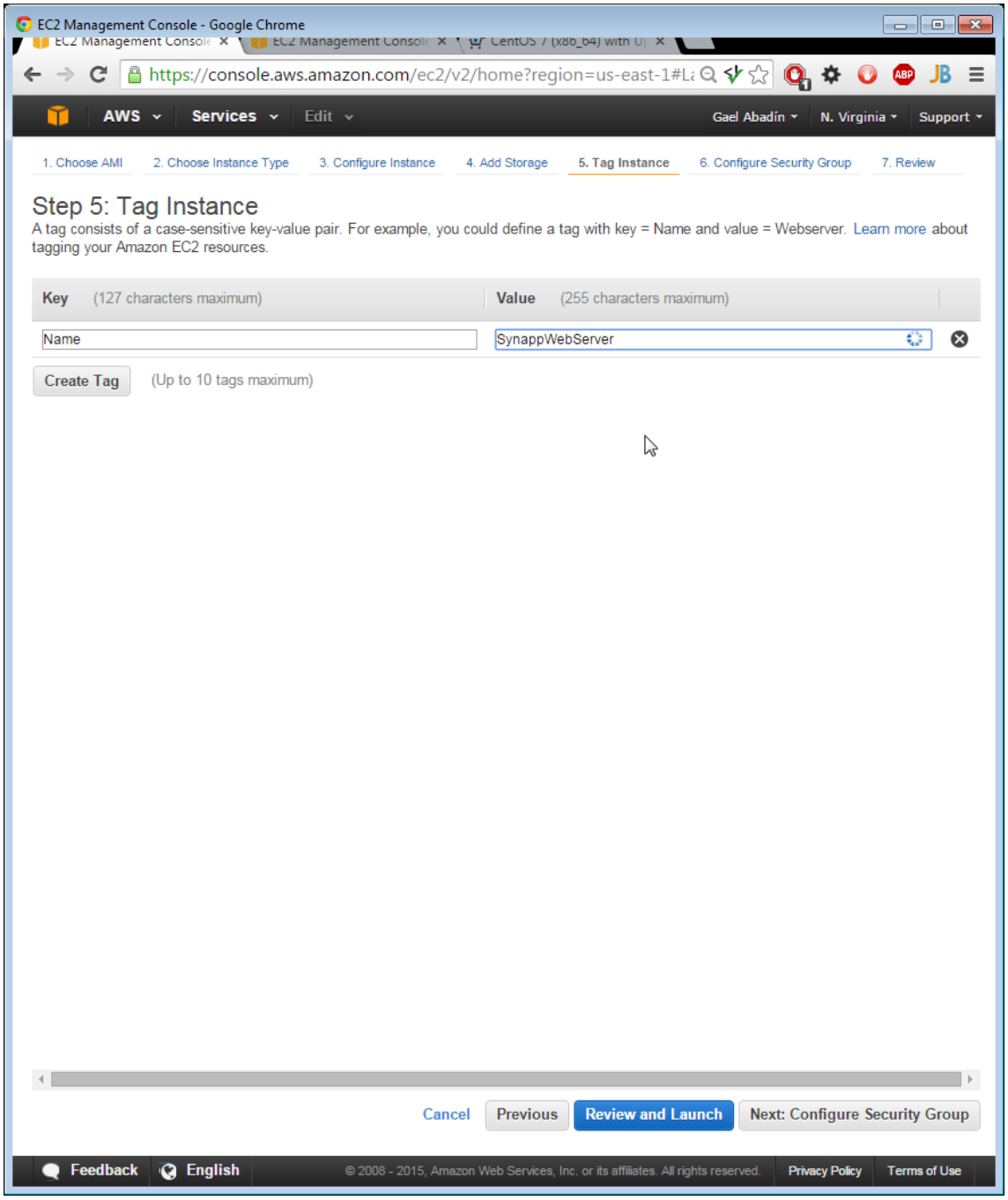
- Fill in the instance details form, so it looks more or less like this. It is a no brainer. You will have to create a VPC and subnet if you don't have one (As a general rule avoid EC2 Classic even when it is an option; if you become a frequent AWS user you'll thank me for that many times). In this case I did not enable a public IP because I wanted to transfer an Elastic IP address from another instance, as I will show you later on.



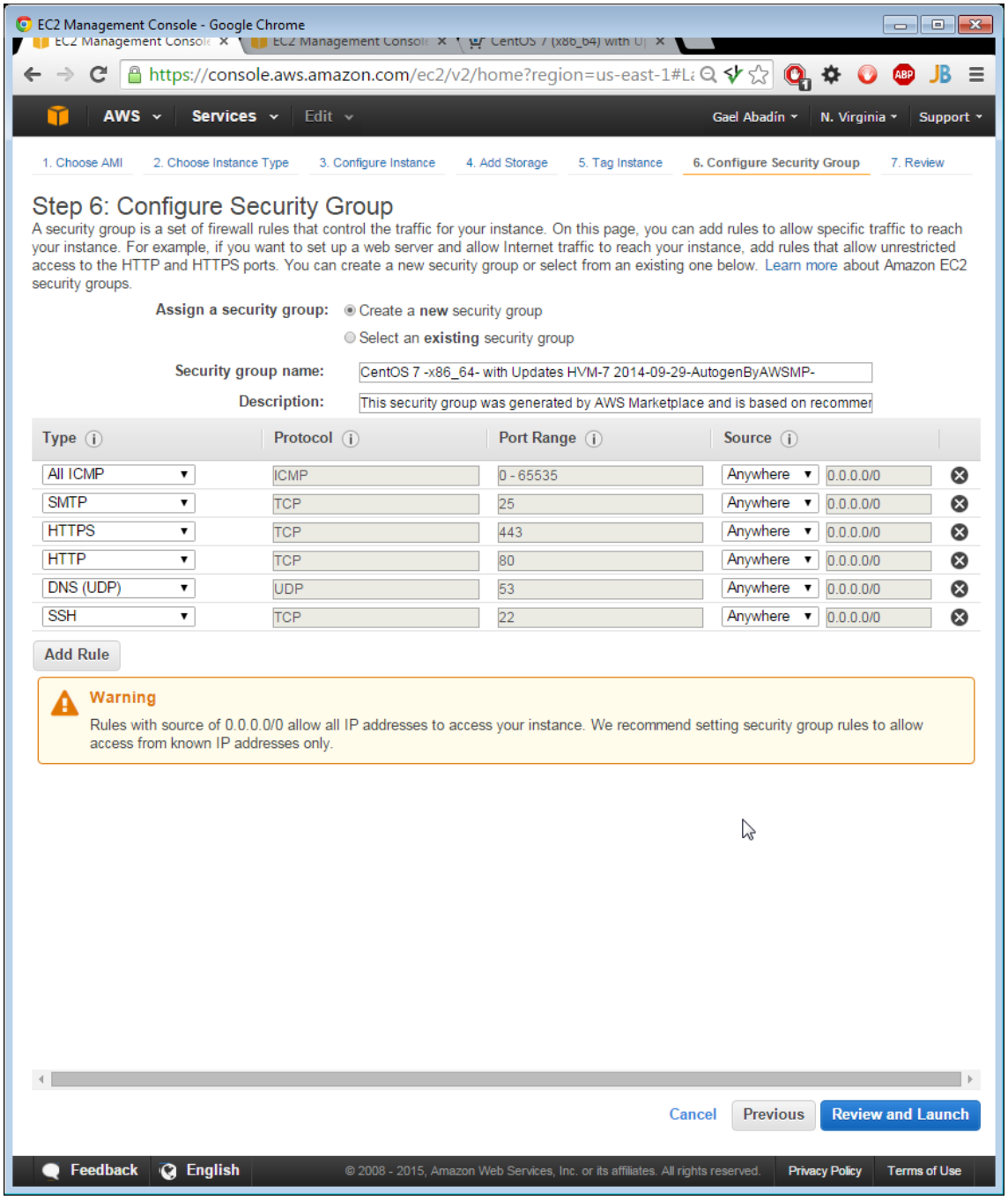
- For the storage, I chose the General purpose SSD option because I don't expect significant load most of the time, it is cheap and it performs better than magnetic storage on occasional peaks. The only app I will deploy on this server will not require more than a few megabytes per user, and that's the worst case scenario. Choosing 8 GB will leave me with 1 GB for swapping plus 4GB of free space for their needs, and it didn't took much of a design effort to make sure I could move all the user space to another drive without downtime or hassle if I ever require it.



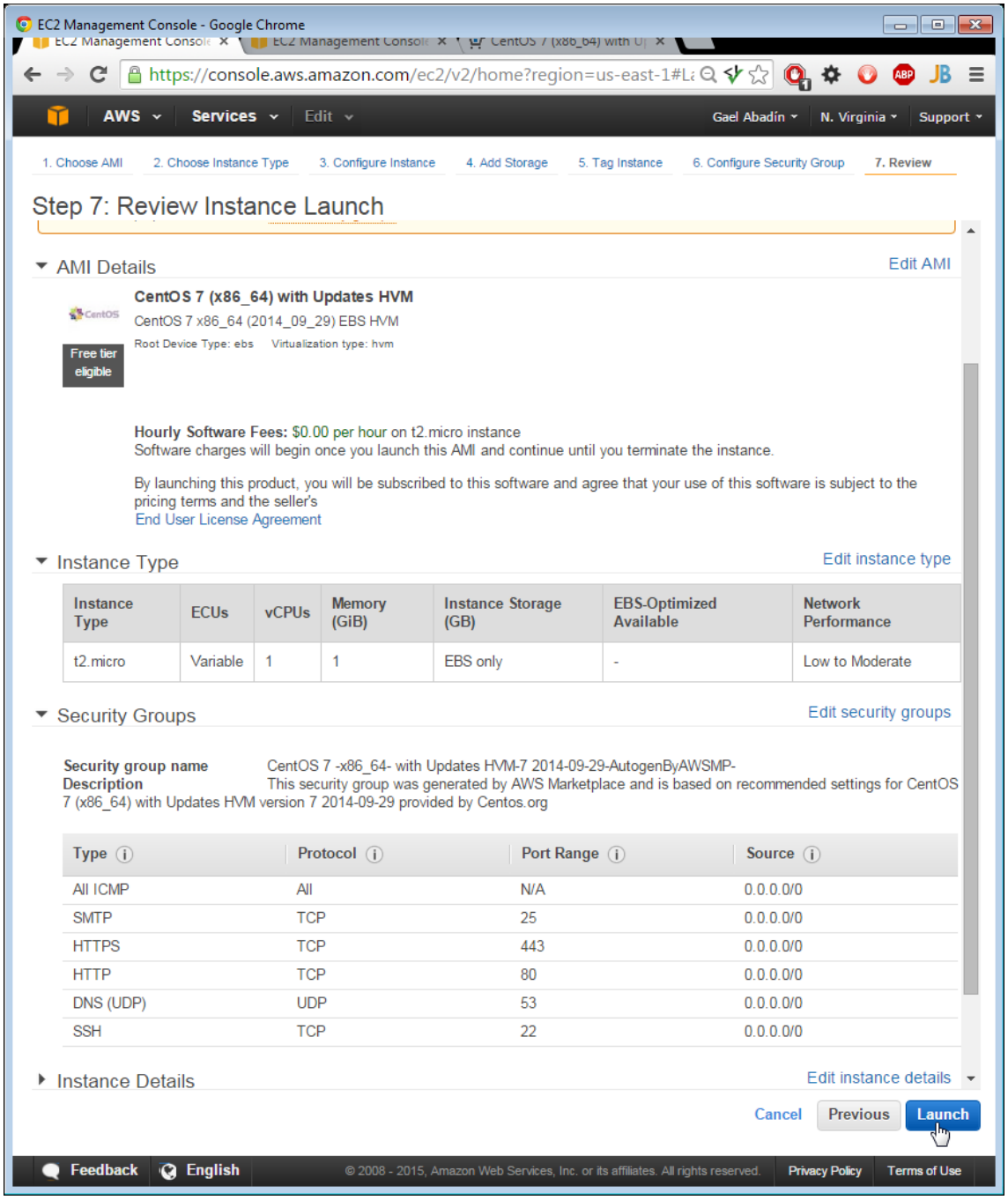
- Moving on to the tagging step. Here you name your instance. This should be an easy one.

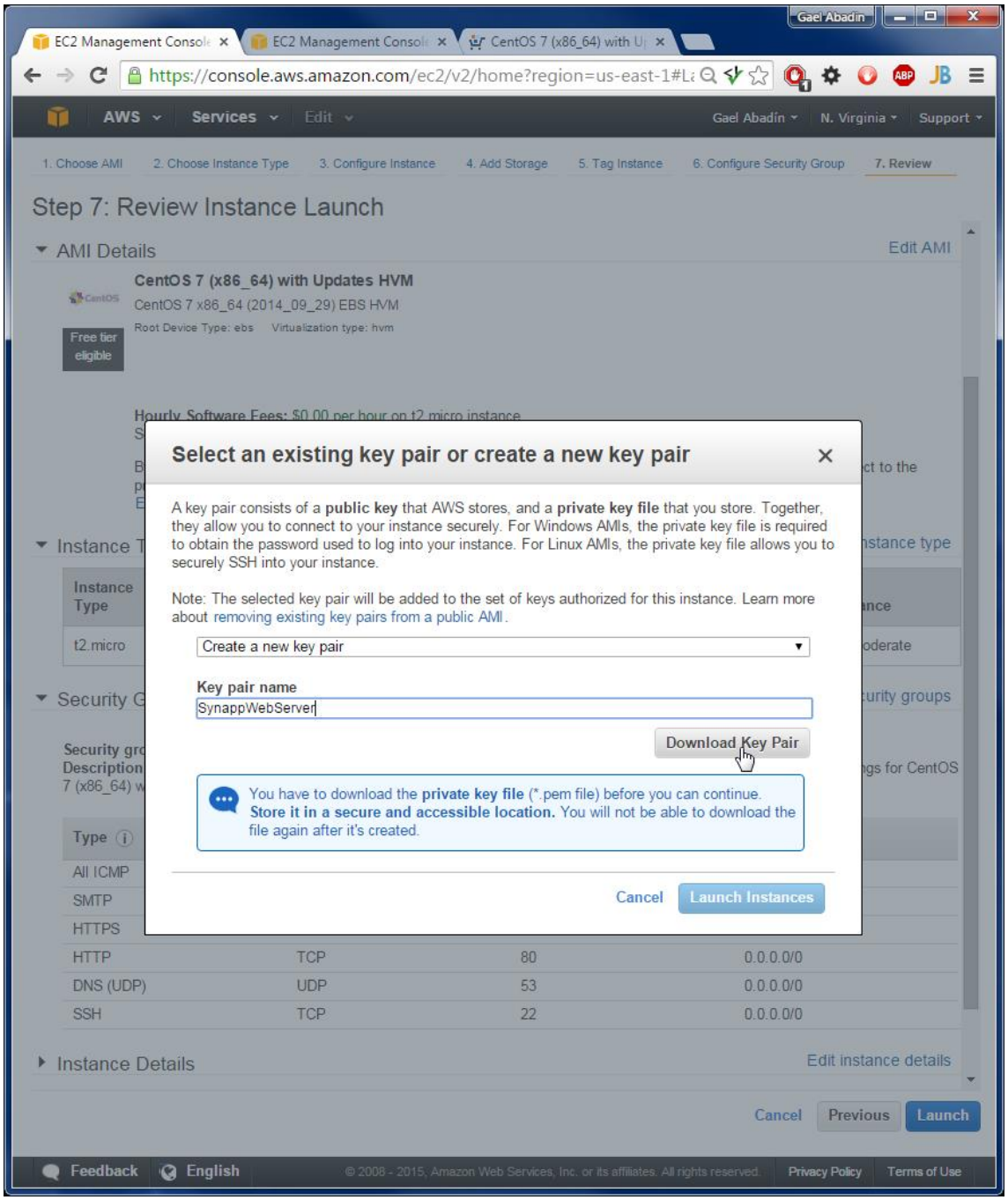


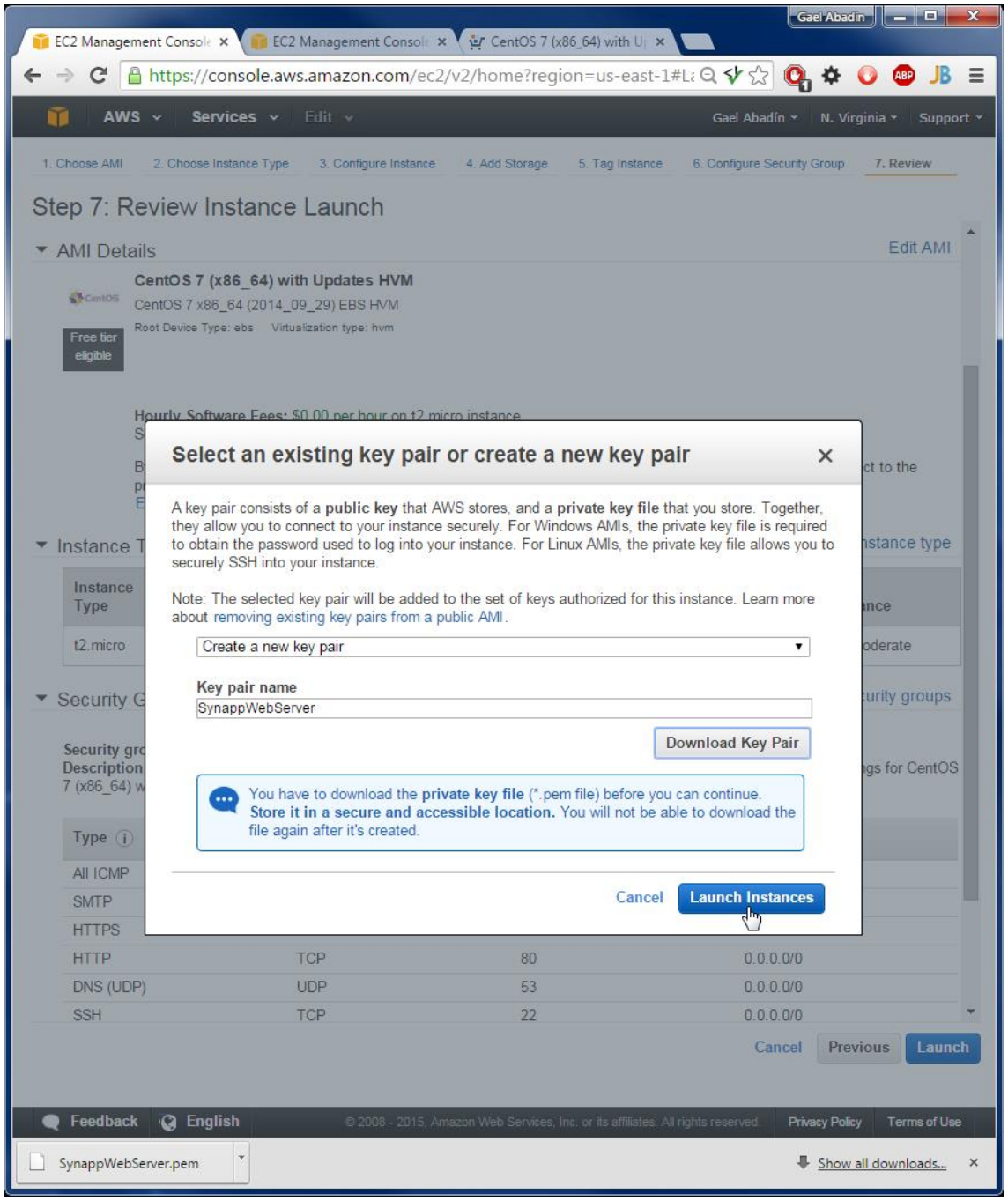
- The next step is a little bit tricky. You can always edit this settings and open up any service you require, so I suggest to leave it to the absolute minimum. Something I usually do is restrict the source of all incoming connections to my IP until I finish deploying the app and I am ready to go live, which I recommend you do instead. I have adapted this workflow a little bit and set the machine live right away to cover the whole EC2 management process in one part instead of adding an extra bit at the end of the last one. If you are doing the whole set up in one session it won't make much of a difference, anyway.



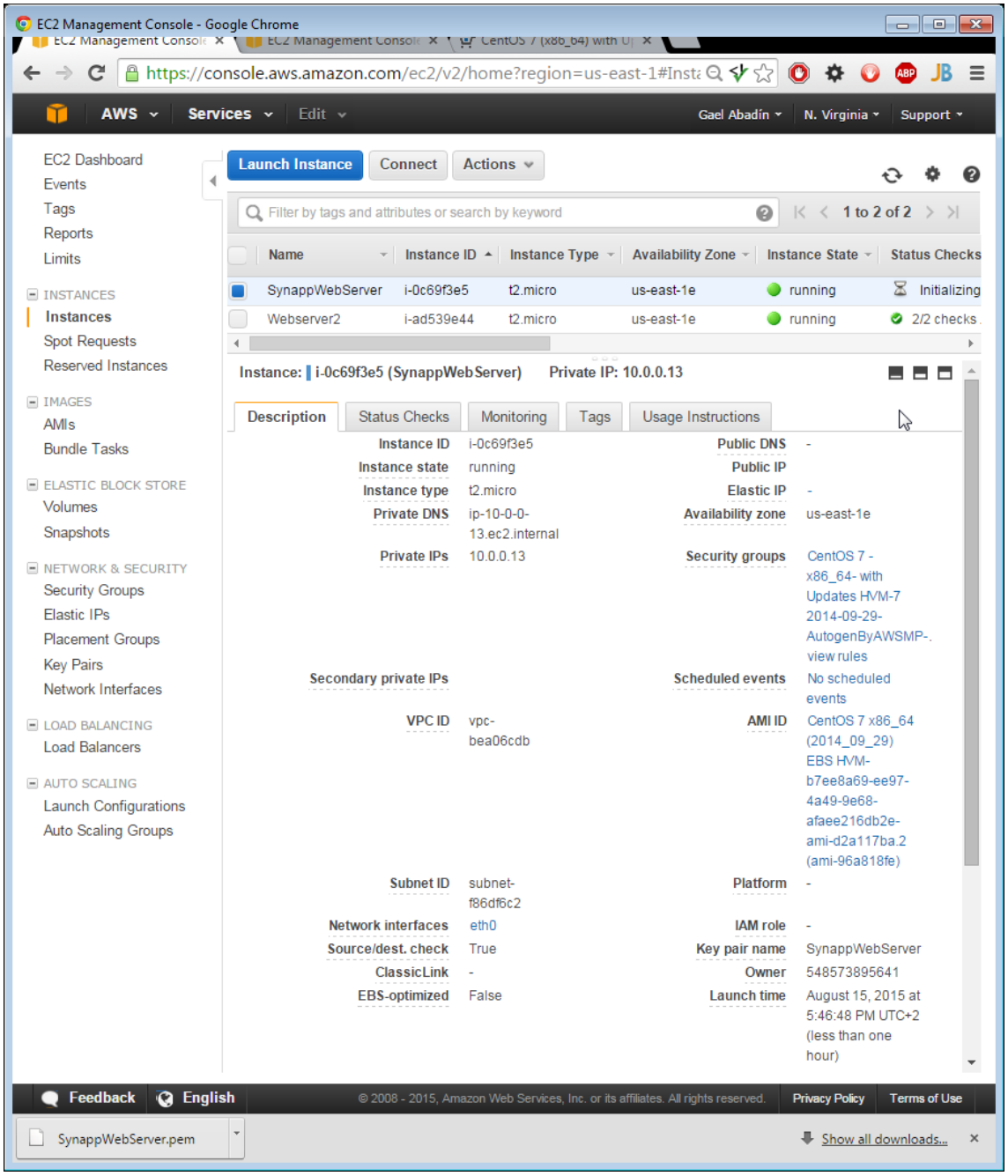
- Last step before launch is to review the configuration, and the most important! You must generate and download a public/private key pair to connect to the instance you are about to launch. Later on I will show you how to set up this connection.


EC2 Management Console - Google Chrome
https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#L
AWS
Services
Edit
Gael Abadín
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Tag Instance
6. Configure Security Group
7. Review
Step 7: Review Instance Launch
AMI Details
Edit AMI
CentOS 7 (x86_64) with Updates HVM
CentOS 7 x86_64 (2014_09_29) EBS HVM
Root Device Type: ebs Virtualization type: hvm
Free tier eligible
Hourly Software Fees: $0.00 per hour on t2.micro instance
Software charges will begin once you launch this AMI and continue until you terminate the instance.
By launching this product, you will be subscribed to this software and agree that your use of this software is subject to the pricing terms and the seller's
End User License Agreement
Instance Type
Edit instance type
Instance Type | ECUs | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance
t2.micro | Variable | 1 | 1 | EBS only | - | Low to Moderate
Security Groups
Edit security groups
Security group name CentOS 7 -x86_64- with Updates HVM-7 2014-09-29-AutogenByAWSMP-
Description This security group was generated by AWS Marketplace and is based on recommended settings for CentOS 7 (x86_64) with Updates HVM version 7 2014-09-29 provided by Centos.org
Type | Protocol | Port Range | Source
All ICMP | All | N/A | 0.0.0.0/0
SMTP | TCP | 25 | 0.0.0.0/0
HTTPS | TCP | 443 | 0.0.0.0/0
HTTP | TCP | 80 | 0.0.0.0/0
DNS (UDP) | UDP | 53 | 0.0.0.0/0
SSH | TCP | 22 | 0.0.0.0/0
Instance Details
Edit instance details
Cancel
Previous
Launch
Feedback
English
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

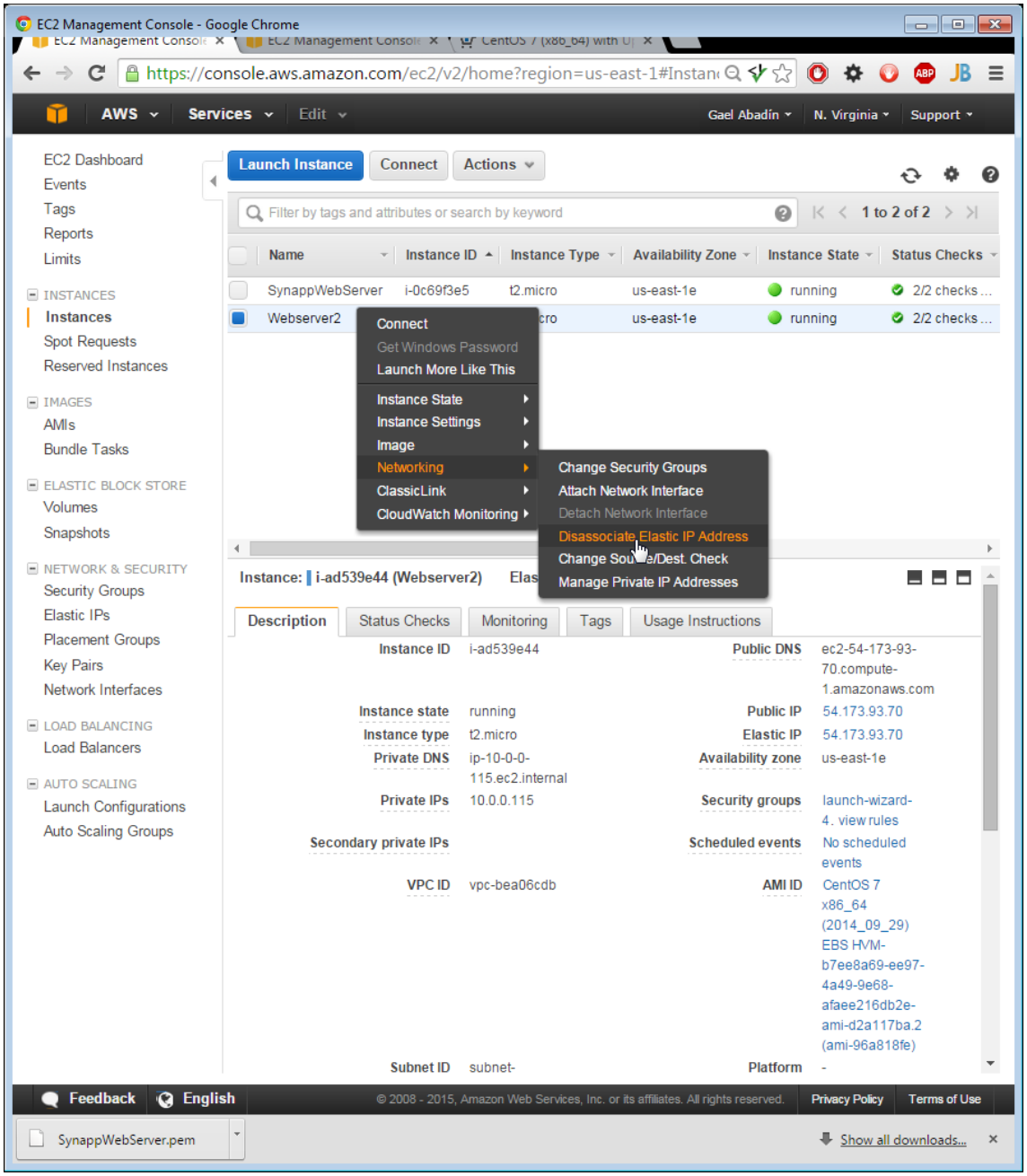

Gael Abadín
EC2 Management Console
EC2 Management Console
CentOS 7 (x86_64) with Updates
https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Launch
AWS
Services
Edit
Gael Abadín
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Tag Instance
6. Configure Security Group
7. Review
Step 7: Review Instance Launch
AMI Details
Edit AMI
CentOS 7 (x86_64) with Updates HVM
CentOS 7 x86_64 (2014_09_29) EBS HVM
Root Device Type: ebs Virtualization type: hvm
Free tier eligible
Hourly Software Fees: $0.00 per hour on t2.micro instance
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
SynappWebServer
Download Key Pair
You have to download the private key file (*.pem file) before you can continue. Store it in a secure and accessible location. You will not be able to download the file again after it's created.
Cancel
Launch Instances
Instance Type
t2.micro
Security Groups
Type
All ICMP
SMTP
HTTPS
HTTP TCP 80 0.0.0.0/0
DNS (UDP) UDP 53 0.0.0.0/0
SSH TCP 22 0.0.0.0/0
Instance Details
Edit instance details
Cancel
Previous
Launch
Feedback
English
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

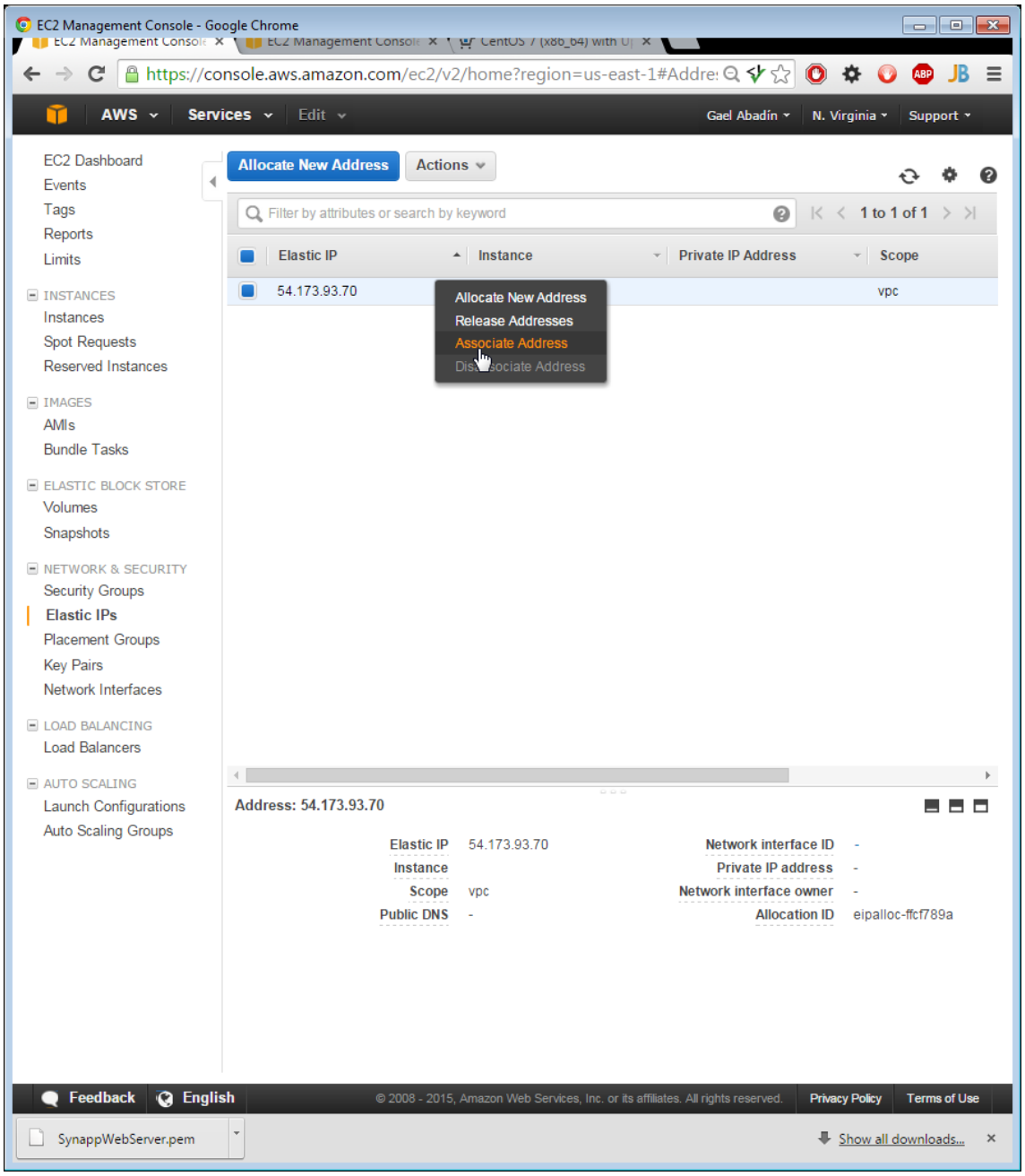


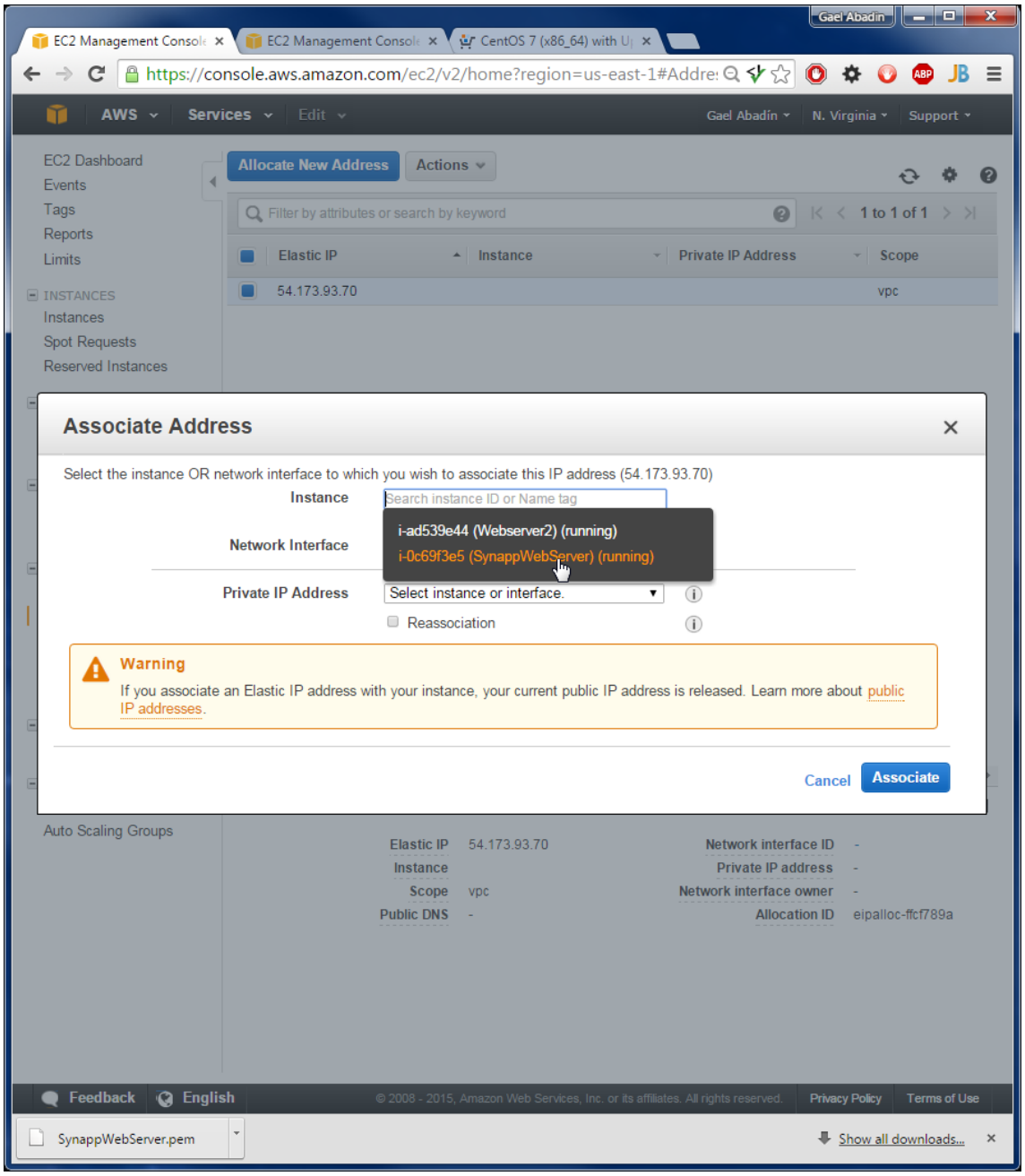
- This is what you will see when your instance is running. Wait for the checks to show and that will mean it is ready to accept connections. If you have not assigned a public IP address you will have to connect internally through another machine on the subnet or, like I am about to do, assign a public IP address to the machine so you can establish a direct connection from your PC.



- Creating and assigning an Elastic public IP address to an instance is very easy. You can see I am releasing it from a running instance but you can also do a hot swap reassignment. Just select the Elastic IPs tab on the menu and fill in the form on the screen, like this.

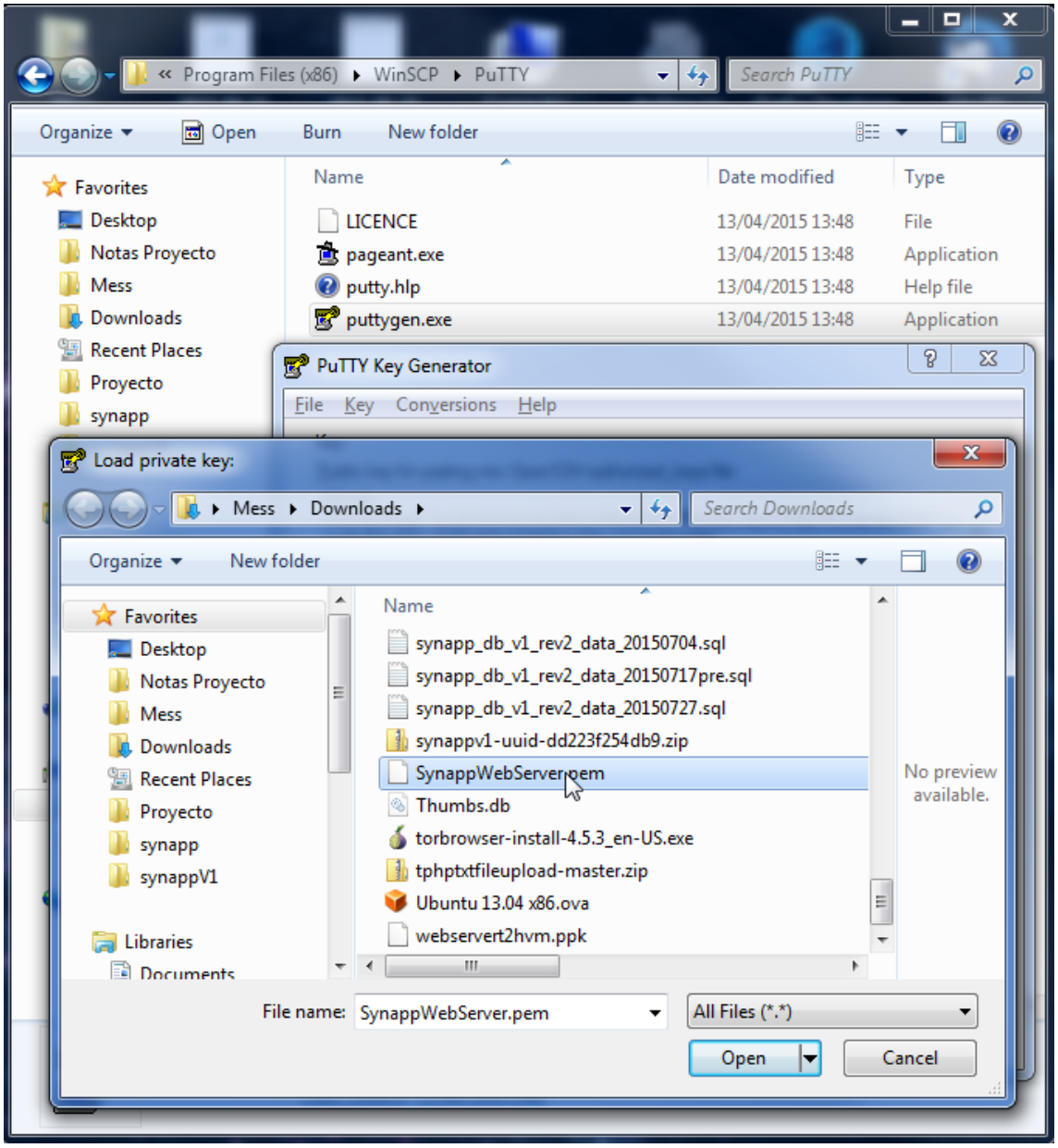

EC2 Management Console - Google Chrome
https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instan
AWS
Services
Edit
Gael Abadín
N. Virginia
Support
EC2 Dashboard
Events
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
AUTO SCALING
Launch Configurations
Auto Scaling Groups
Launch Instance
Connect
Actions
Filter by tags and attributes or search by keyword
1 to 2 of 2
Name
Instance ID
Instance Type
Availability Zone
Instance State
Status Checks
SynappWebServer
i-0c69f3e5
t2.micro
us-east-1e
running
2/2 checks ...
Webserver2
us-east-1e
running
2/2 checks ...
Connect
Get Windows Password
Launch More Like This
Instance State
Instance Settings
Image
Networking
ClassicLink
CloudWatch Monitoring
Change Security Groups
Attach Network Interface
Detach Network Interface
Disassociate Elastic IP Address
Change Source/Dest. Check
Manage Private IP Addresses
Instance: i-ad539e44 (Webserver2)
Description
Status Checks
Monitoring
Tags
Usage Instructions
Instance ID
i-ad539e44
Instance state
running
Instance type
t2.micro
Private DNS
ip-10-0-0-115.ec2.internal
Private IPs
10.0.0.115
Secondary private IPs
VPC ID
vpc-bea06cdb
Subnet ID
subnet-
Public DNS
ec2-54-173-93-70.compute-1.amazonaws.com
Public IP
54.173.93.70
Elastic IP
54.173.93.70
Availability zone
us-east-1e
Security groups
launch-wizard-4. view rules
Scheduled events
No scheduled events
AMI ID
CentOS 7 x86_64 (2014_09_29) EBS HVM-b7ee8a69-ee97-4a49-9e68-afaee216db2e-ami-d2a117ba.2 (ami-96a818fe)
Platform
-
Feedback
English
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
SynappWebServer.pem
Show all downloads...

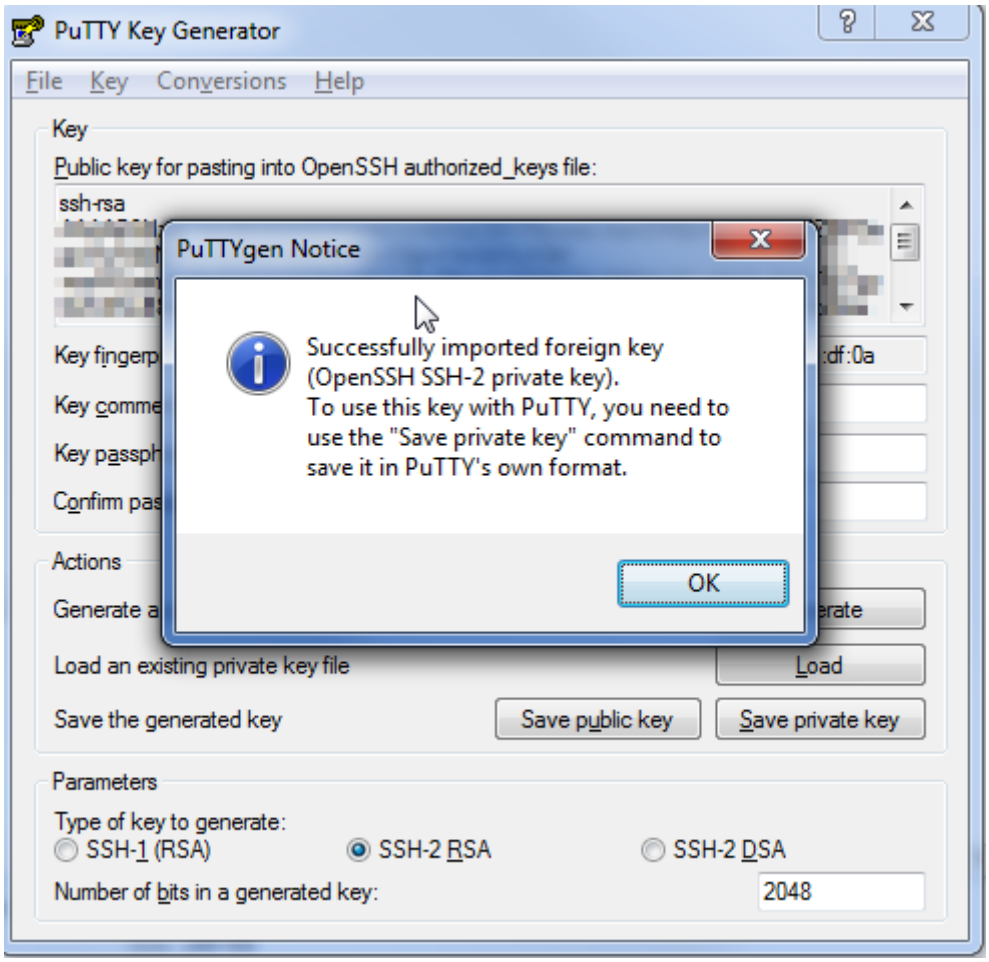

EC2 Management Console - Google Chrome
https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Addre
AWS
Services
Edit
Gael Abadín
N. Virginia
Support
EC2 Dashboard
Events
Tags
Reports
Limits
INSTANCES
Instances
Spot Requests
Reserved Instances
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
NETWORK & SECURITY
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
LOAD BALANCING
Load Balancers
AUTO SCALING
Launch Configurations
Auto Scaling Groups
Allocate New Address
Actions
Filter by attributes or search by keyword
1 to 1 of 1
Elastic IP
Instance
Private IP Address
Scope
54.173.93.70
vpc
Allocate New Address
Release Addresses
Associate Address
Disassociate Address
Address: 54.173.93.70
Elastic IP 54.173.93.70
Instance
Scope vpc
Public DNS -
Network interface ID -
Private IP address -
Network interface owner -
Allocation ID eipalloc-ffcf789a
Feedback
English
© 2008 - 2015, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
SynappWebServer.pem
Show all downloads...

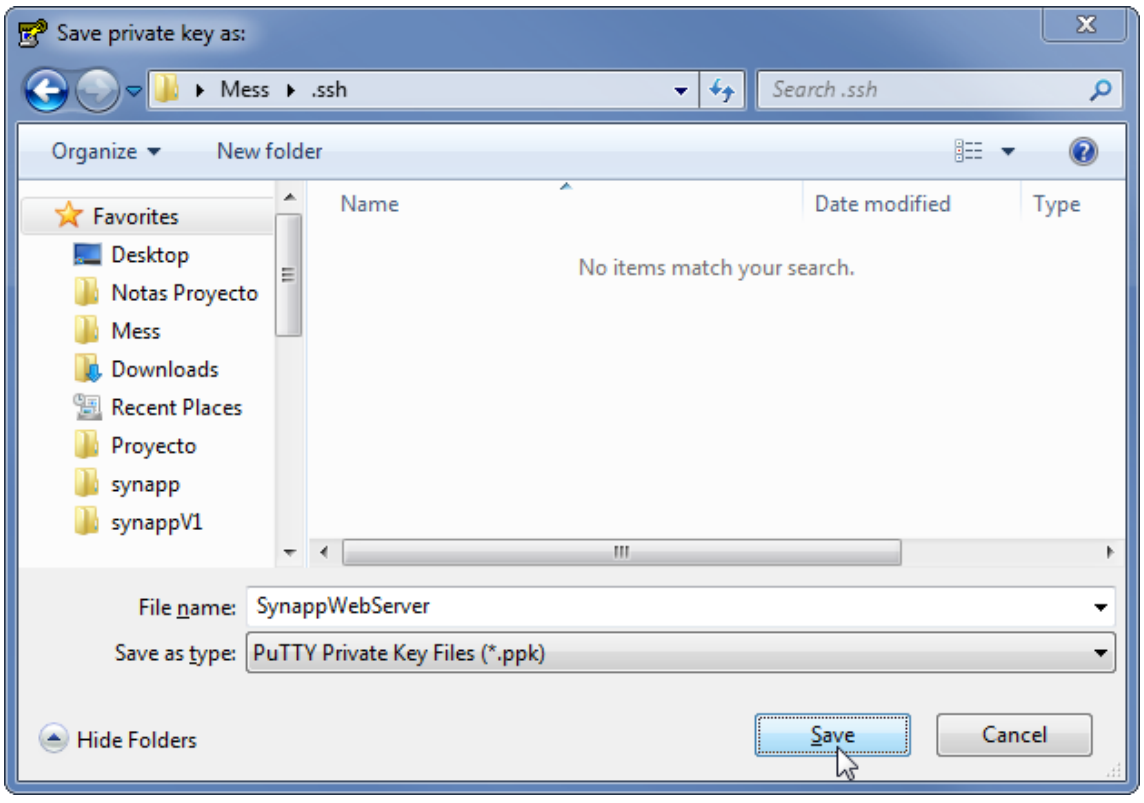


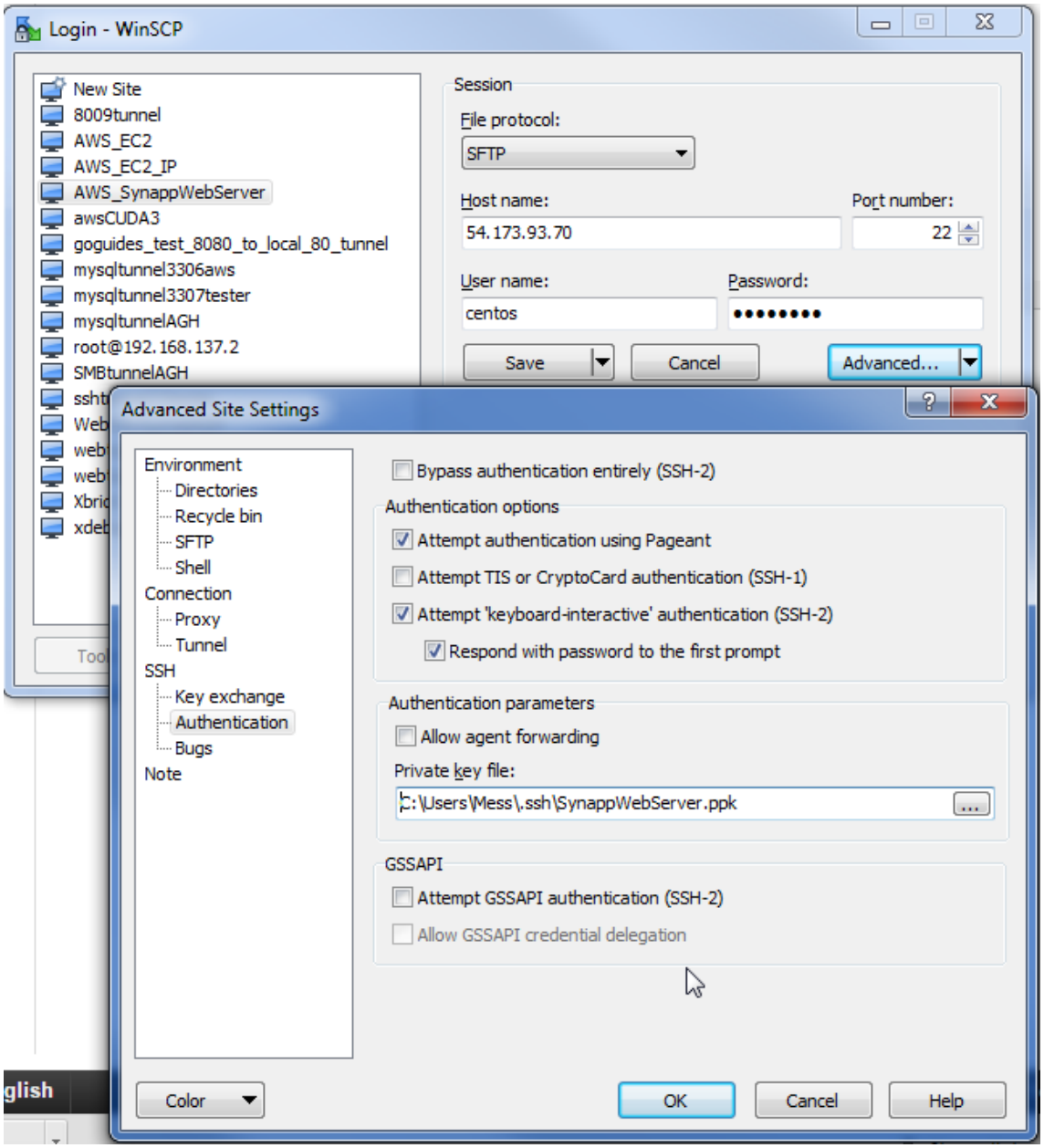
That's all. You set your instance running in no time and we have arrived to the second part of this part where you can see how to connect to it using WinSCP and putty SSH client. With all due respect to Windows users, the reason I am not showing how to do this to Linux desktop users is that you will probably think that I think that you were born yesterday.

- Moving on: the first step is to download and install WinSCP (http://www.winscp.com) and then run the putty key generator that comes with it. This tool will allow you to open the key pair you downloaded before launching the instance (If you didn't you are out of luck because Amazon doesn't store them), and save the private key part so you can assign it to your connection.


« Program Files (x86) ▸ WinSCP ▸ PuTTY
Search PuTTY
Organize
Open
Burn
New folder
Favorites
Desktop
Notas Proyecto
Mess
Downloads
Recent Places
Proyecto
synapp
Name
Date modified
Type
LICENCE
13/04/2015 13:48
File
pageant.exe
13/04/2015 13:48
Application
putty.hlp
13/04/2015 13:48
Help file
puttygen.exe
13/04/2015 13:48
Application
PuTTY Key Generator
File
Key
Conversions
Help
Load private key:
Mess ▸ Downloads ▸
Search Downloads
Organize
New folder
Favorites
Desktop
Notas Proyecto
Mess
Downloads
Recent Places
Proyecto
synapp
synappV1
Libraries
Documents
Name
synapp_db_v1_rev2_data_20150704.sql
synapp_db_v1_rev2_data_20150717pre.sql
synapp_db_v1_rev2_data_20150727.sql
synappv1-uuid-dd223f254db9.zip
SynappWebServer.pem
Thumbs.db
torbrowser-install-4.5.3_en-US.exe
tphptxtfileupload-master.zip
Ubuntu 13.04 x86.ova
webservert2hvm.ppk
No preview available.
File name:
SynappWebServer.pem
All Files (*.*)
Open
Cancel

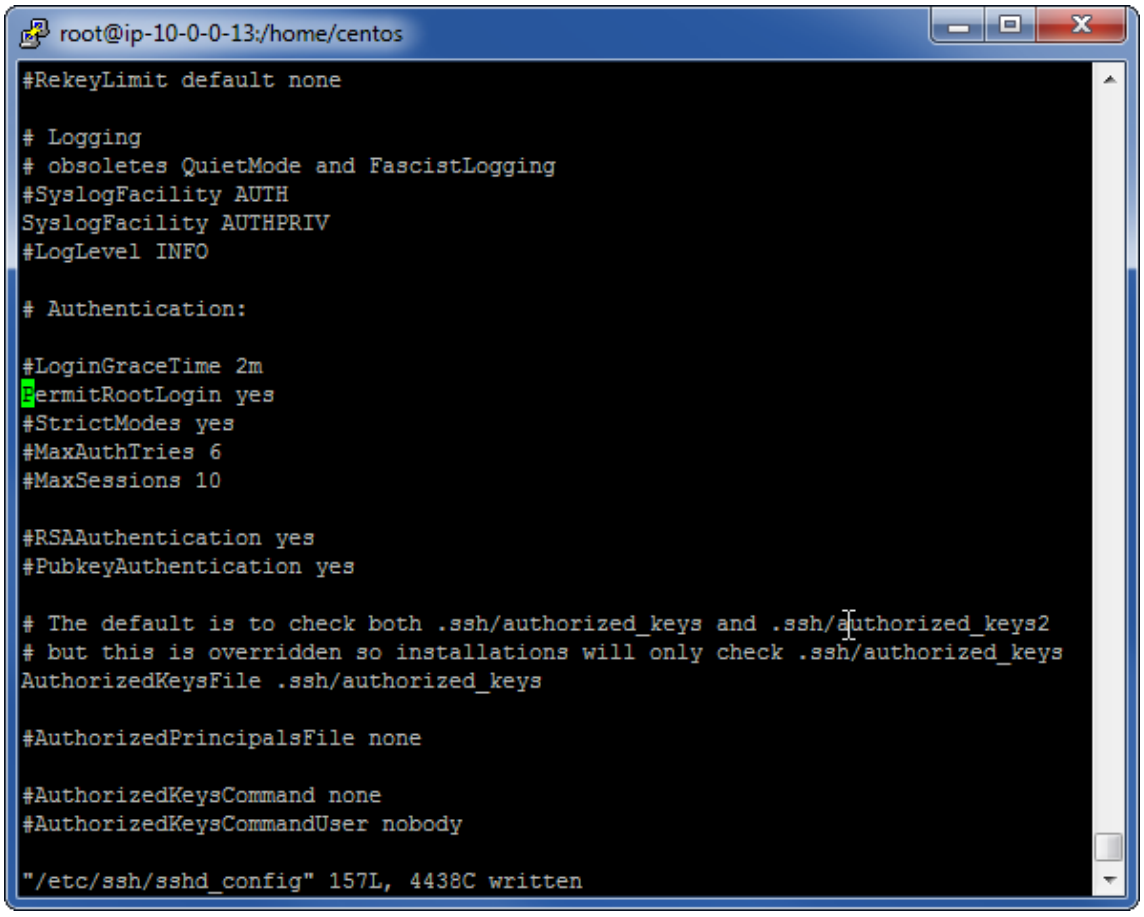

PuTTY Key Generator
File Key Conversions Help
Key
Public key for pasting into OpenSSH authorized_keys file:
ssh-rsa
PuTTYgen Notice
Successfully imported foreign key
(OpenSSH SSH-2 private key).
To use this key with PuTTY, you need to
use the "Save private key" command to
save it in PuTTY's own format.
OK
Actions
Load an existing private key file
Load
Save the generated key
Save public key
Save private key
Parameters
Type of key to generate:
SSH-1 (RSA)
SSH-2 RSA
SSH-2 DSA
Number of bits in a generated key:
2048


Save private key as:
Mess .ssh
Search .ssh
Organize
New folder
Favorites
Desktop
Notas Proyecto
Mess
Downloads
Recent Places
Proyecto
synapp
synappV1
Name
Date modified
Type
No items match your search.
File name: SynappWebServer
Save as type: PuTTY Private Key Files (*.ppk)
Hide Folders
Save
Cancel

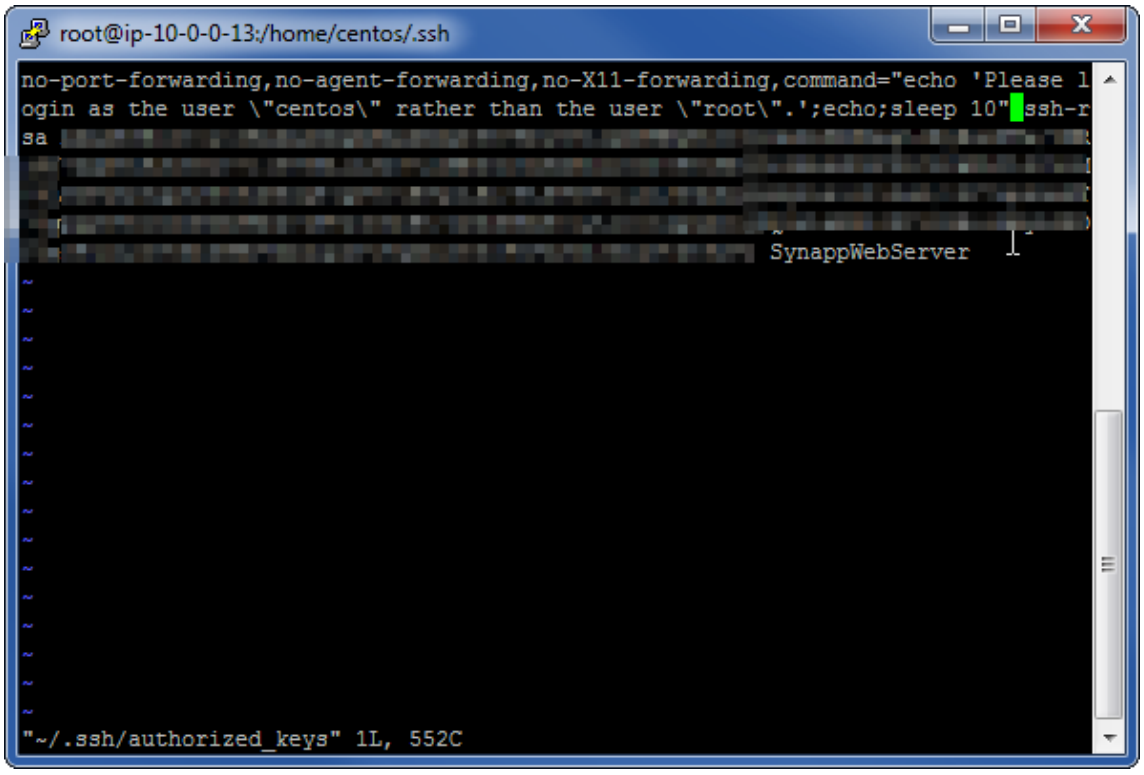


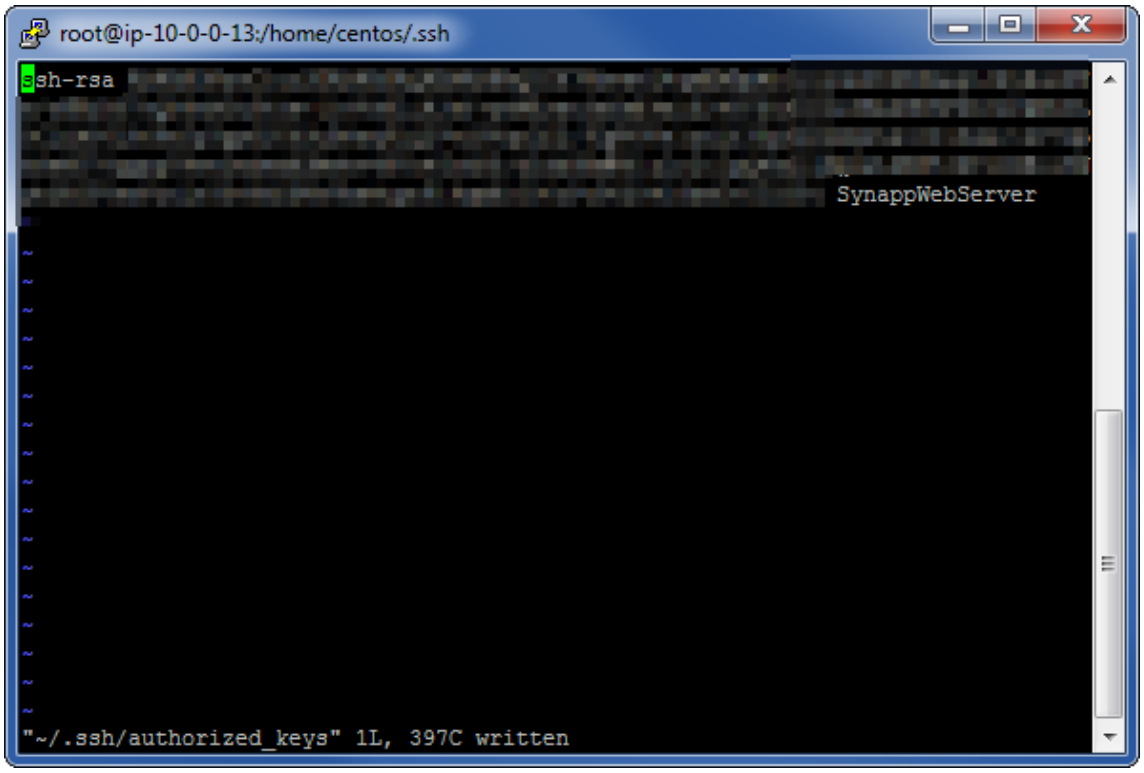
- Once you have done that you are ready to set up a WinSCP or putty connection to your server to transfer files from it and to it or open a terminal to issue commands to the server from it. The only problem is that you have to use the "unprivileged" user centos. A simple sudo su will give you root access on the console but that's still a problem when you want to transfer files to restricted folders, so here is how to do to activate root login.

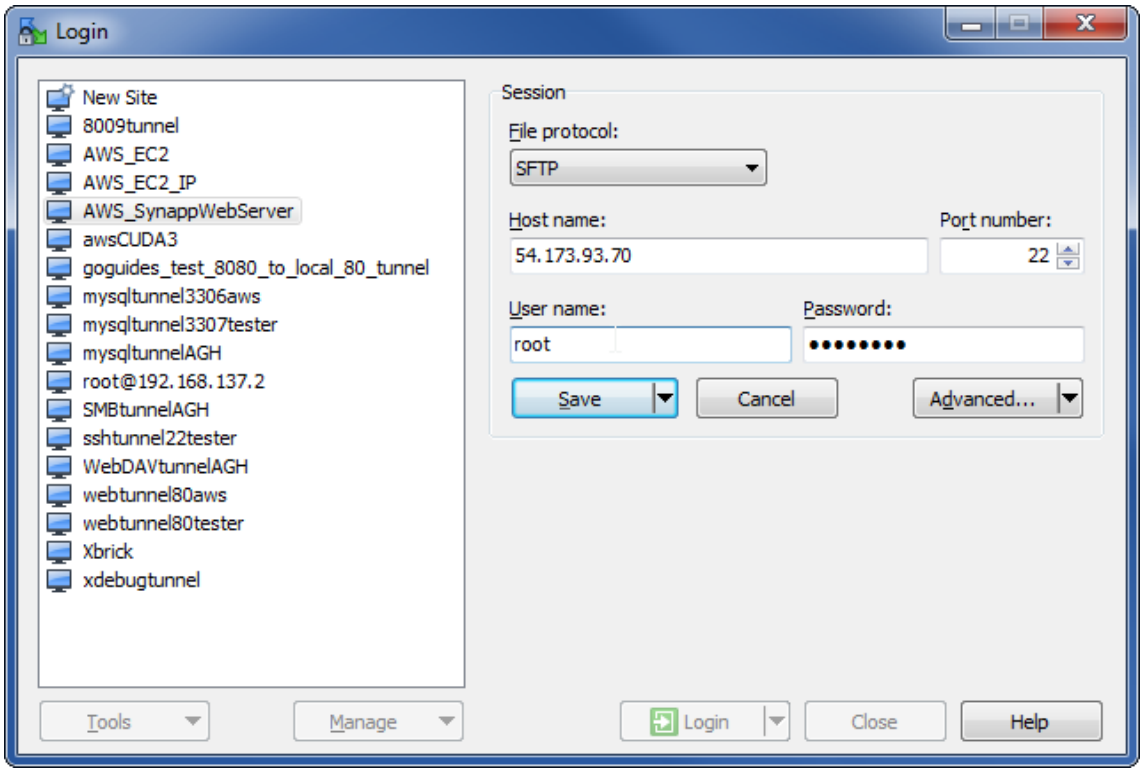
- First, edit /etc/ssh/sshd_config to uncomment and set PermitRootLogin option to yes



- Then, edit /root/.ssh/autorized_keys and delete all the rubble before the key, particularly the echo command, so it ends up like this




root@ip-10-0-0-13:/home/centos/.ssh
ssh-rsa
SynappWebServer
"~/.ssh/authorized_keys" 1L, 397C written


Login
New Site
8009tunnel
AWS_EC2
AWS_EC2_IP
AWS_SynappWebServer
awsCUDA3
goguides_test_8080_to_local_80_tunnel
mysqltunnel3306aws
mysqltunnel3307tester
mysqltunnelAGH
root@192.168.137.2
SMBtunnelAGH
sshtunnel22tester
WebDAVtunnelAGH
webtunnel80aws
webtunnel80tester
Xbrick
xdebugtunnel
Session
File protocol:
SFTP
Host name:
54.173.93.70
Port number:
22
User name:
root
Password:
Save
Cancel
Advanced...
Tools
Manage
Login
Close
Help

In case you are interested on activating a swap partition to avoid occasional out-of-memory service crashes:

```
dd if=/dev/zero of=/swapfile bs=1M count=2048
chown root:root /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
echo '/swapfile swap swap defaults 0 0' >> /etc/fstab
```

You should set the swappiness to a minimum (sysctl vm.swappiness=10 and add vm.swappiness=10 to /etc/sysctl.conf to persist it on reboot) and monitor the swap usage closely, though: Amazon charges EBS volumes based on USAGE and they will charge you A LOT if you make heavy use of them. In that case you will be saving money and getting a much greater performance by upgrading your instance to a "non-EBS only", mounting a swap partition on ephemeral storage instead if you still require it.

You have now deployed an AWS EC2 instance and you are able to connect to it. The next part of this tutorial will show how to provision this instance to set up a LAMP stack on it. That simply means installing packages using the package manager; a piece of cake. Then there will be three more parts: On the first one I will show how to obtain a signed certificate, because I am tired of visiting sites with self-signed certs when it's only 9 US dollars and a few minutes of your time to set up a signed one that will last you at least a year. After that I will publish another short part on how to configure the Maria DB SQL service, the PHP FPM service and the Apache Web Server on our CentOS 7 instance. Finally, the last part of this series will teach you how set up and deploy a PHP web application using git and composer, plus how to set up the records of a DNS zone file so your domain points to your EC2 instance and you can give your users a nice way to access your web app.

2/5: ADDING DISTRO REPOS AND PROVISIONING A CENTOS 7 LAMP INSTANCE

This is the second part of a five part series on how to set up a web application with proper SSL support and a LAMP stack on a CentOS 7 Amazon Elastic Compute Cloud virtual machine.

On this part I will show you how to provision the EC2 instance we launched on the previous part.

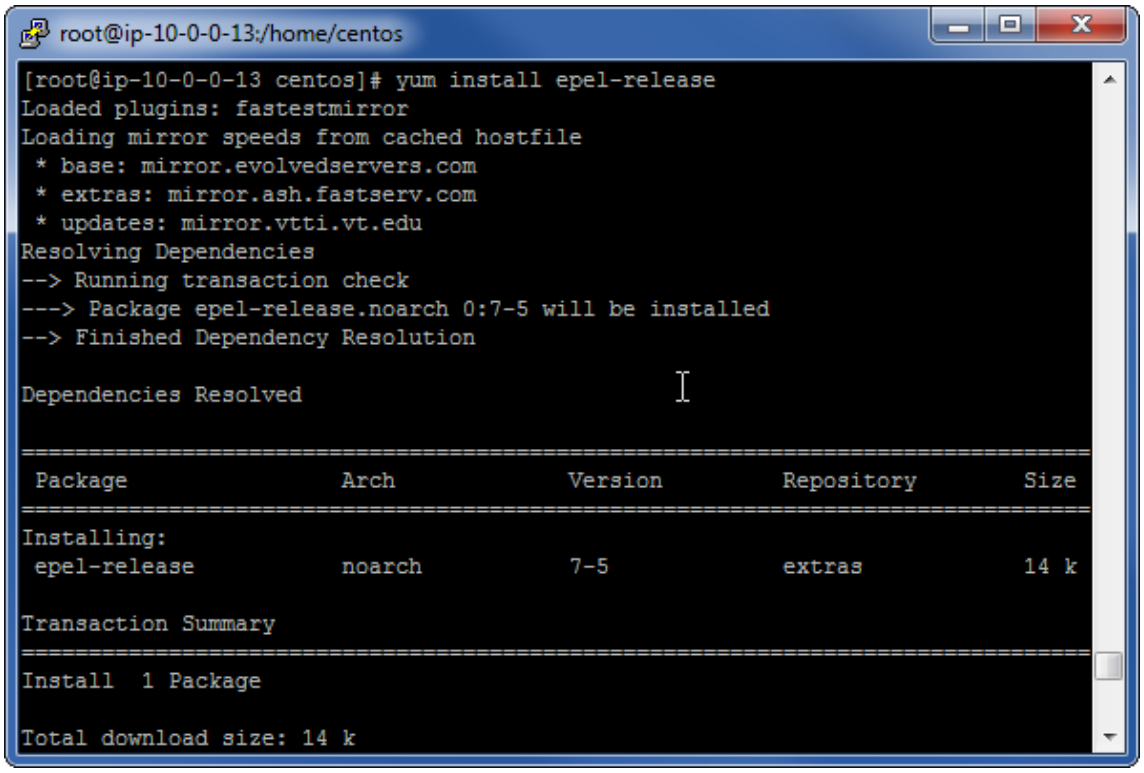
This will be a short and easy one. The goal is to install all the updated packages we need to deploy the services required by our PHP web application.

These requirements may vary depending on what kind of web application you want to deploy. At the very least you usually need a web server and PHP interpreter; Often enough you will need an SQL database too, and the capability to send mails from your PHP web app, using

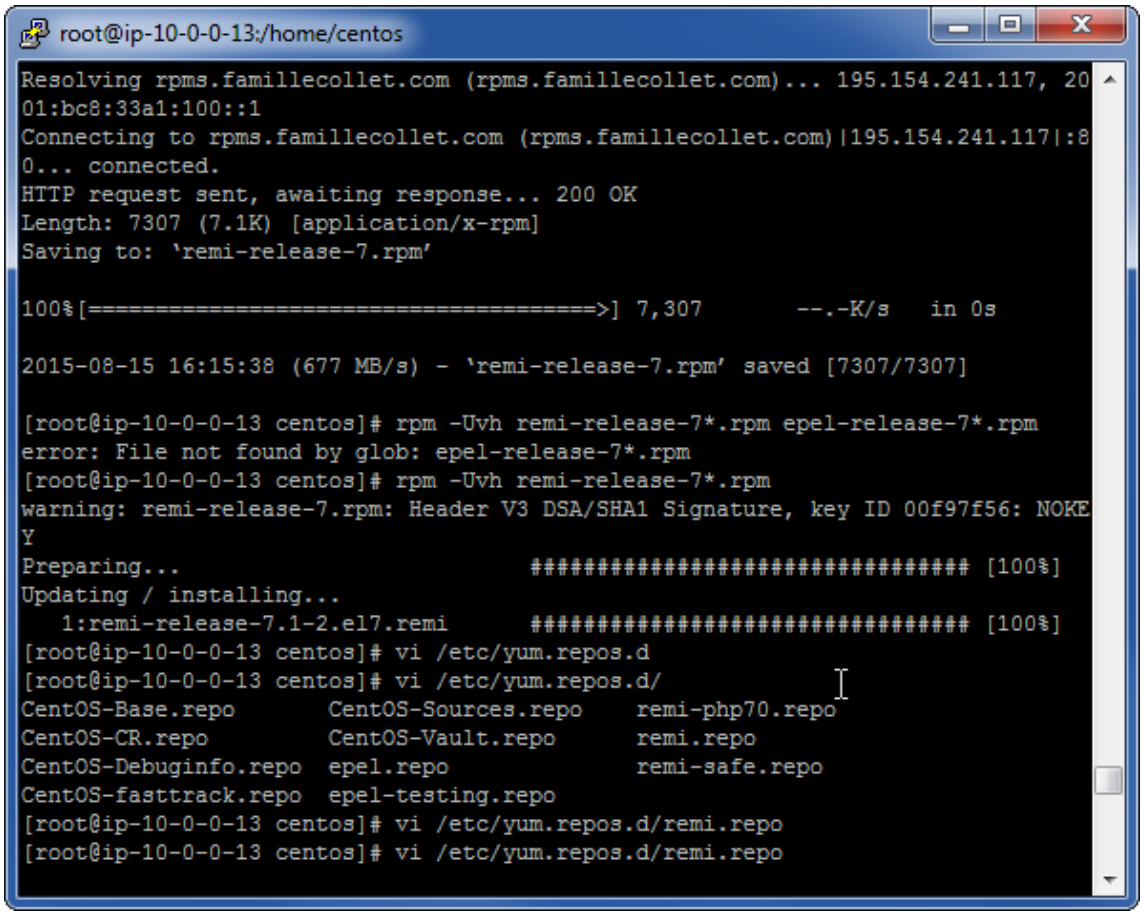
sendmail or postfix, for example. We will take care of installing the pertinent packages and all the dependencies involved.

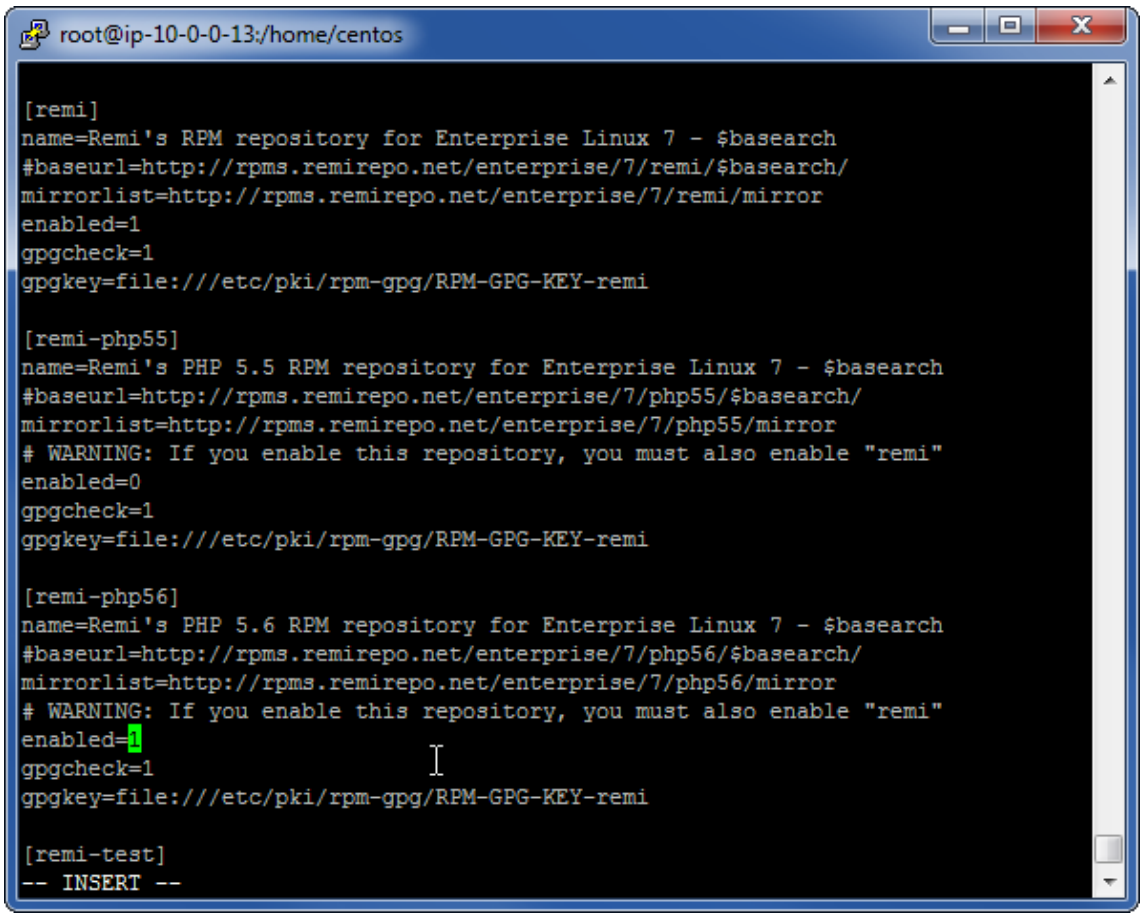
In order to get the latest stable versions of the packages we are about to install, we will start by adding a few repositories to those included on the standard CentOS release: epel, remi and MariaDB.

For epel repositories, that boils down to performing a simple yum install epel-release, as you can see in the screen capture.

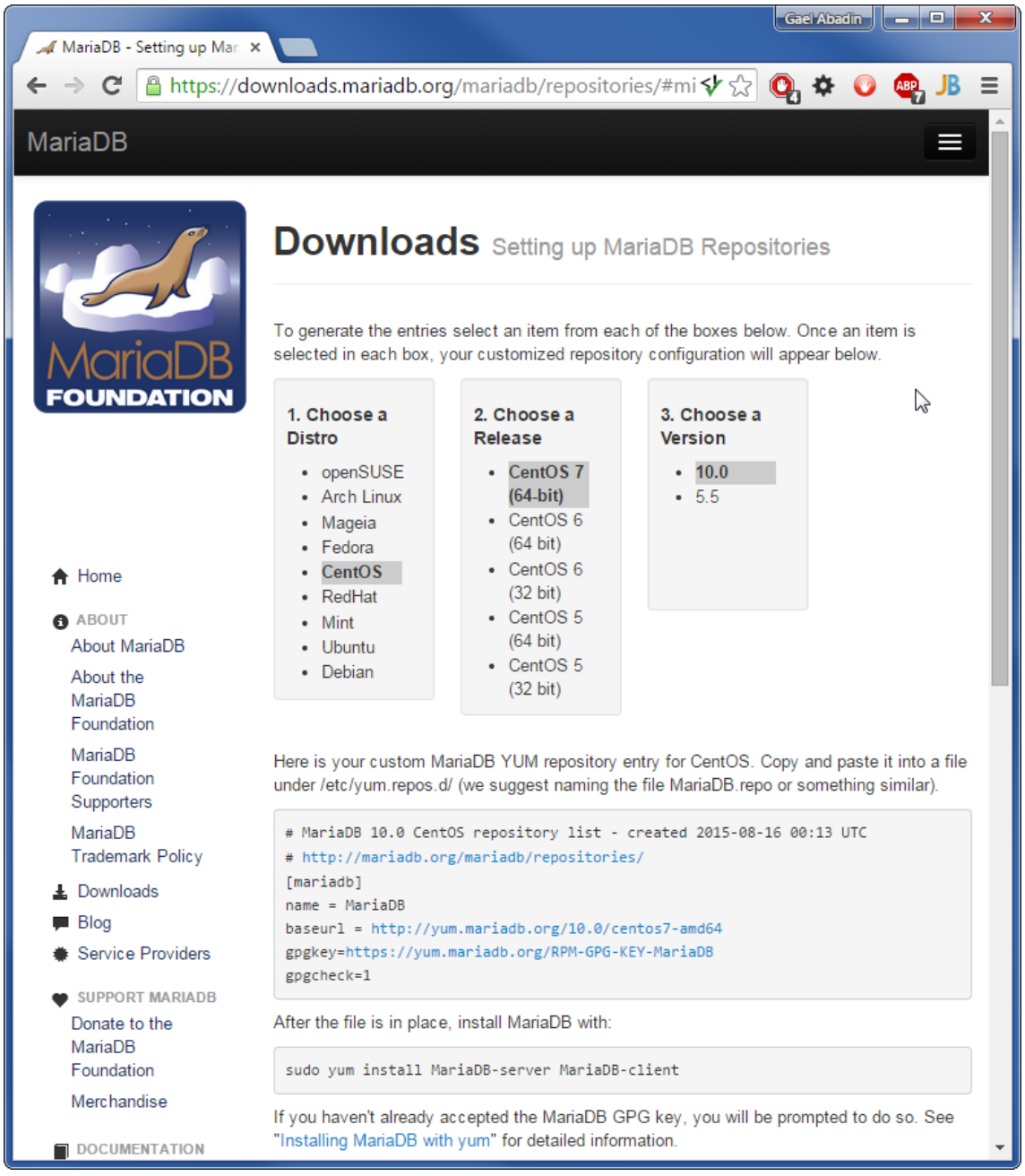


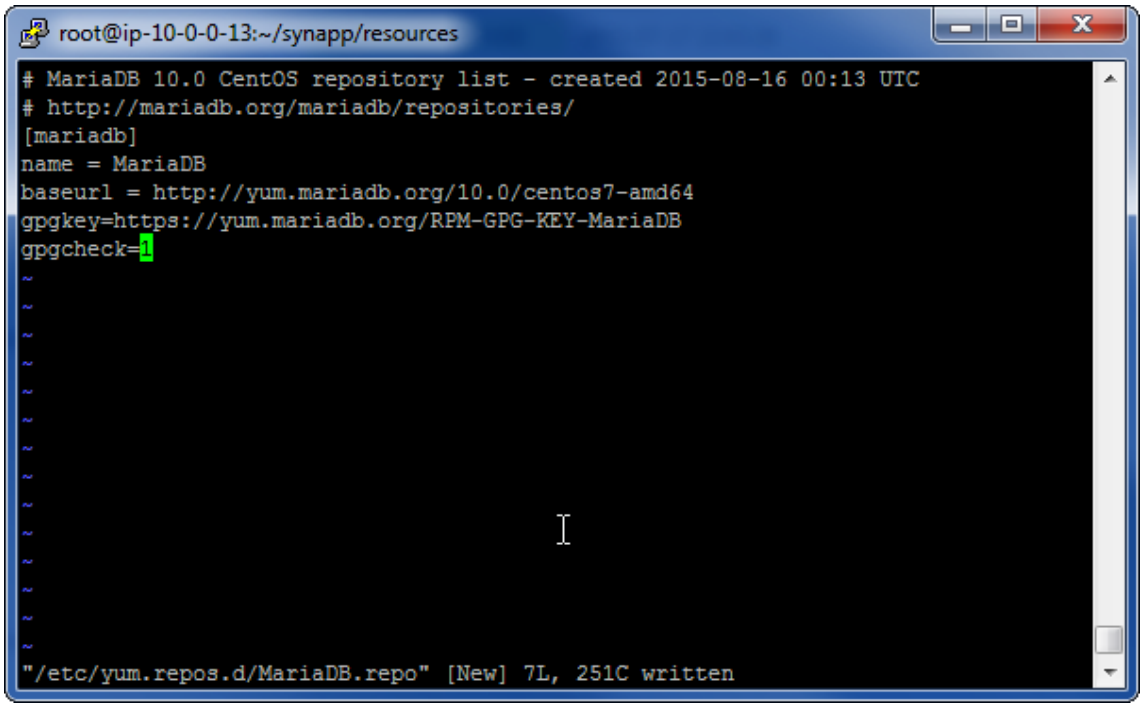
In the case of remi repos, we need to wget the rpm package and then yum-install it. After that, don't forget to edit the etc/yum.repos.d/remi.repo to activate the PHP and remi-release repositories.


root@ip-10-0-0-13:/home/centos
Resolving rpms.famillecollet.com (rpms.famillecollet.com)... 195.154.241.117, 20
01:bc8:33a1:100::1
Connecting to rpms.famillecollet.com (rpms.famillecollet.com)|195.154.241.117|:8
0... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7307 (7.1K) [application/x-rpm]
Saving to: 'remi-release-7.rpm'
100%[======================================>] 7,307 --.-K/s in 0s
2015-08-15 16:15:38 (677 MB/s) - 'remi-release-7.rpm' saved [7307/7307]
[root@ip-10-0-0-13 centos]# rpm -Uvh remi-release-7*.rpm epel-release-7*.rpm
error: File not found by glob: epel-release-7*.rpm
[root@ip-10-0-0-13 centos]# rpm -Uvh remi-release-7*.rpm
warning: remi-release-7.rpm: Header V3 DSA/SHA1 Signature, key ID 00f97f56: NOKE
Y
Preparing... ################################# [100%]
Updating / installing...
1:remi-release-7.1-2.el7.remi ################################# [100%]
[root@ip-10-0-0-13 centos]# vi /etc/yum.repos.d
[root@ip-10-0-0-13 centos]# vi /etc/yum.repos.d/
CentOS-Base.repo CentOS-Sources.repo remi-php70.repo
CentOS-CR.repo CentOS-Vault.repo remi.repo
CentOS-Debuginfo.repo epel.repo remi-safe.repo
CentOS-fasttrack.repo epel-testing.repo
[root@ip-10-0-0-13 centos]# vi /etc/yum.repos.d/remi.repo
[root@ip-10-0-0-13 centos]# vi /etc/yum.repos.d/remi.repo

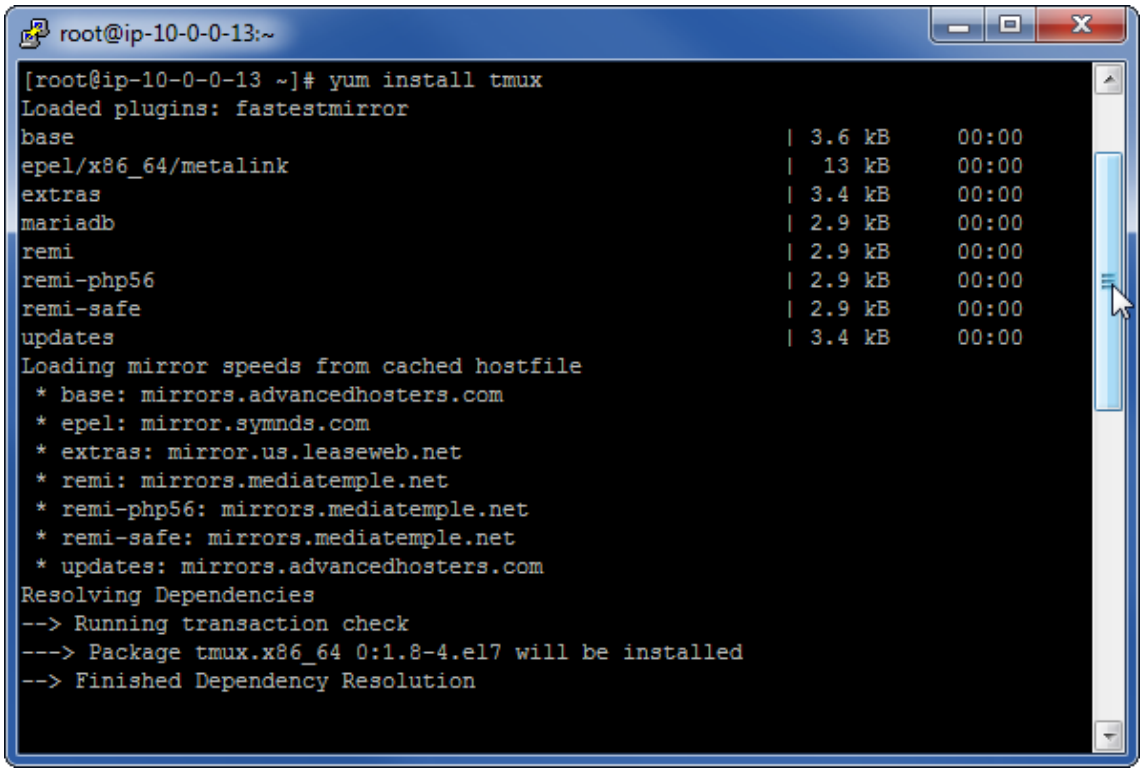

root@ip-10-0-0-13:/home/centos
[remi]
name=Remi's RPM repository for Enterprise Linux 7 - $basearch
#baseurl=http://rpms.remirepo.net/enterprise/7/remi/$basearch/
mirrorlist=http://rpms.remirepo.net/enterprise/7/remi/mirror
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-remi
[remi-php55]
name=Remi's PHP 5.5 RPM repository for Enterprise Linux 7 - $basearch
#baseurl=http://rpms.remirepo.net/enterprise/7/php55/$basearch/
mirrorlist=http://rpms.remirepo.net/enterprise/7/php55/mirror
# WARNING: If you enable this repository, you must also enable "remi"
enabled=0
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-remi
[remi-php56]
name=Remi's PHP 5.6 RPM repository for Enterprise Linux 7 - $basearch
#baseurl=http://rpms.remirepo.net/enterprise/7/php56/$basearch/
mirrorlist=http://rpms.remirepo.net/enterprise/7/php56/mirror
# WARNING: If you enable this repository, you must also enable "remi"
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-remi
[remi-test]
-- INSERT --

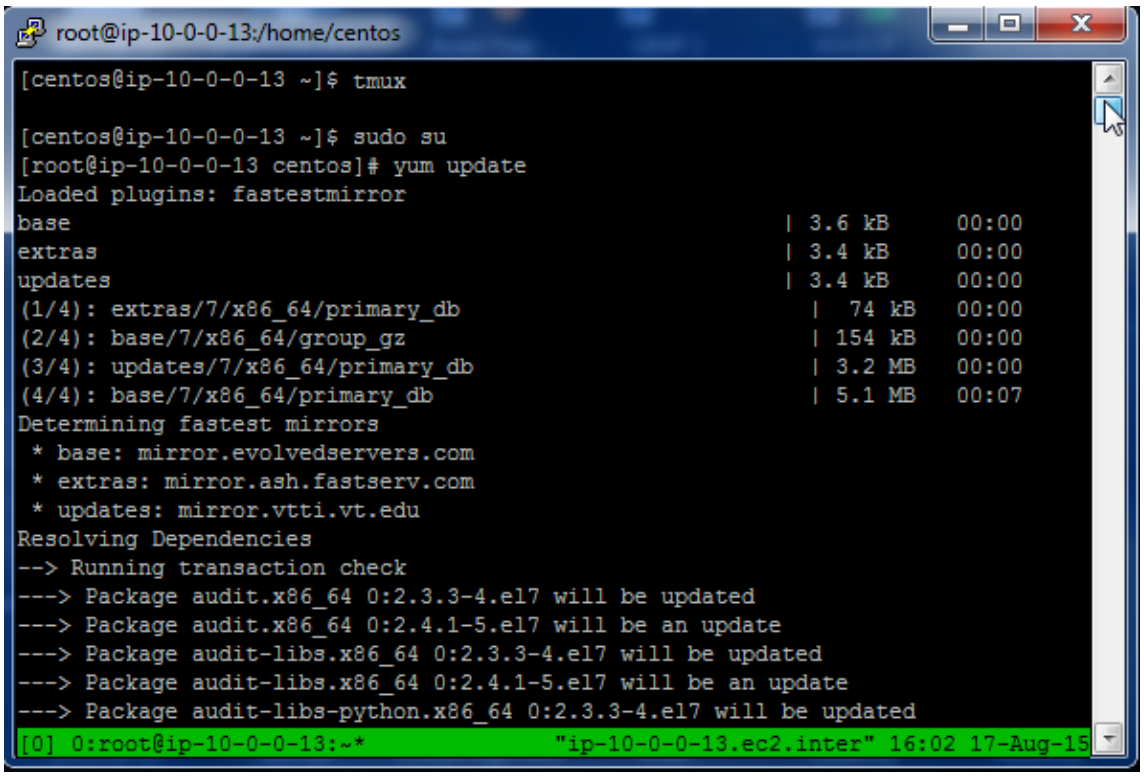
For mariadb there is a web page detailing the process (https://downloads.mariadb.org/mariadb/repositories/#mirror=tedeco&distro=CentOS&distro_release=centos7-amd64--centos7&version=10.0), which is as simple as copying the repo definition to a .repo file in etc/yum.repos.d directory

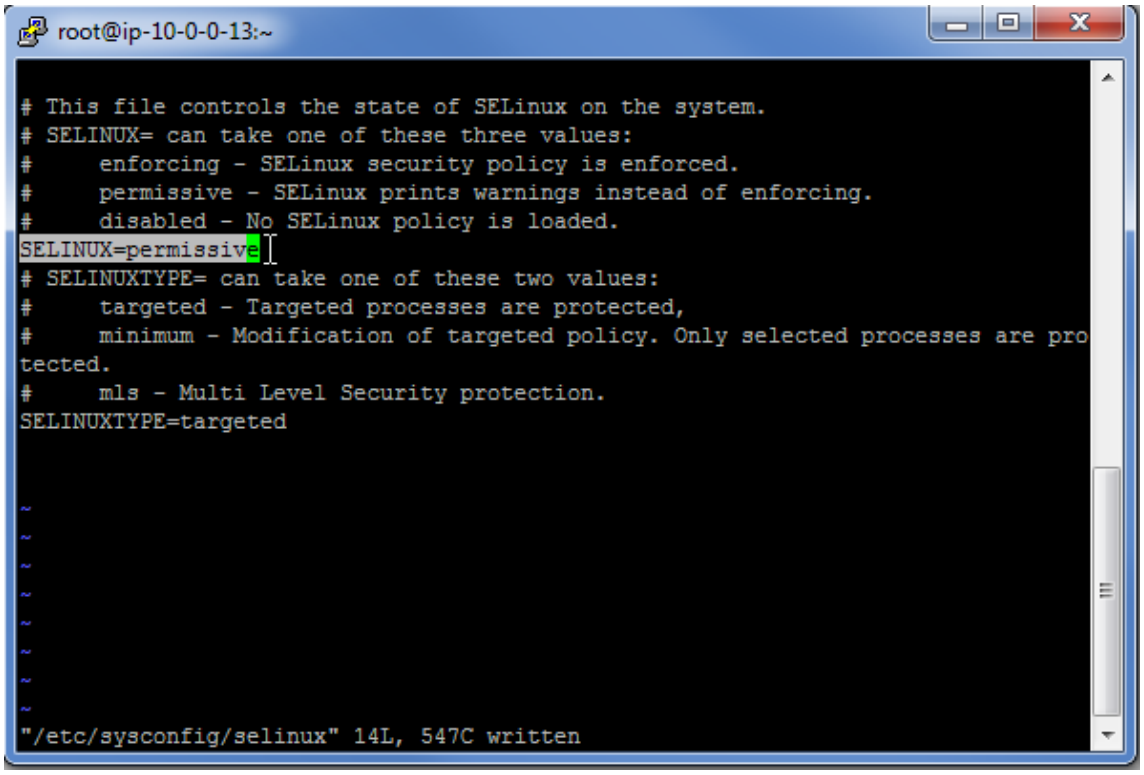




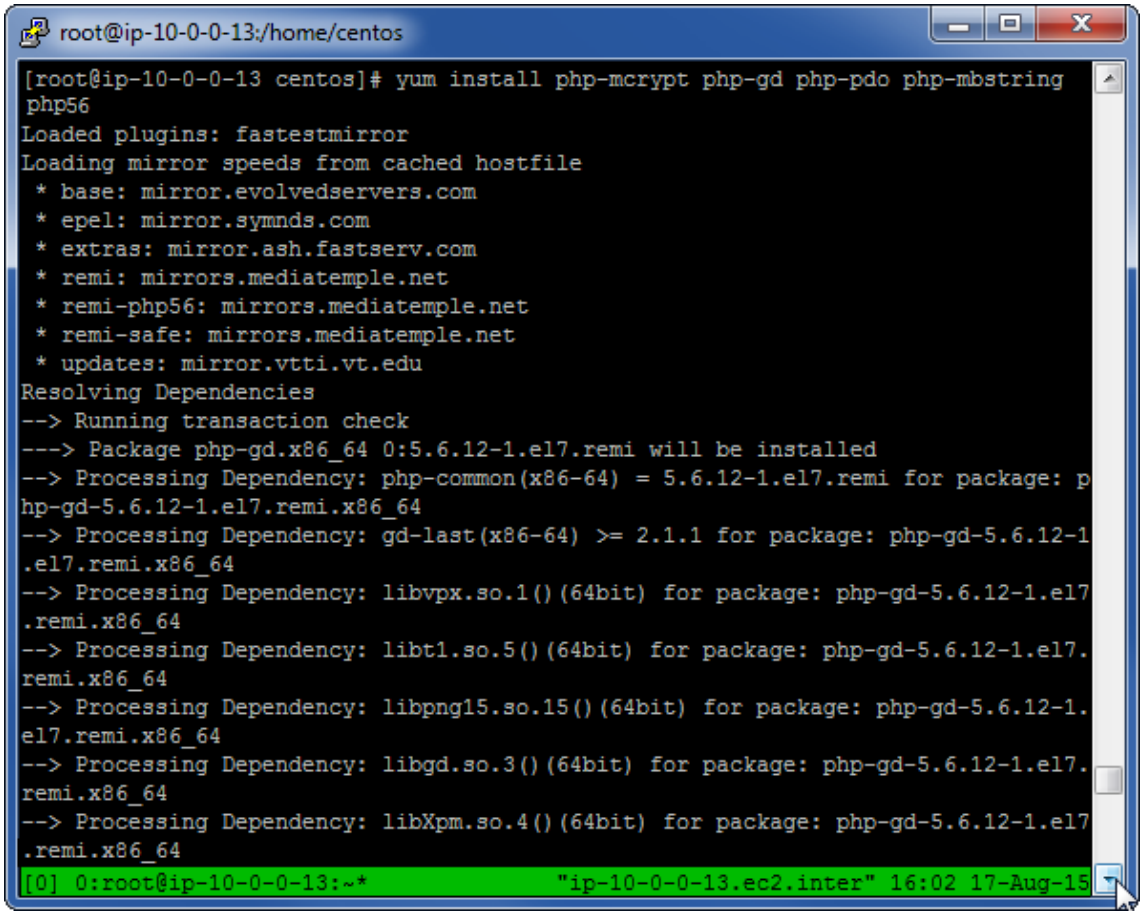
After we set up the repos, the first thing we want to do is a system update. But before that, a little piece of advice: When you are working remotely it is a good idea to use a virtual terminal session manager like tmux or gnu screen to avoid broken connections terminating lengthy processes, specially if you manage your servers from a WiFi connected terminal like I often do. A simple yum install tmux will provide you with the tmux command that you can execute without parameters to start a new session. Once you do that, just run yum update and wait for the update to finish. A kernel update will be performed, which requires a system restart for the new kernel to be loaded. Instead of doing that right away we will first edit the selinux configuration by on /etc/sysconfig/selinux file and set the SELINUX property to permissive, which also requires a system restart to take effect. Don't freak out by this global security policy change. Unless your server is a juicy target for hackers or you are a very careless sysadmin or you share admin rights on this server with a few other people which shouldn't be trusted with having god powers on a production server you should be OK. Although I admit that's a lot of ifs...

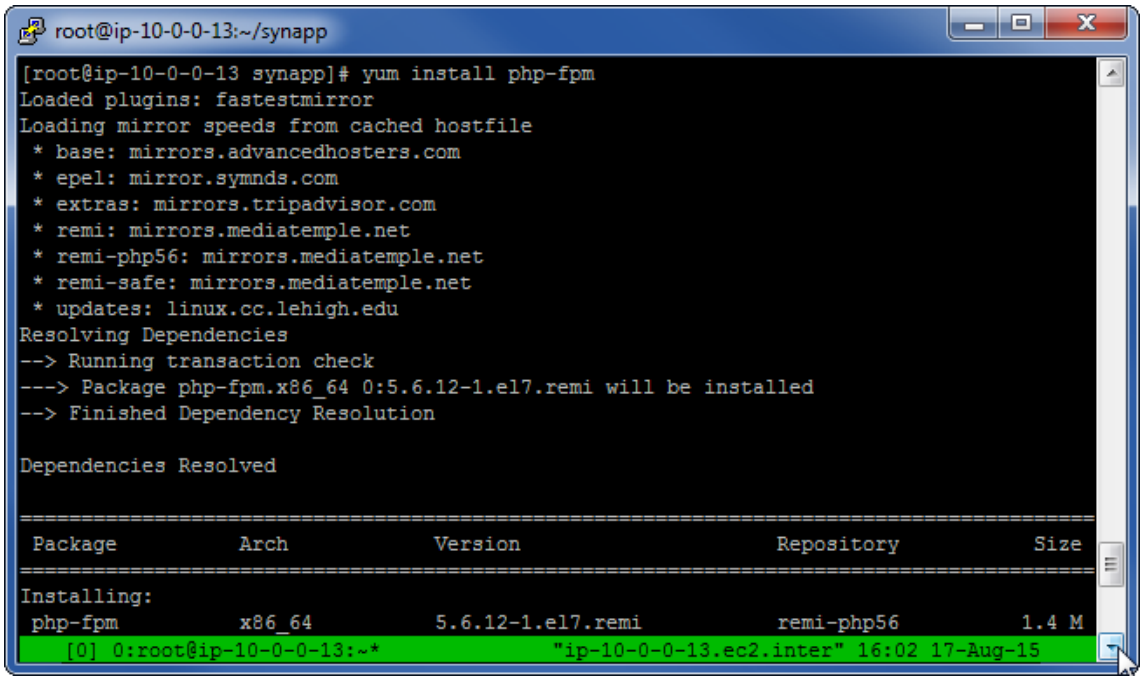

root@ip-10-0-0-13:~
[root@ip-10-0-0-13 ~]# yum install tmux
Loaded plugins: fastestmirror
base | 3.6 kB 00:00
epel/x86_64/metalink | 13 kB 00:00
extras | 3.4 kB 00:00
mariadb | 2.9 kB 00:00
remi | 2.9 kB 00:00
remi-php56 | 2.9 kB 00:00
remi-safe | 2.9 kB 00:00
updates | 3.4 kB 00:00
Loading mirror speeds from cached hostfile
* base: mirrors.advancedhosters.com
* epel: mirror.symnds.com
* extras: mirror.us.leaseweb.net
* remi: mirrors.mediatemple.net
* remi-php56: mirrors.mediatemple.net
* remi-safe: mirrors.mediatemple.net
* updates: mirrors.advancedhosters.com
Resolving Dependencies
--> Running transaction check
---> Package tmux.x86_64 0:1.8-4.el7 will be installed
--> Finished Dependency Resolution

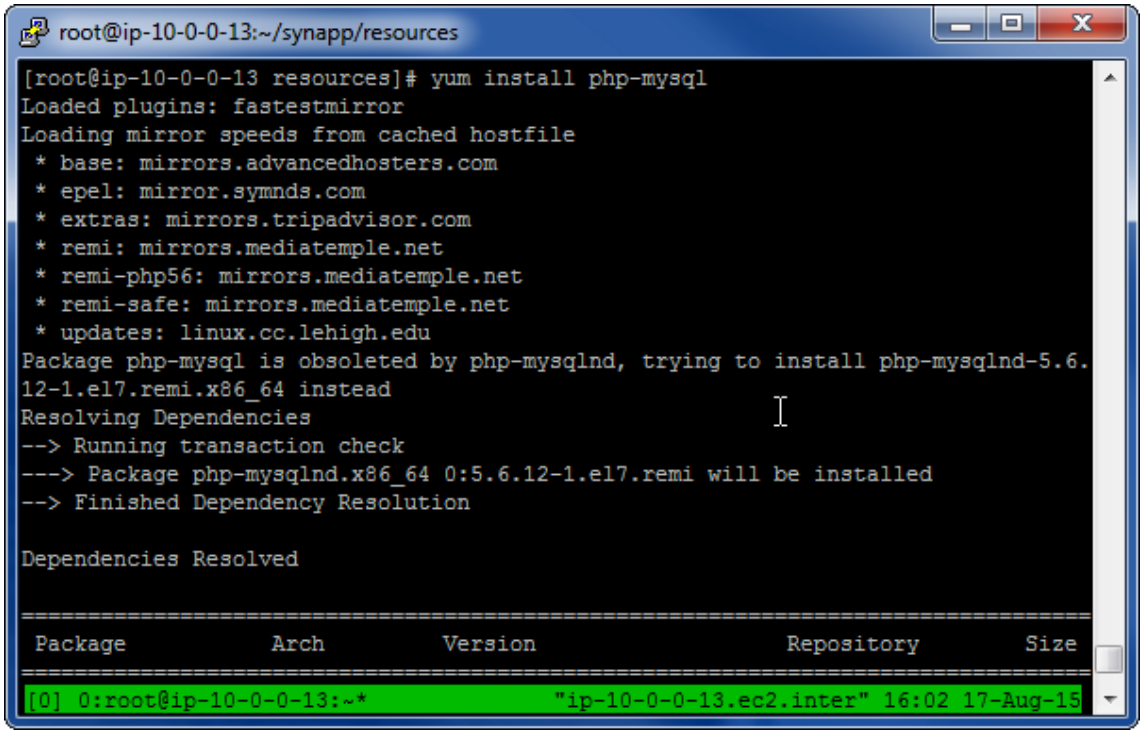

root@ip-10-0-0-13:/home/centos
[centos@ip-10-0-0-13 ~]$ tmux
[centos@ip-10-0-0-13 ~]$ sudo su
[root@ip-10-0-0-13 centos]# yum update
Loaded plugins: fastestmirror
base | 3.6 kB 00:00
extras | 3.4 kB 00:00
updates | 3.4 kB 00:00
(1/4): extras/7/x86_64/primary_db | 74 kB 00:00
(2/4): base/7/x86_64/group_gz | 154 kB 00:00
(3/4): updates/7/x86_64/primary_db | 3.2 MB 00:00
(4/4): base/7/x86_64/primary_db | 5.1 MB 00:07
Determining fastest mirrors
* base: mirror.evolvedservers.com
* extras: mirror.ash.fastserv.com
* updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package audit.x86_64 0:2.3.3-4.el7 will be updated
---> Package audit.x86_64 0:2.4.1-5.el7 will be an update
---> Package audit-libs.x86_64 0:2.3.3-4.el7 will be updated
---> Package audit-libs.x86_64 0:2.4.1-5.el7 will be an update
---> Package audit-libs-python.x86_64 0:2.3.3-4.el7 will be updated
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15



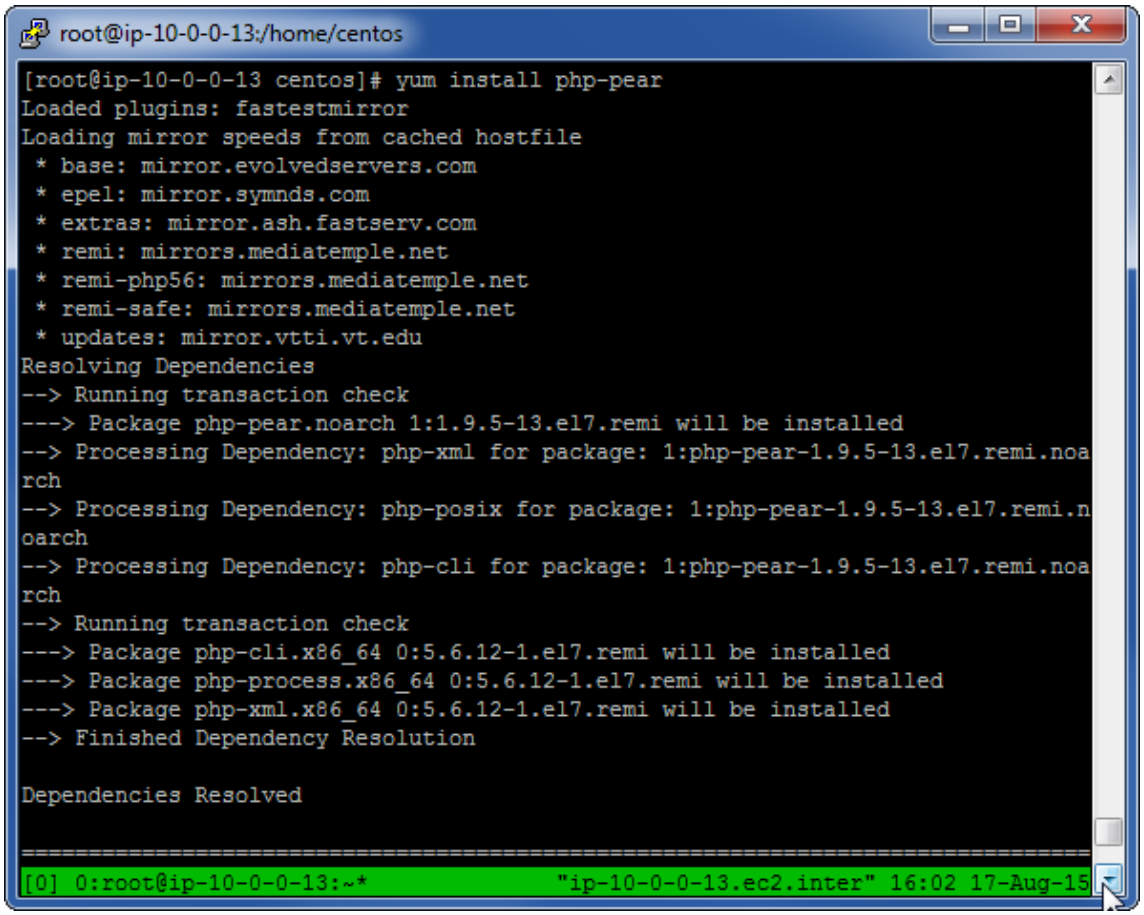
All that's left to do now is performing a yum-install on the required packages, as you can see in the following screen captures. You can remove any package you don't think you are going to need, although I suggest you leave them all, just in case.

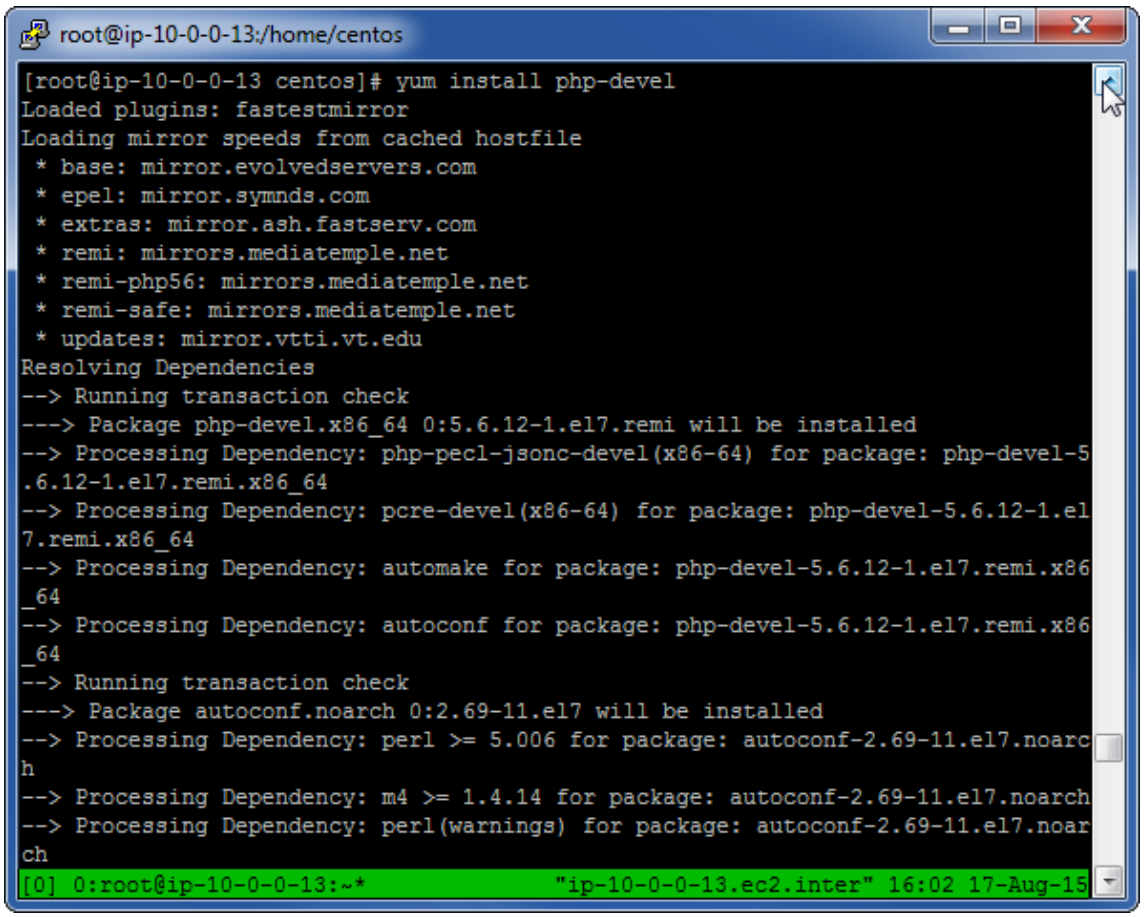


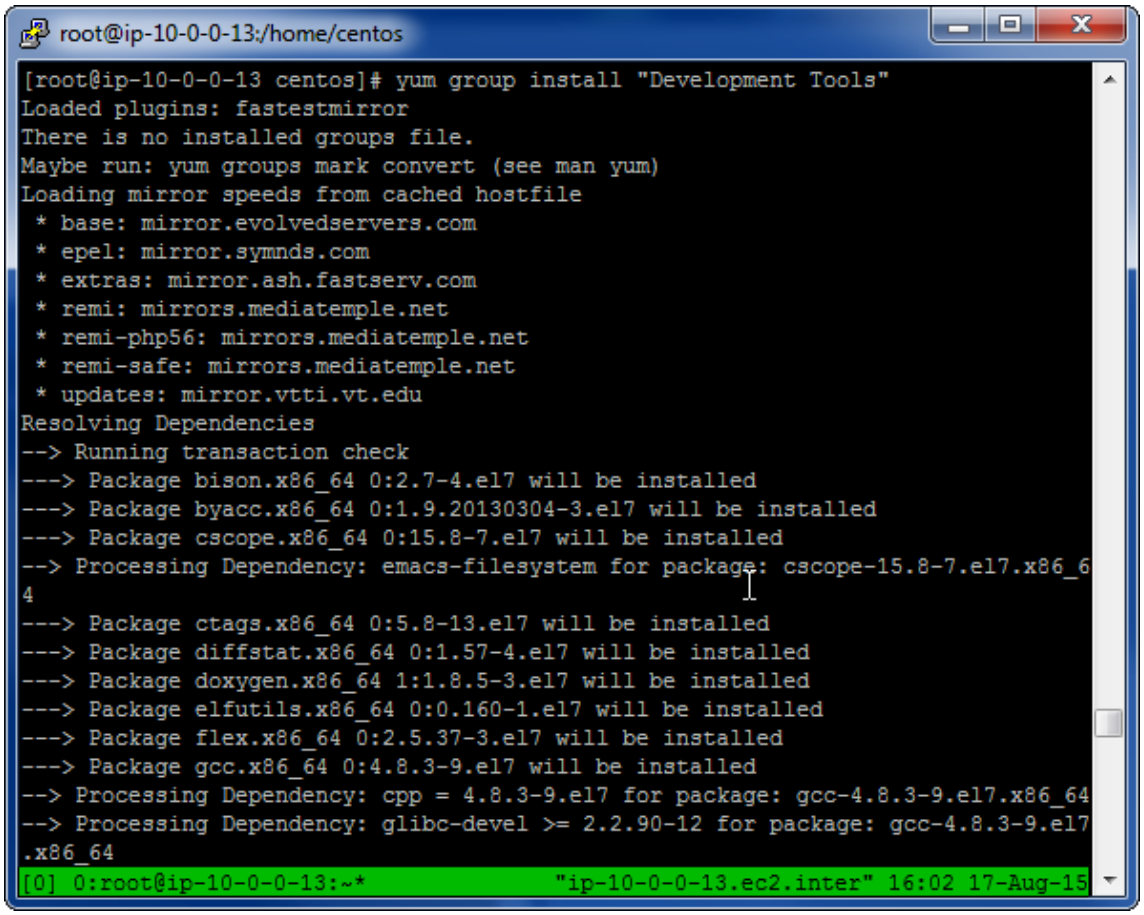


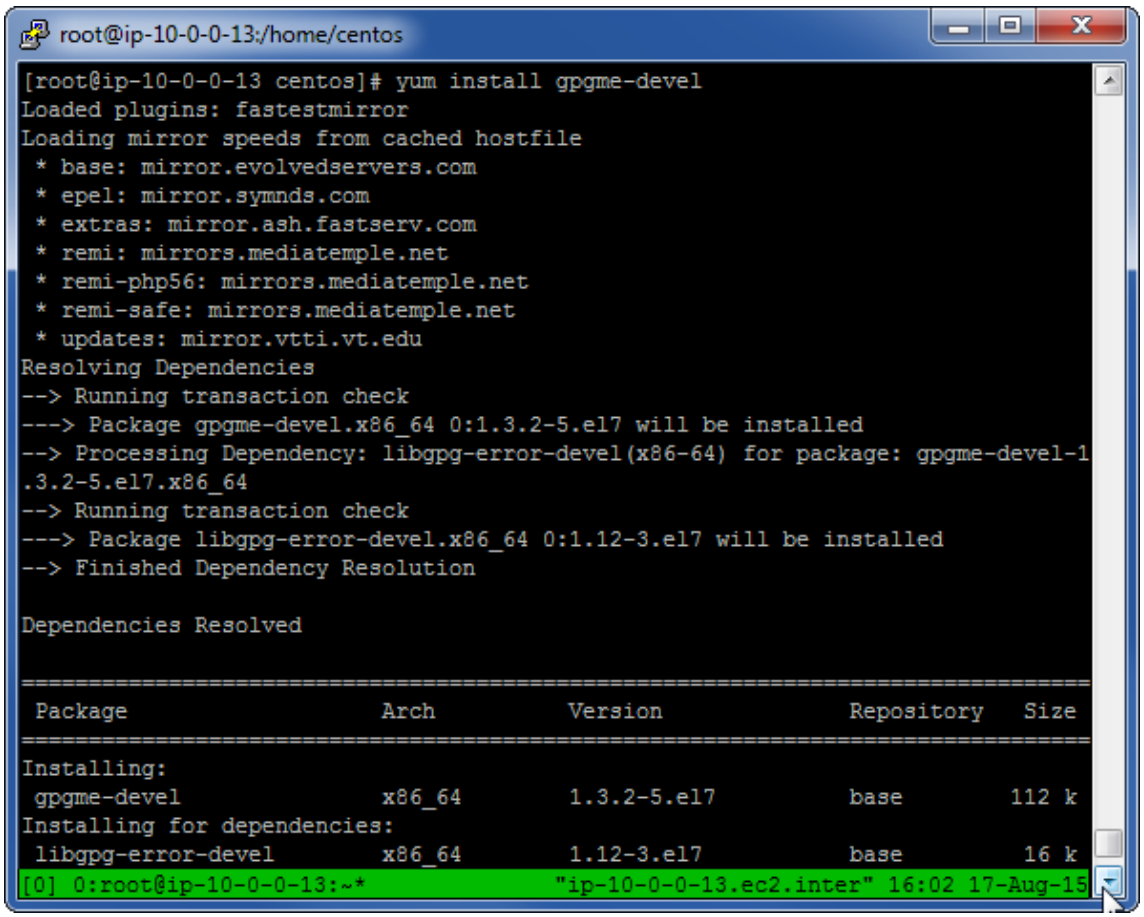


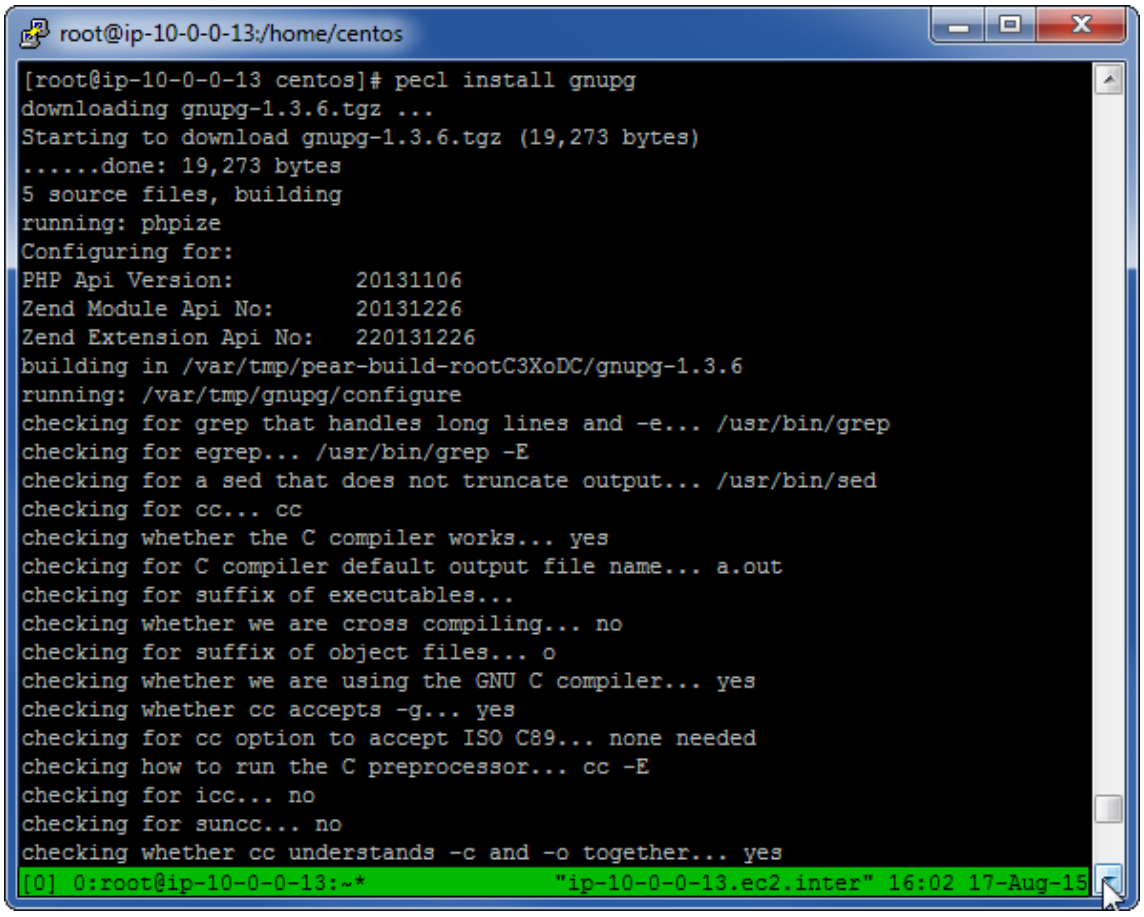
On the last part of this tutorial I will be deploying a PHP web application with GPG encryption support, which doesn't come packaged on CentOS, bringing up the opportunity to show how simple is to install PECL and the development tools required to build a PHP PECL extension from the PECL repository. This is something you should actually skip if you don't need it, because if will install a lot of dependencies that will take a relatively significant amount of space for an 8 GB drive

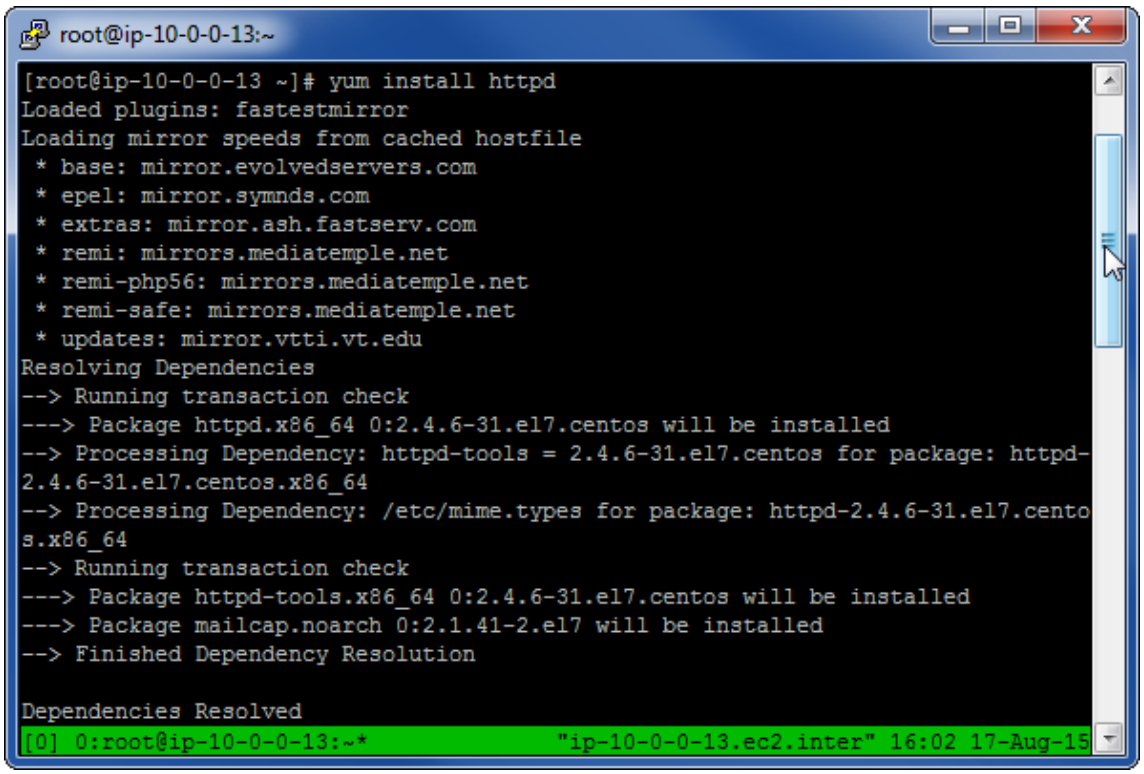

root@ip-10-0-0-13:/home/centos
[root@ip-10-0-0-13 centos]# yum install php-pear
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.evolvedservers.com
 * epel: mirror.symnds.com
 * extras: mirror.ash.fastserv.com
 * remi: mirrors.mediatemple.net
 * remi-php56: mirrors.mediatemple.net
 * remi-safe: mirrors.mediatemple.net
 * updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package php-pear.noarch 1:1.9.5-13.el7.remi will be installed
--> Processing Dependency: php-xml for package: 1:php-pear-1.9.5-13.el7.remi.noa
rch
--> Processing Dependency: php-posix for package: 1:php-pear-1.9.5-13.el7.remi.n
oarch
--> Processing Dependency: php-cli for package: 1:php-pear-1.9.5-13.el7.remi.noa
rch
--> Running transaction check
---> Package php-cli.x86_64 0:5.6.12-1.el7.remi will be installed
---> Package php-process.x86_64 0:5.6.12-1.el7.remi will be installed
---> Package php-xml.x86_64 0:5.6.12-1.el7.remi will be installed
--> Finished Dependency Resolution
Dependencies Resolved
[0] 0:root@ip-10-0-0-13:~*
"ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15

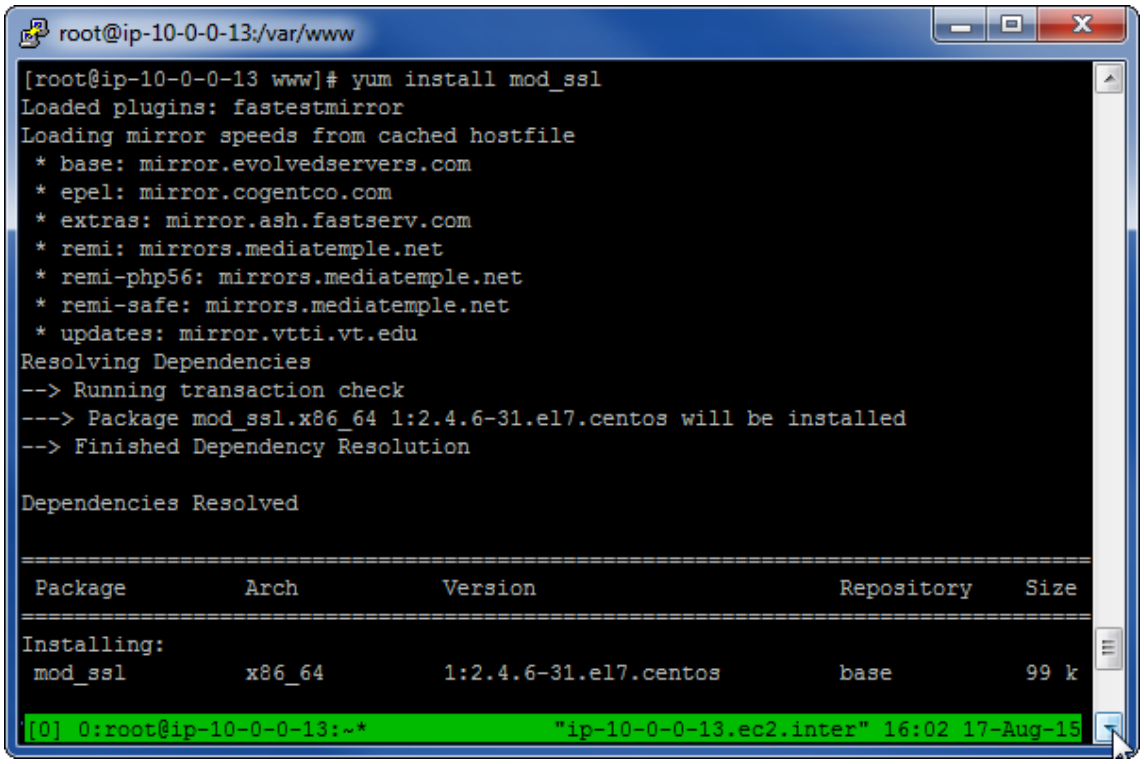

root@ip-10-0-0-13:/home/centos
[root@ip-10-0-0-13 centos]# yum install php-devel
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.evolvedservers.com
 * epel: mirror.symnds.com
 * extras: mirror.ash.fastserv.com
 * remi: mirrors.mediatemple.net
 * remi-php56: mirrors.mediatemple.net
 * remi-safe: mirrors.mediatemple.net
 * updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package php-devel.x86_64 0:5.6.12-1.el7.remi will be installed
--> Processing Dependency: php-pecl-jsonc-devel(x86-64) for package: php-devel-5
.6.12-1.el7.remi.x86_64
--> Processing Dependency: pcre-devel(x86-64) for package: php-devel-5.6.12-1.el
7.remi.x86_64
--> Processing Dependency: automake for package: php-devel-5.6.12-1.el7.remi.x86
_64
--> Processing Dependency: autoconf for package: php-devel-5.6.12-1.el7.remi.x86
_64
--> Running transaction check
---> Package autoconf.noarch 0:2.69-11.el7 will be installed
--> Processing Dependency: perl >= 5.006 for package: autoconf-2.69-11.el7.noarc
h
--> Processing Dependency: m4 >= 1.4.14 for package: autoconf-2.69-11.el7.noarch
--> Processing Dependency: perl(warnings) for package: autoconf-2.69-11.el7.noar
ch
[0] 0:root@ip-10-0-0-13:~*
"ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15

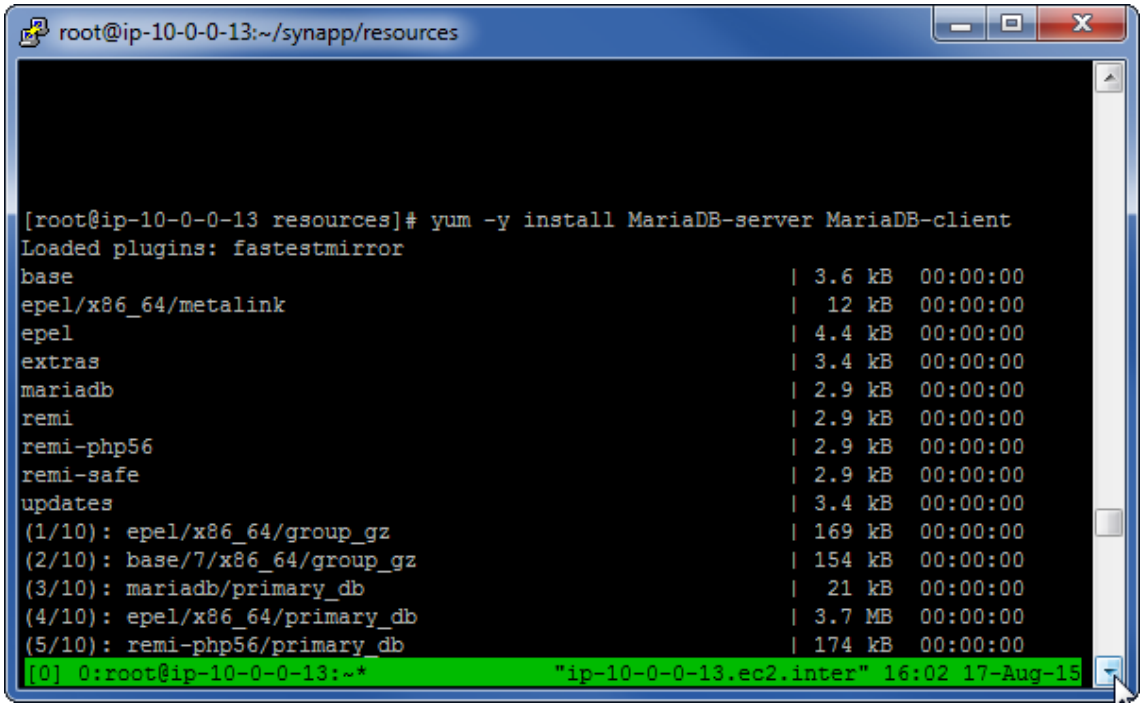

root@ip-10-0-0-13:/home/centos
[root@ip-10-0-0-13 centos]# yum group install "Development Tools"
Loaded plugins: fastestmirror
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
Loading mirror speeds from cached hostfile
* base: mirror.evolvedservers.com
* epel: mirror.symnds.com
* extras: mirror.ash.fastserv.com
* remi: mirrors.mediatemple.net
* remi-php56: mirrors.mediatemple.net
* remi-safe: mirrors.mediatemple.net
* updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package bison.x86_64 0:2.7-4.el7 will be installed
---> Package byacc.x86_64 0:1.9.20130304-3.el7 will be installed
---> Package cscope.x86_64 0:15.8-7.el7 will be installed
--> Processing Dependency: emacs-filesystem for package: cscope-15.8-7.el7.x86_64
---> Package ctags.x86_64 0:5.8-13.el7 will be installed
---> Package diffstat.x86_64 0:1.57-4.el7 will be installed
---> Package doxygen.x86_64 1:1.8.5-3.el7 will be installed
---> Package elfutils.x86_64 0:0.160-1.el7 will be installed
---> Package flex.x86_64 0:2.5.37-3.el7 will be installed
---> Package gcc.x86_64 0:4.8.3-9.el7 will be installed
--> Processing Dependency: cpp = 4.8.3-9.el7 for package: gcc-4.8.3-9.el7.x86_64
--> Processing Dependency: glibc-devel >= 2.2.90-12 for package: gcc-4.8.3-9.el7.x86_64
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15

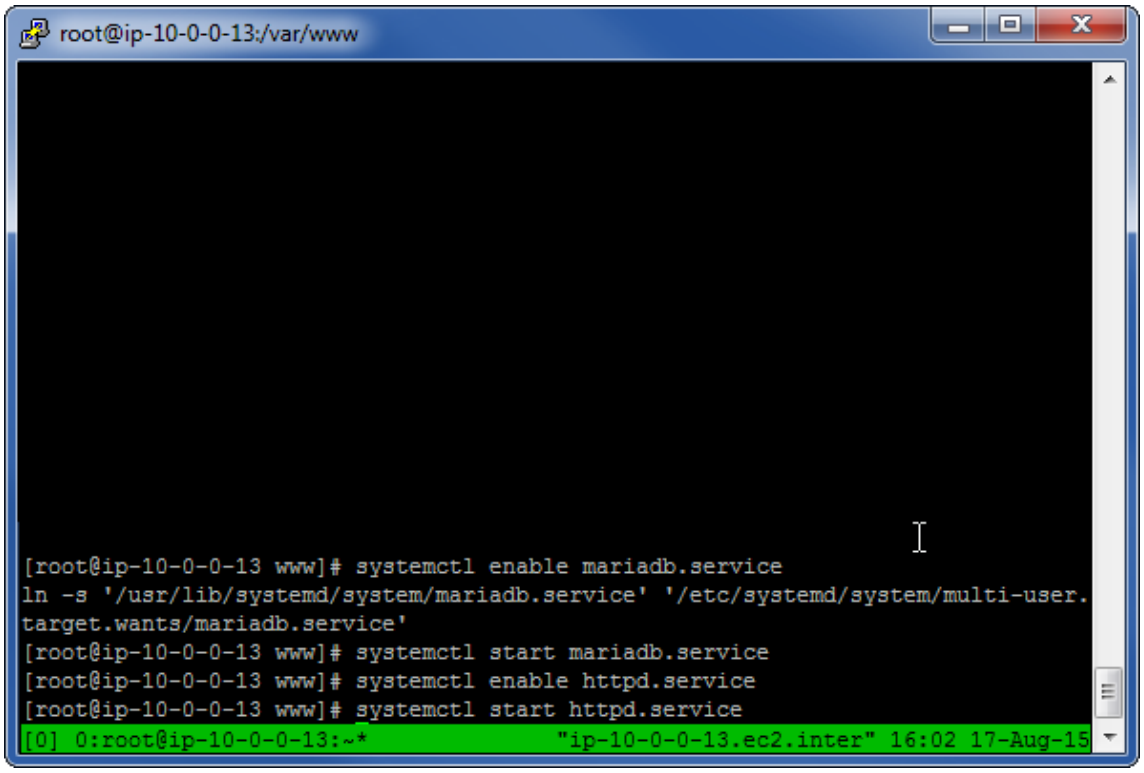

root@ip-10-0-0-13:/home/centos
[root@ip-10-0-0-13 centos]# yum install gpgme-devel
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
* base: mirror.evolvedservers.com
* epel: mirror.symnds.com
* extras: mirror.ash.fastserv.com
* remi: mirrors.mediatemple.net
* remi-php56: mirrors.mediatemple.net
* remi-safe: mirrors.mediatemple.net
* updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package gpgme-devel.x86_64 0:1.3.2-5.el7 will be installed
--> Processing Dependency: libgpg-error-devel(x86-64) for package: gpgme-devel-1.3.2-5.el7.x86_64
--> Running transaction check
---> Package libgpg-error-devel.x86_64 0:1.12-3.el7 will be installed
--> Finished Dependency Resolution
Dependencies Resolved
Package Arch Version Repository Size
Installing:
gpgme-devel x86_64 1.3.2-5.el7 base 112 k
Installing for dependencies:
libgpg-error-devel x86_64 1.12-3.el7 base 16 k
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15

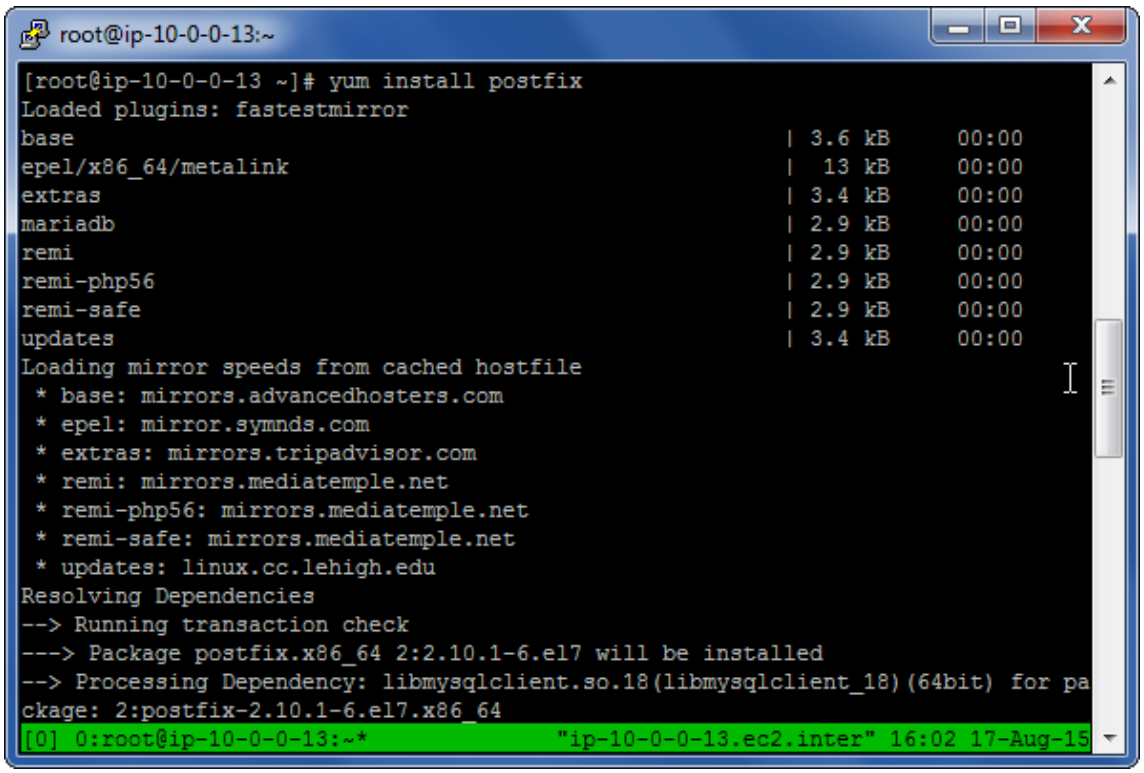

root@ip-10-0-0-13:/home/centos
[root@ip-10-0-0-13 centos]# pecl install gnupg
downloading gnupg-1.3.6.tgz ...
Starting to download gnupg-1.3.6.tgz (19,273 bytes)
......done: 19,273 bytes
5 source files, building
running: phpize
Configuring for:
PHP Api Version: 20131106
Zend Module Api No: 20131226
Zend Extension Api No: 220131226
building in /var/tmp/pear-build-rootC3XoDC/gnupg-1.3.6
running: /var/tmp/gnupg/configure
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for a sed that does not truncate output... /usr/bin/sed
checking for cc... cc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether cc accepts -g... yes
checking for cc option to accept ISO C89... none needed
checking how to run the C preprocessor... cc -E
checking for icc... no
checking for suncc... no
checking whether cc understands -c and -o together... yes
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15


root@ip-10-0-0-13:~
[root@ip-10-0-0-13 ~]# yum install httpd
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
* base: mirror.evolvedservers.com
* epel: mirror.symnds.com
* extras: mirror.ash.fastserv.com
* remi: mirrors.mediatemple.net
* remi-php56: mirrors.mediatemple.net
* remi-safe: mirrors.mediatemple.net
* updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.4.6-31.el7.centos will be installed
--> Processing Dependency: httpd-tools = 2.4.6-31.el7.centos for package: httpd-2.4.6-31.el7.centos.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.6-31.el7.centos.x86_64
--> Running transaction check
---> Package httpd-tools.x86_64 0:2.4.6-31.el7.centos will be installed
---> Package mailcap.noarch 0:2.1.41-2.el7 will be installed
--> Finished Dependency Resolution
Dependencies Resolved
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15


root@ip-10-0-0-13:/var/www
[root@ip-10-0-0-13 www]# yum install mod_ssl
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.evolvedservers.com
 * epel: mirror.cogentco.com
 * extras: mirror.ash.fastserv.com
 * remi: mirrors.mediatemple.net
 * remi-php56: mirrors.mediatemple.net
 * remi-safe: mirrors.mediatemple.net
 * updates: mirror.vtti.vt.edu
Resolving Dependencies
--> Running transaction check
---> Package mod_ssl.x86_64 1:2.4.6-31.el7.centos will be installed
--> Finished Dependency Resolution
Dependencies Resolved
Package Arch Version Repository Size
Installing:
mod_ssl x86_64 1:2.4.6-31.el7.centos base 99 k
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15


root@ip-10-0-0-13:~/synapp/resources
[root@ip-10-0-0-13 resources]# yum -y install MariaDB-server MariaDB-client
Loaded plugins: fastestmirror
base | 3.6 kB 00:00:00
epel/x86_64/metalink | 12 kB 00:00:00
epel | 4.4 kB 00:00:00
extras | 3.4 kB 00:00:00
mariadb | 2.9 kB 00:00:00
remi | 2.9 kB 00:00:00
remi-php56 | 2.9 kB 00:00:00
remi-safe | 2.9 kB 00:00:00
updates | 3.4 kB 00:00:00
(1/10): epel/x86_64/group_gz | 169 kB 00:00:00
(2/10): base/7/x86_64/group_gz | 154 kB 00:00:00
(3/10): mariadb/primary_db | 21 kB 00:00:00
(4/10): epel/x86_64/primary_db | 3.7 MB 00:00:00
(5/10): remi-php56/primary_db | 174 kB 00:00:00
[0] 0:root@ip-10-0-0-13:~* "ip-10-0-0-13.ec2.inter" 16:02 17-Aug-15





And that's all. We have reached the end of this part and your instance is now provisioned with the services required to run a PHP web application. Before I show you on another part how to configure these services, I suggest you see the next part on this tutorial where you can see how to obtain a cheap 9 US dollar signed certificate to securely deploy your web application, protecting your users against somebody stealing your server's identity. If you are not concerned about your users being scammed that way, you may skip that part and go straight

ahead to the services configuration part, which should be listed on the tutorial playlist and the description below.

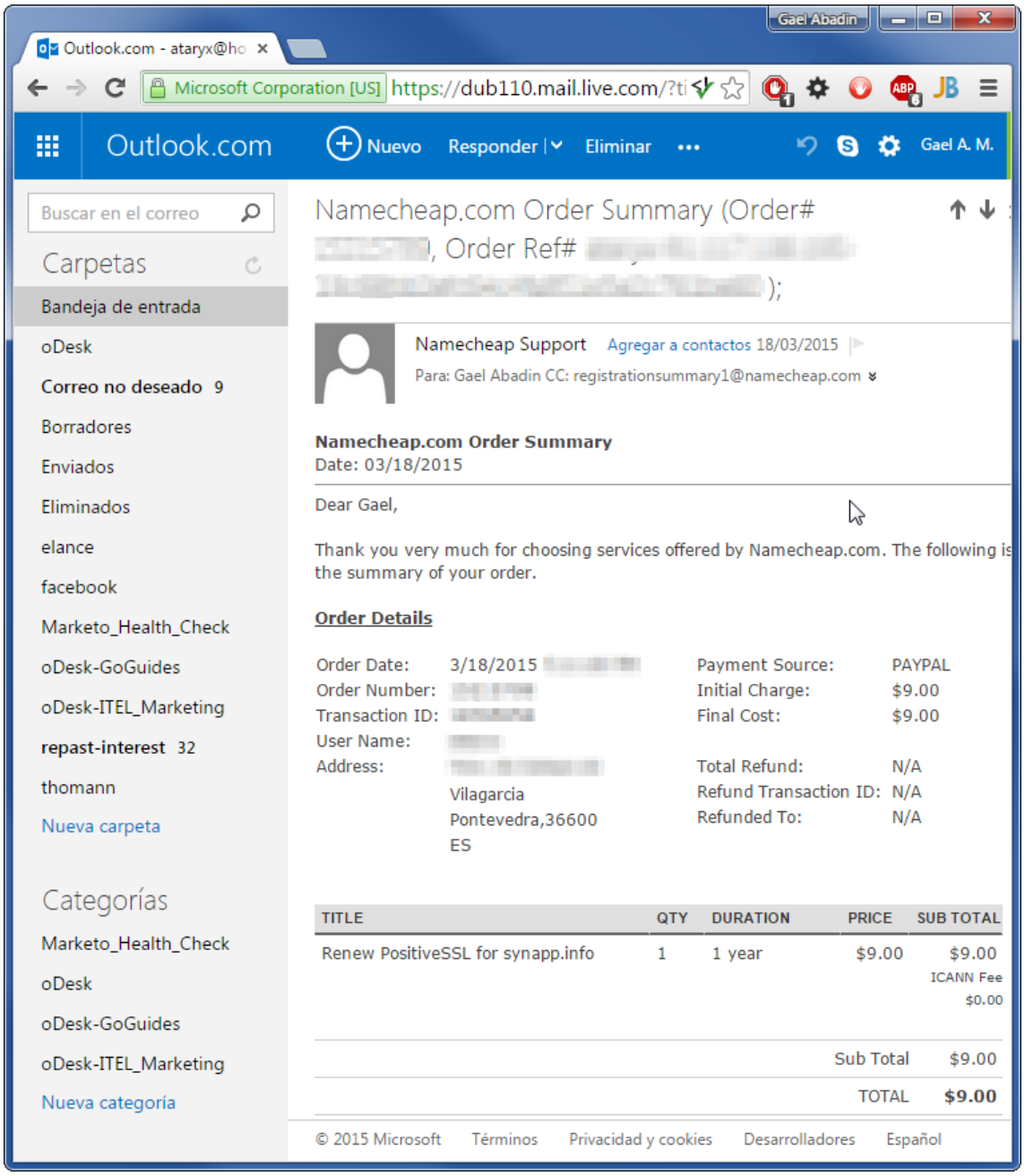
3/5: OBTAINING A VALID CA SIGNED SSL CERTIFICATE

The third part of this five part series on how to deploy an SSL enhanced web application and bring up a CentOS LAMP stack on the Amazon cloud from the ground.

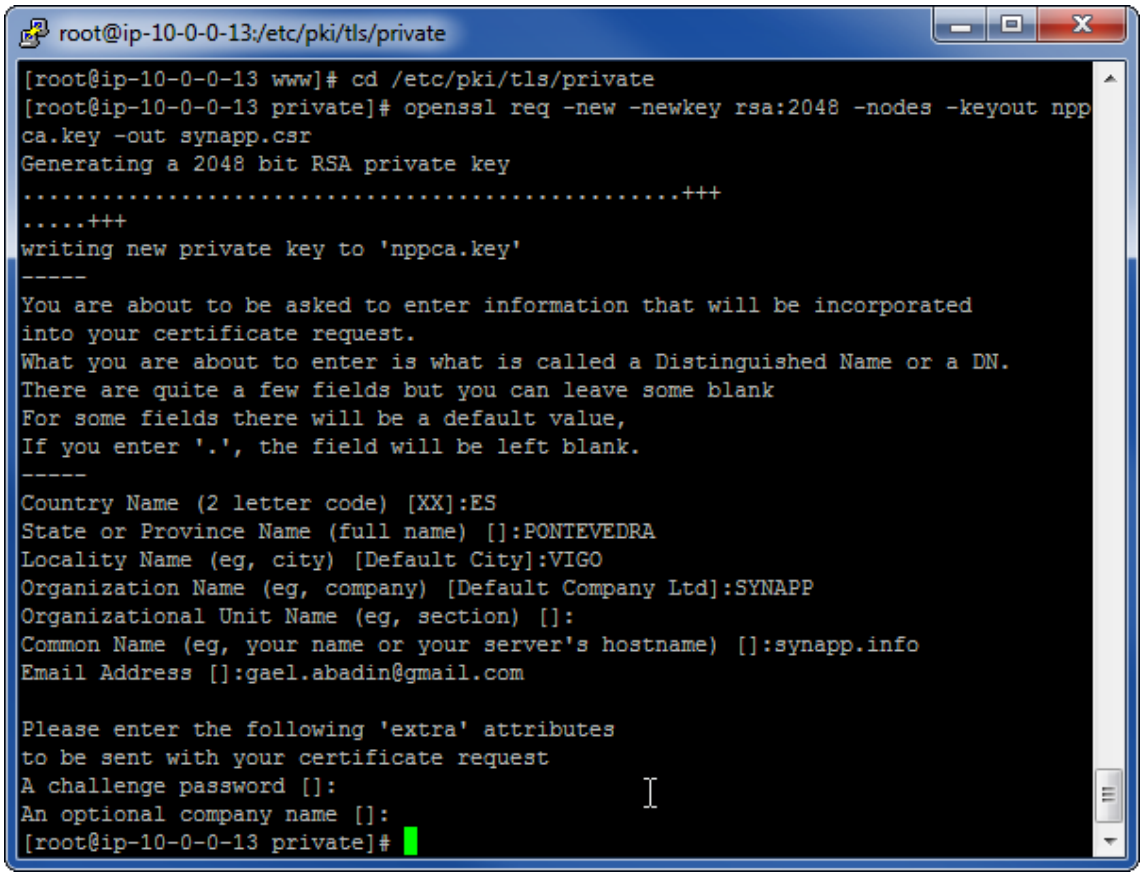
On this part I will shou you how to get your certificate signed so your users can trust that your webapp is your webapp and not some impersonator trying to do nasty things.

I want to demonstrate how easy and simple this procedure is by clearly explaining all there is to know in less than two minutes so let's cut to the chase:

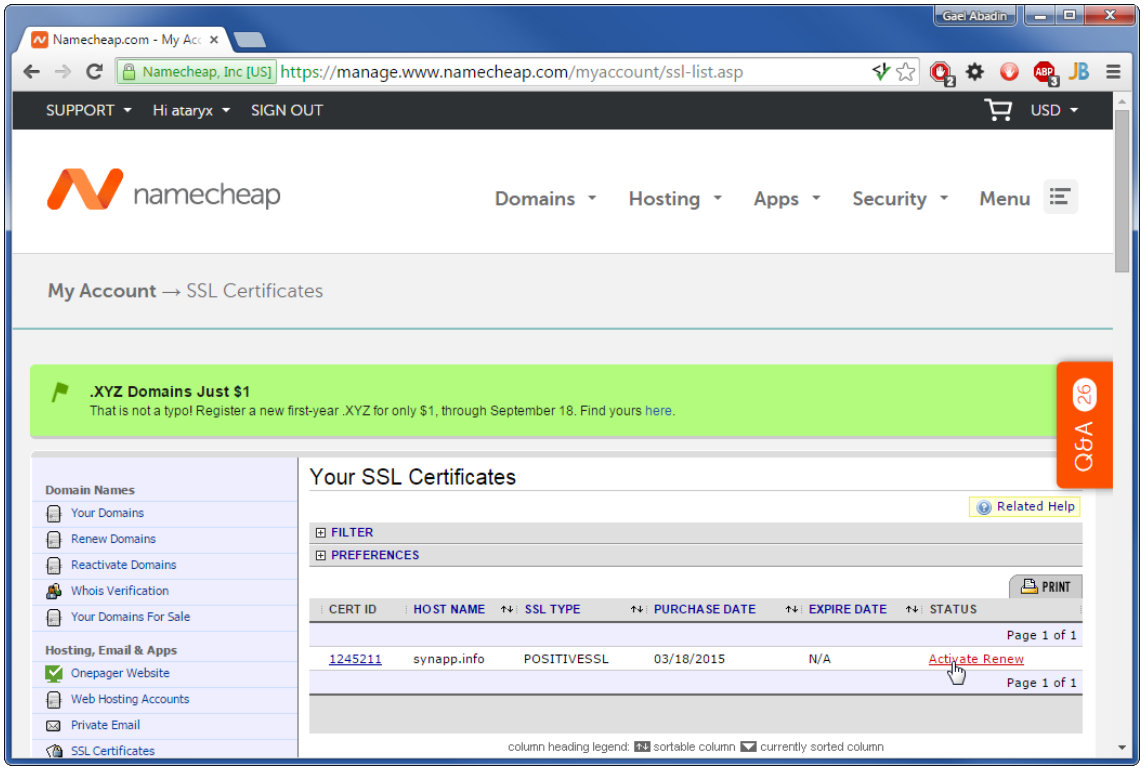
Go to namecheap.com or whatever registrar you like best and put in your cart the cheapest SSL certificate you find. Yes, the cheapest will do; you don't need anything else. Here is my confirmation order I received the moment I entered my billing details. As simple as buying anything on eBay or Amazon or any other place on the Internet.

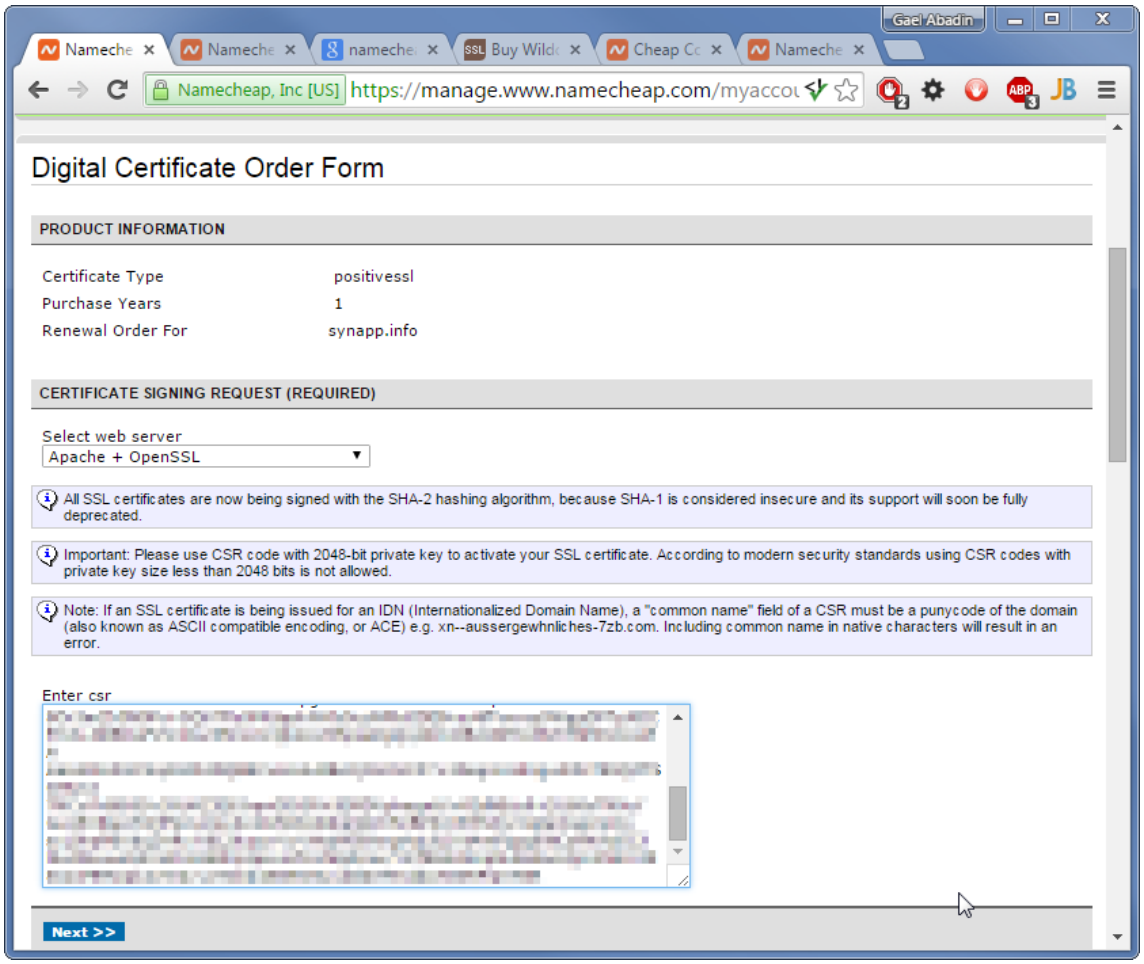


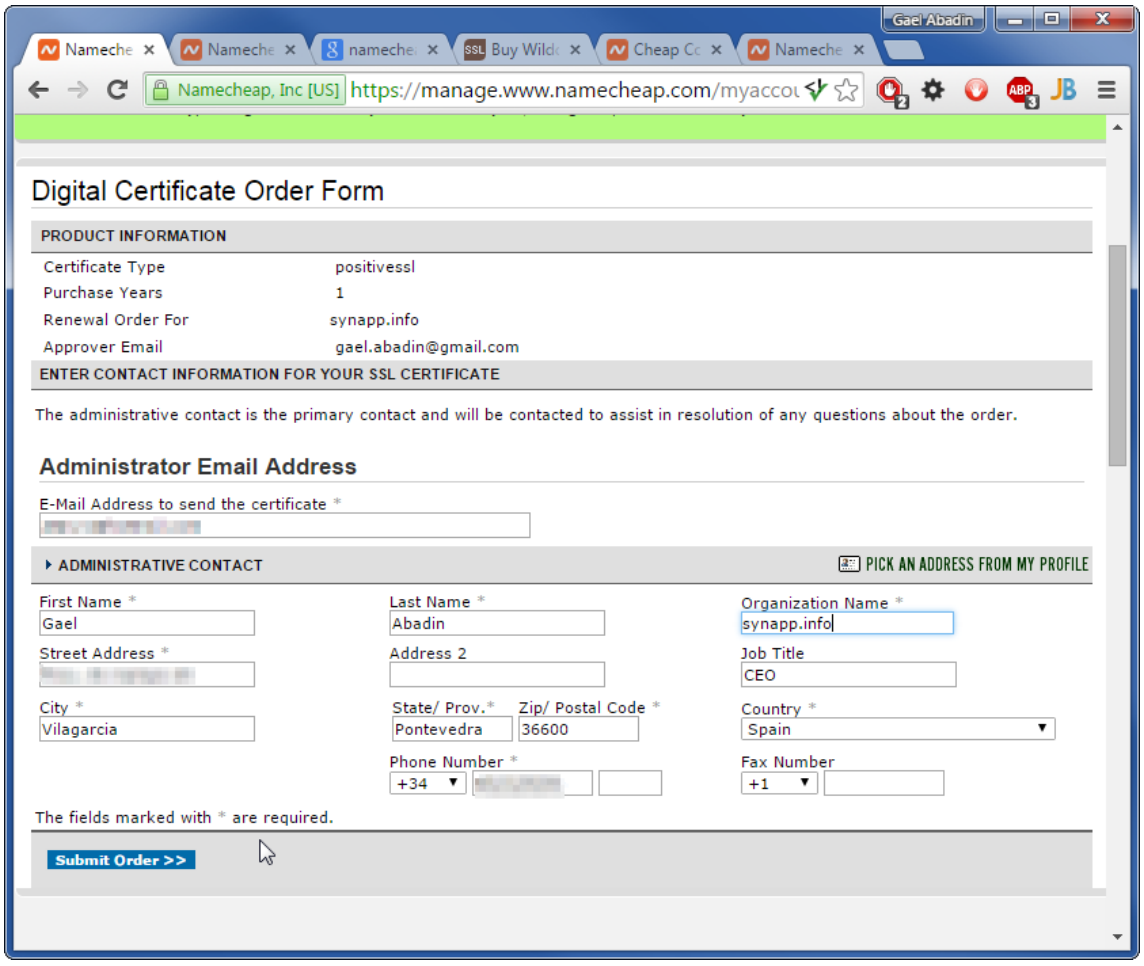
Next thing you have to do is to give them your certificate signing request code which you will generate along with your certificate key using openssl, like you are seeing on this screen caption.

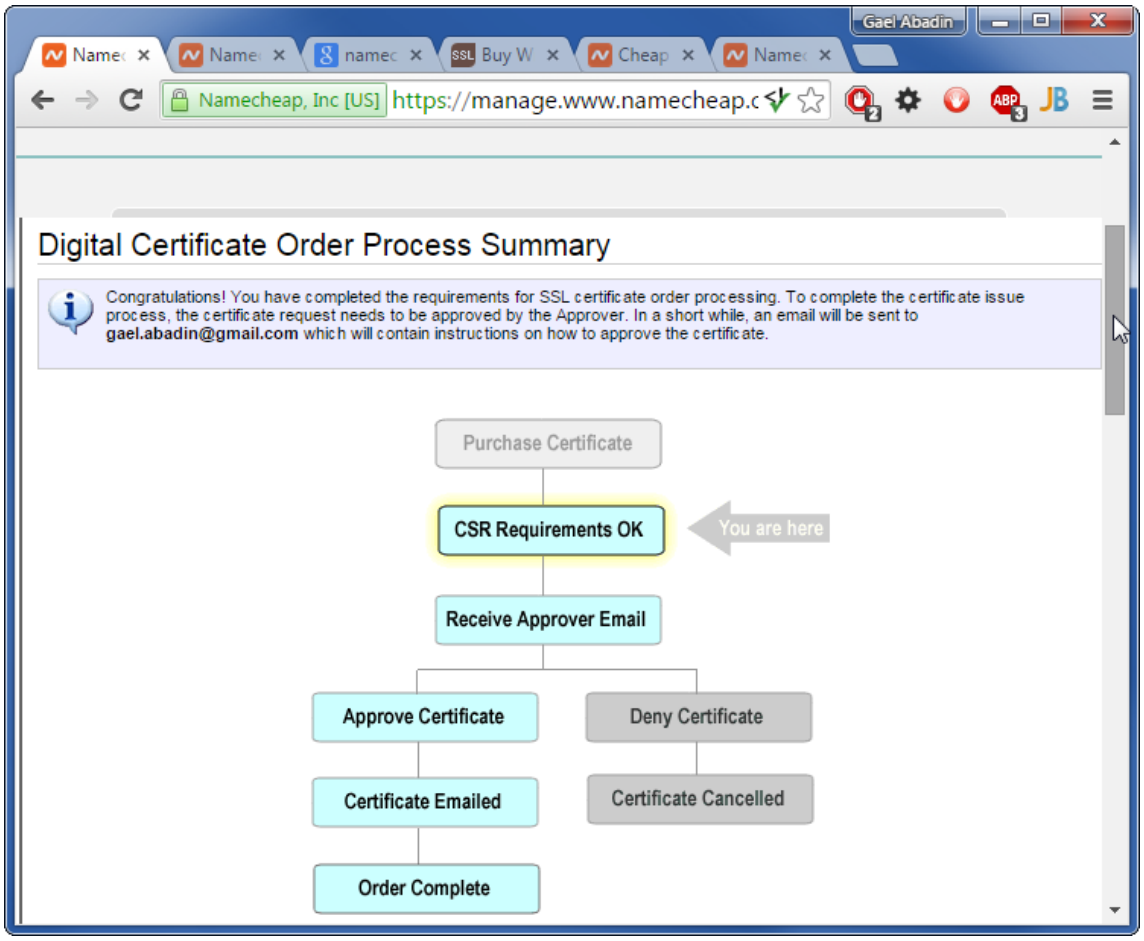


Depending on your provider the CSR form access and details may change, but you will be looking at something like this.

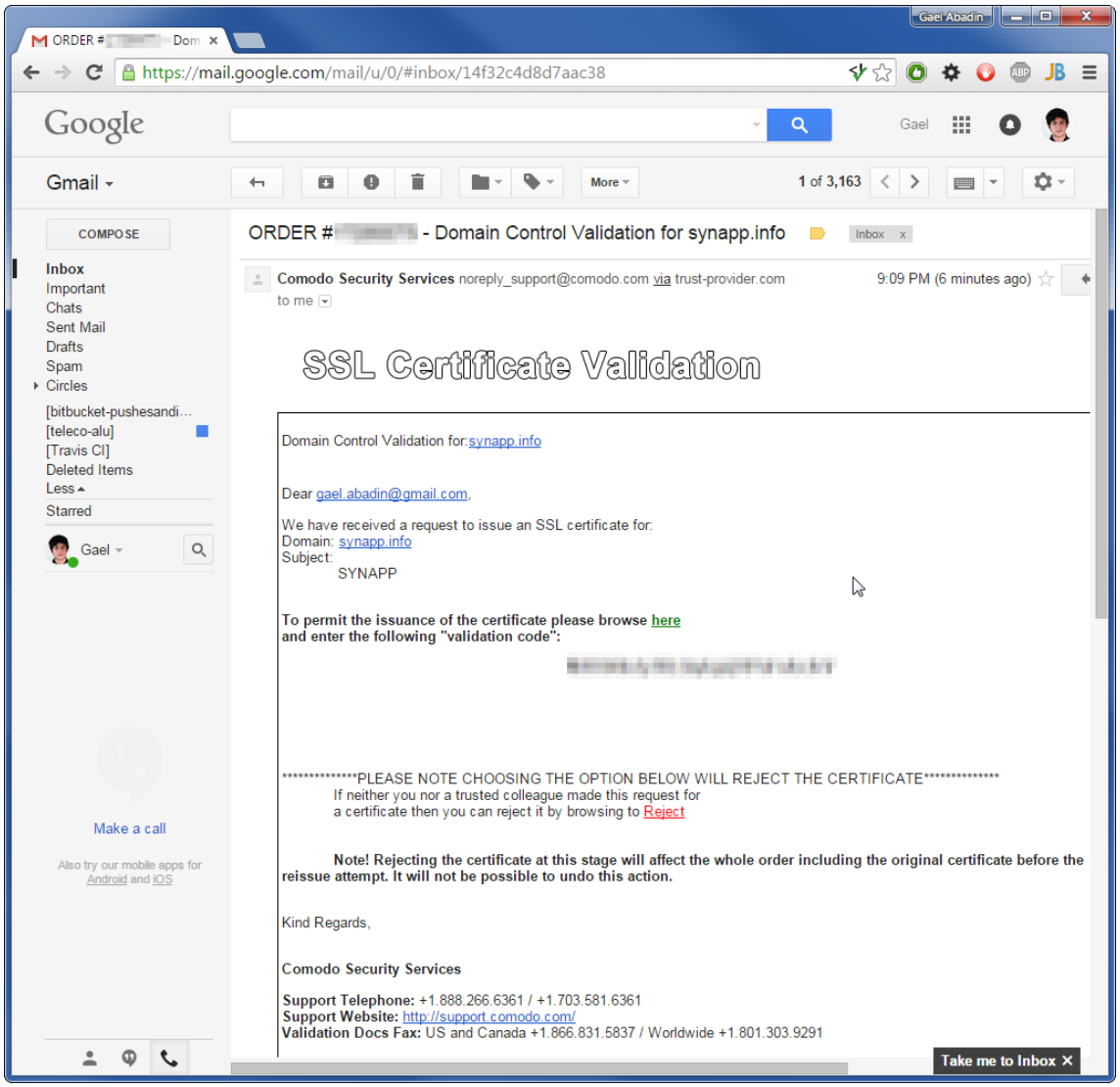



Gael Abadin
Namecheap, Inc [US] https://manage.www.namecheap.com/myaccou
Digital Certificate Order Form
PRODUCT INFORMATION
Certificate Type positivessl
Purchase Years 1
Renewal Order For synapp.info
CERTIFICATE SIGNING REQUEST (REQUIRED)
Select web server
Apache + OpenSSL
All SSL certificates are now being signed with the SHA-2 hashing algorithm, because SHA-1 is considered insecure and its support will soon be fully deprecated.
Important: Please use CSR code with 2048-bit private key to activate your SSL certificate. According to modern security standards using CSR codes with private key size less than 2048 bits is not allowed.
Note: If an SSL certificate is being issued for an IDN (Internationalized Domain Name), a "common name" field of a CSR must be a punycode of the domain (also known as ASCII compatible encoding, or ACE) e.g. xn--aussergewhnliches-7zb.com. Including common name in native characters will result in an error.
Enter csr
Next >>

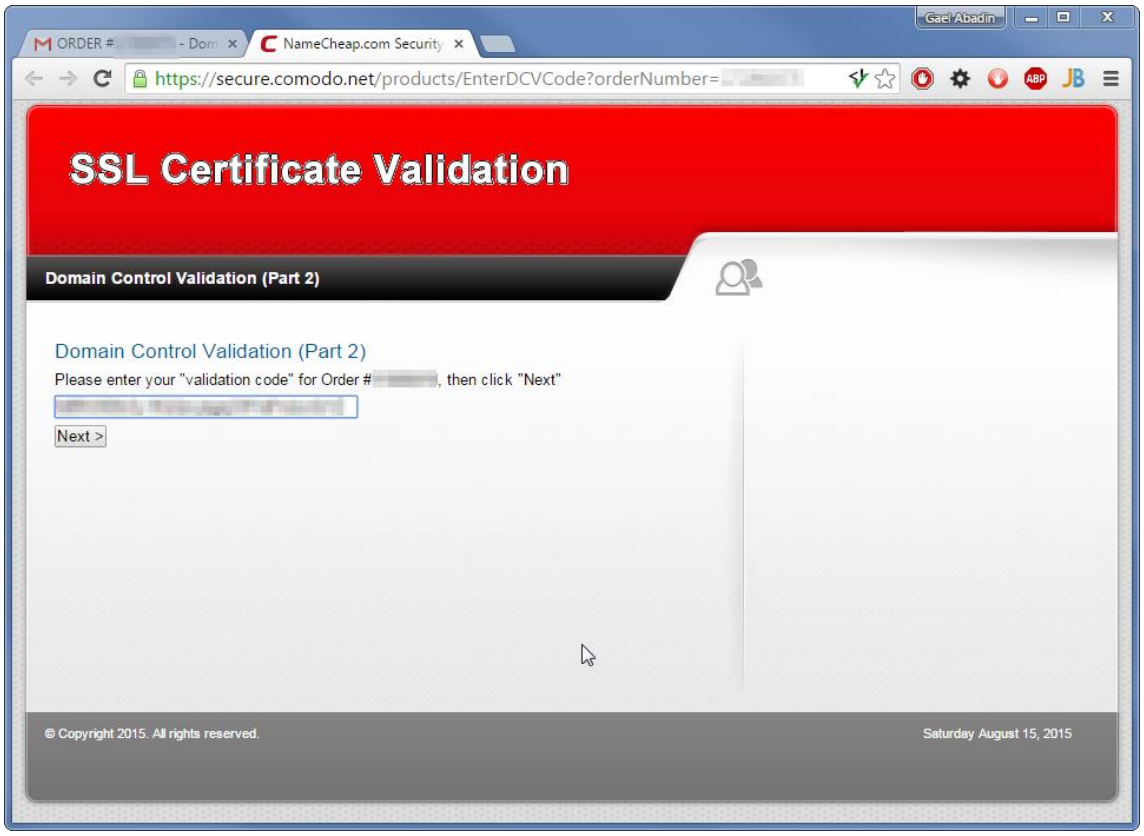

Gael Abadin
Namecheap, Inc [US]
https://manage.www.namecheap.com/myaccou
Digital Certificate Order Form
PRODUCT INFORMATION
Certificate Type positivessl
Purchase Years 1
Renewal Order For synapp.info
Approver Email gael.abadin@gmail.com
ENTER CONTACT INFORMATION FOR YOUR SSL CERTIFICATE
The administrative contact is the primary contact and will be contacted to assist in resolution of any questions about the order.
Administrator Email Address
E-Mail Address to send the certificate *
ADMINISTRATIVE CONTACT
PICK AN ADDRESS FROM MY PROFILE
First Name * Gael
Last Name * Abadin
Organization Name * synapp.info
Street Address *
Address 2
Job Title CEO
City * Vilagarcia
State/ Prov.* Pontevedra
Zip/ Postal Code * 36600
Country * Spain
Phone Number * +34
Fax Number +1
The fields marked with * are required.
Submit Order >>

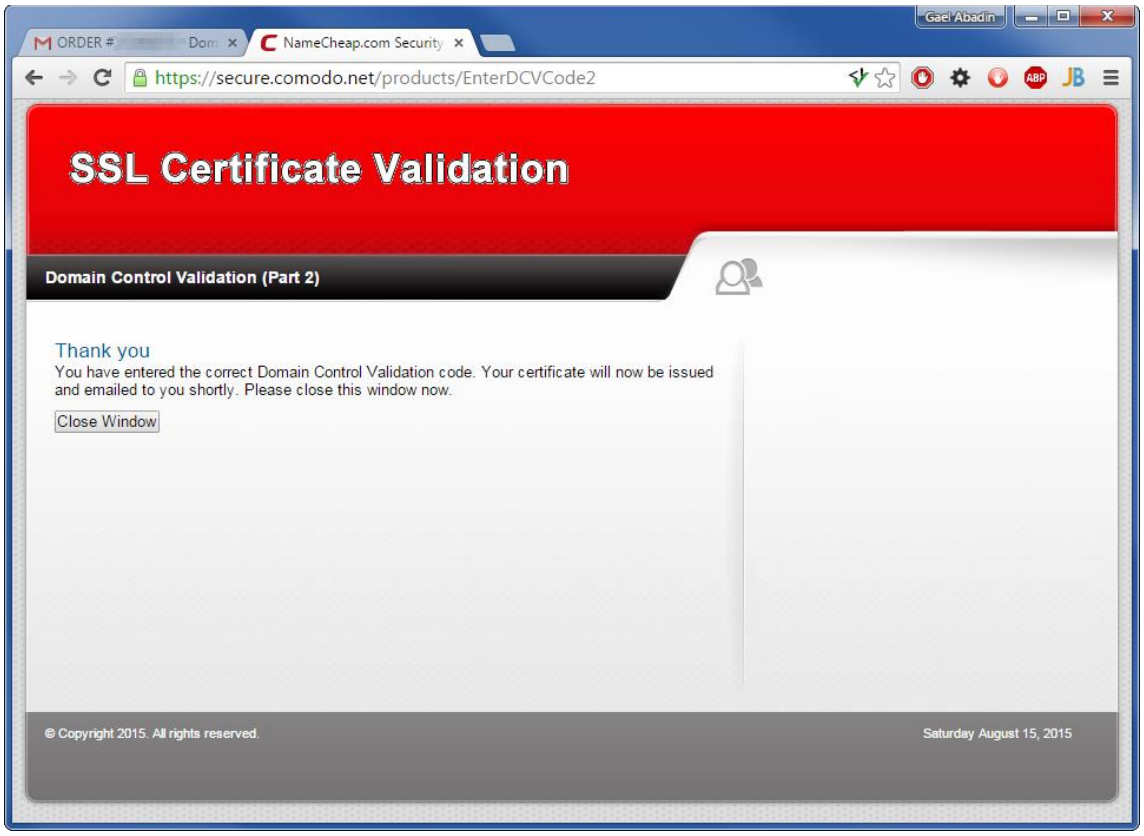


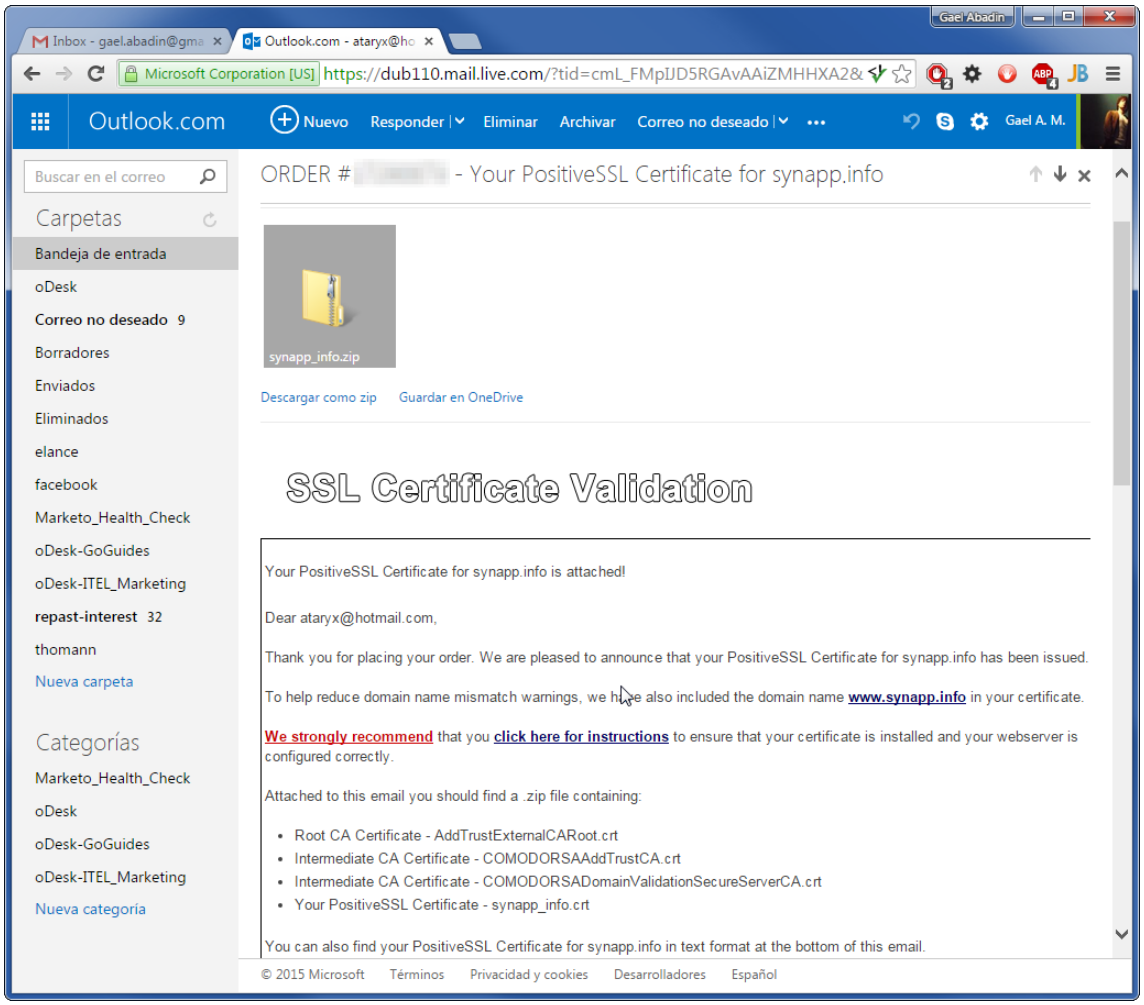
Once you fill and send the form you will receive a confirmation code on your email and a link to generate your certificate.



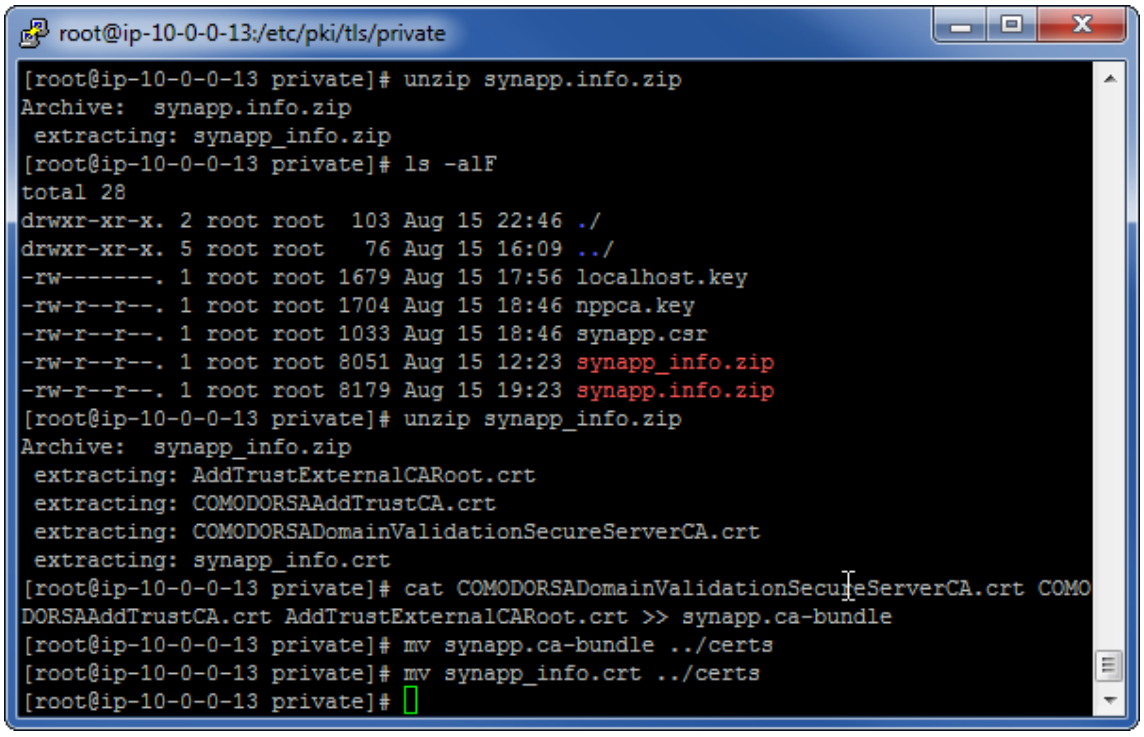
Follow that link to complete the process and the certificate will be sent to your email address.


Gael Abadin
ORDER #
NameCheap.com Security
https://secure.comodo.net/products/EnterDCVCode?orderNumber=
SSL Certificate Validation
Domain Control Validation (Part 2)
Domain Control Validation (Part 2)
Please enter your "validation code" for Order #, then click "Next"
Next >
© Copyright 2015. All rights reserved.
Saturday August 15, 2015


Gael Abadin
ORDER #
NameCheap.com Security
https://secure.comodo.net/products/EnterDCVCode2
SSL Certificate Validation
Domain Control Validation (Part 2)
Thank you
You have entered the correct Domain Control Validation code. Your certificate will now be issued and emailed to you shortly. Please close this window now.
Close Window
© Copyright 2015. All rights reserved.
Saturday August 15, 2015



Now upload the contents of the certificate bundle to your server's /etc/pki/tls/certs and put them together by concatenating them in the order shown in the caption (from highest to lowest hierarchy), as shown on the screen.



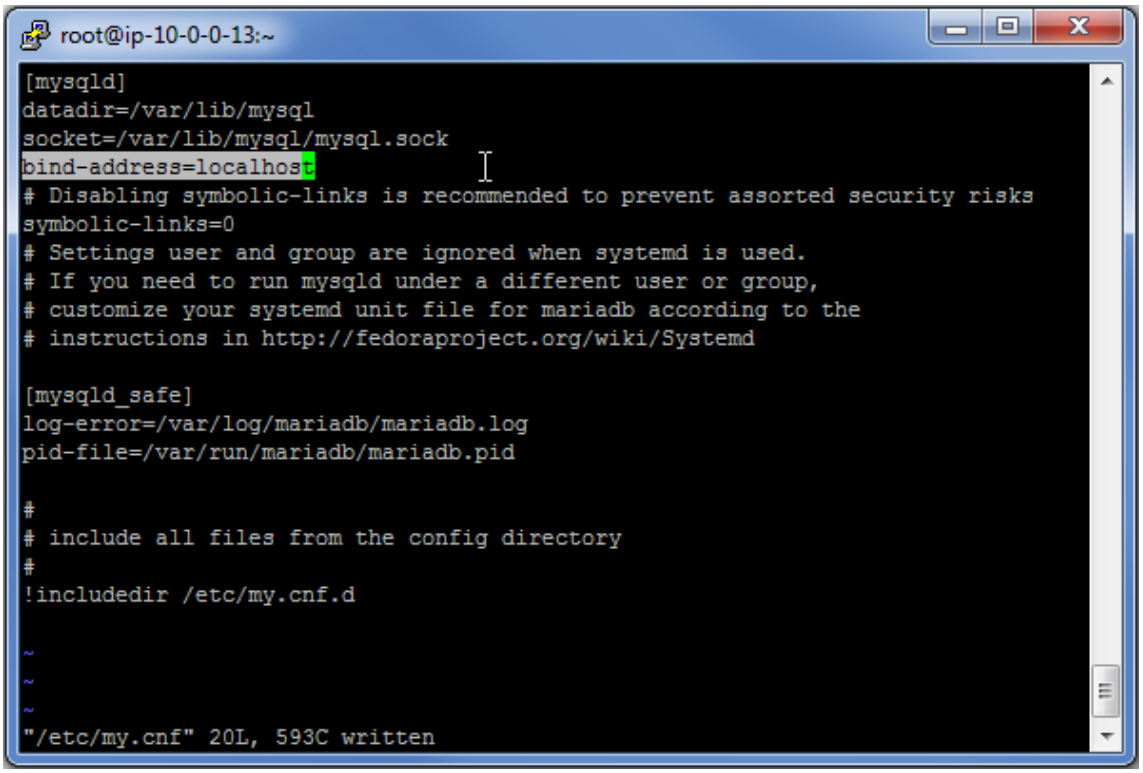
And that's it! You have a signed certificate bundle ready to be used with Apache Web Server or any other web server with SSL support. An important reminder: The certificate bundle is public and your server will be passing it along to any connected client. The private key on the other hand is super secret stuff and you should be very careful who you give access to it, which ideally should be nobody.

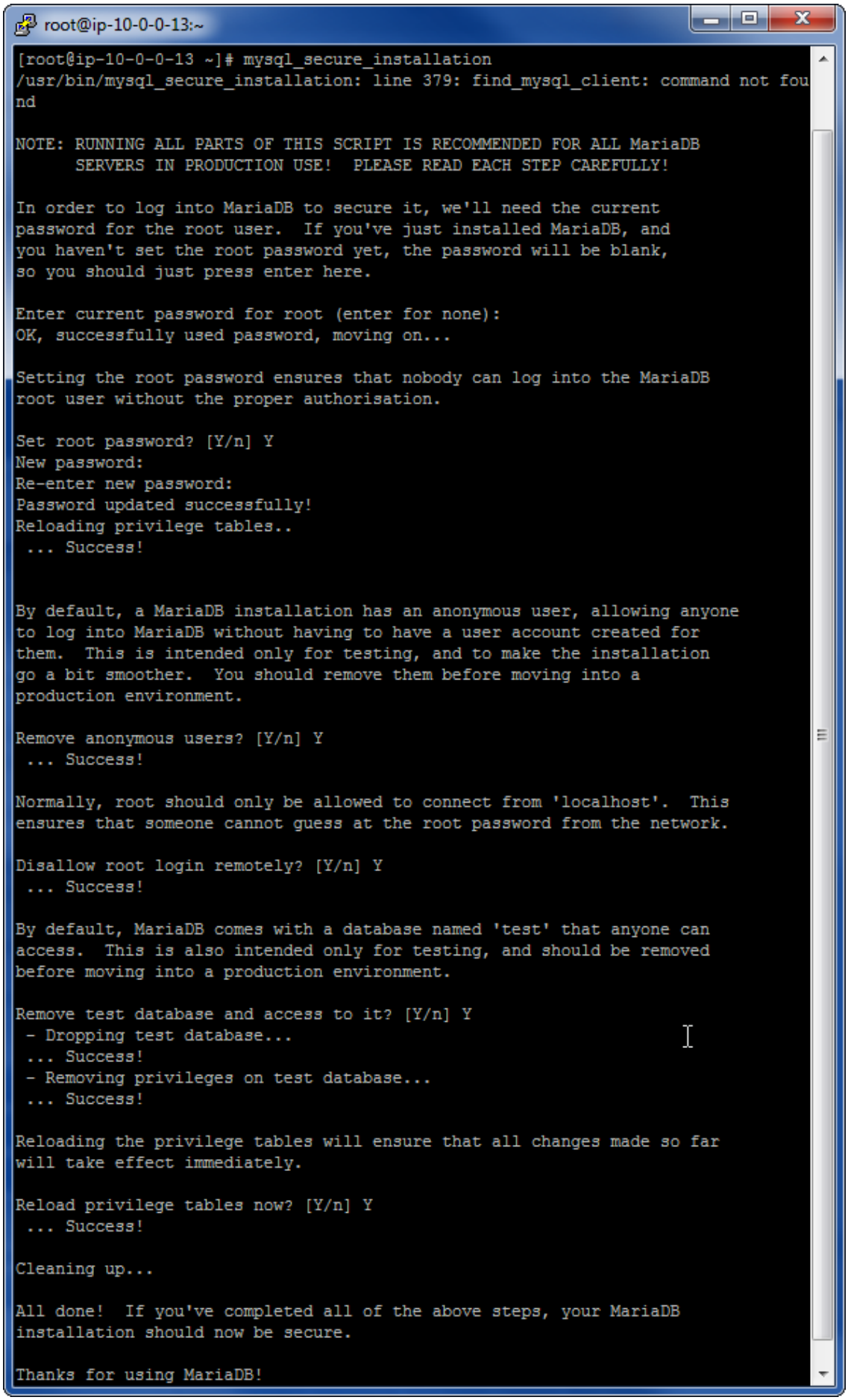
On the next part I will be configuring apache's SSL module, using the certificate and key generated on this part, which will take like 2 seconds, plus all the other things that need to be configured before deploying our application, which will take a little longer, but not much longer. Meet me there if you want to know more.

4/5: CONFIGURING A CENTOS 7 LAMP STACK

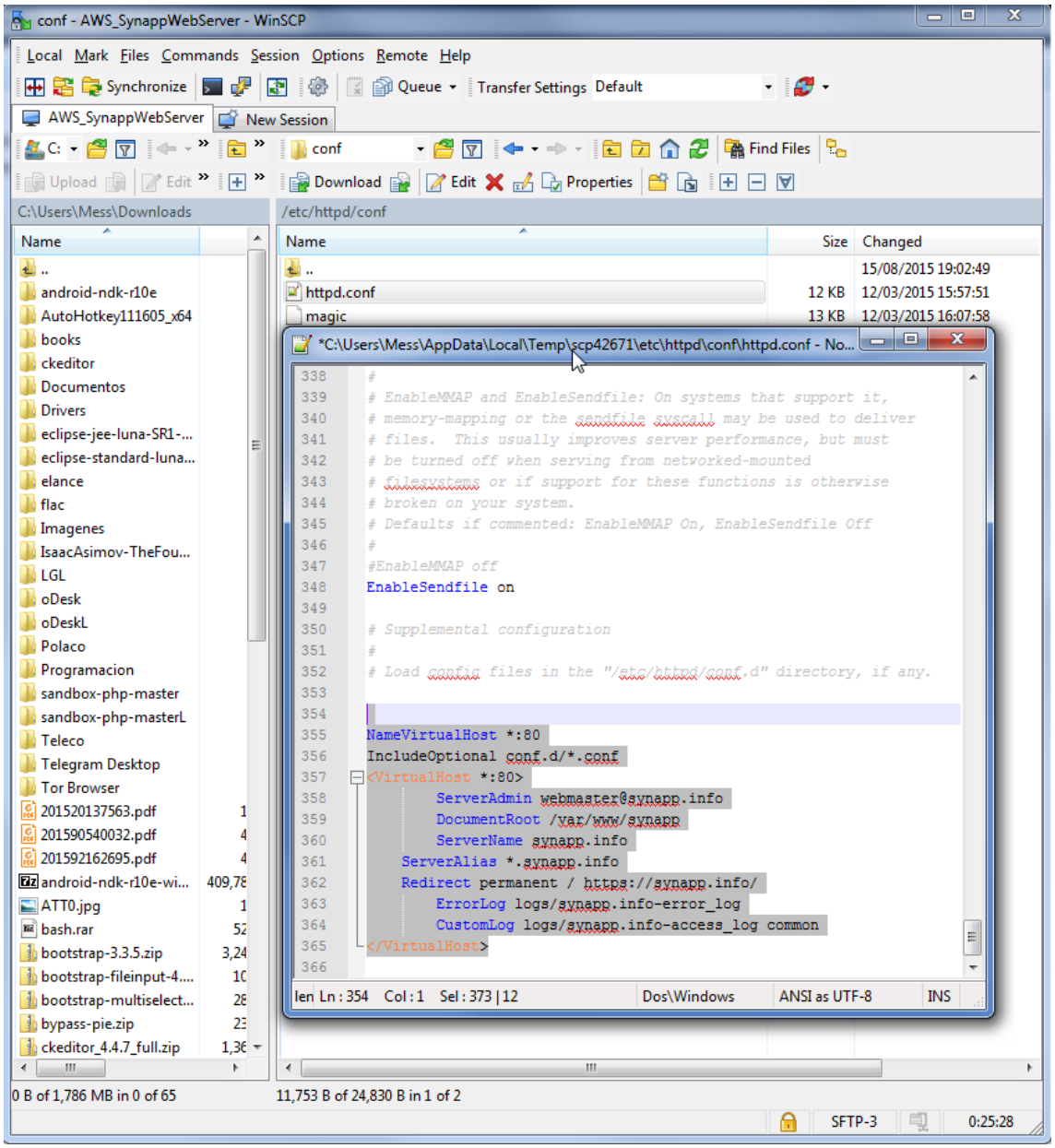
This is the fourth part out of a five part series on how to deploy a PHP Web Application on an Apache Web server on a CentOS 7 machine on Amazon Elastic Compute Cloud. If you didn't get lost and ended up here by chance, by the end of this part you will know how to set up your web server, PHP and MySQL services on your production machine to bring joy to thousands of users with an infrastructure that costs you less than a sandwich per month.

I will start by showing how to edit /etc/my.cnf file to bind the mariadb daemon to localhost address so you can connect to it from the local machine using a TCP client. After that I suggest you run mysql_secure_installation script like shown on the screen to perform some security checks and configuration tasks.

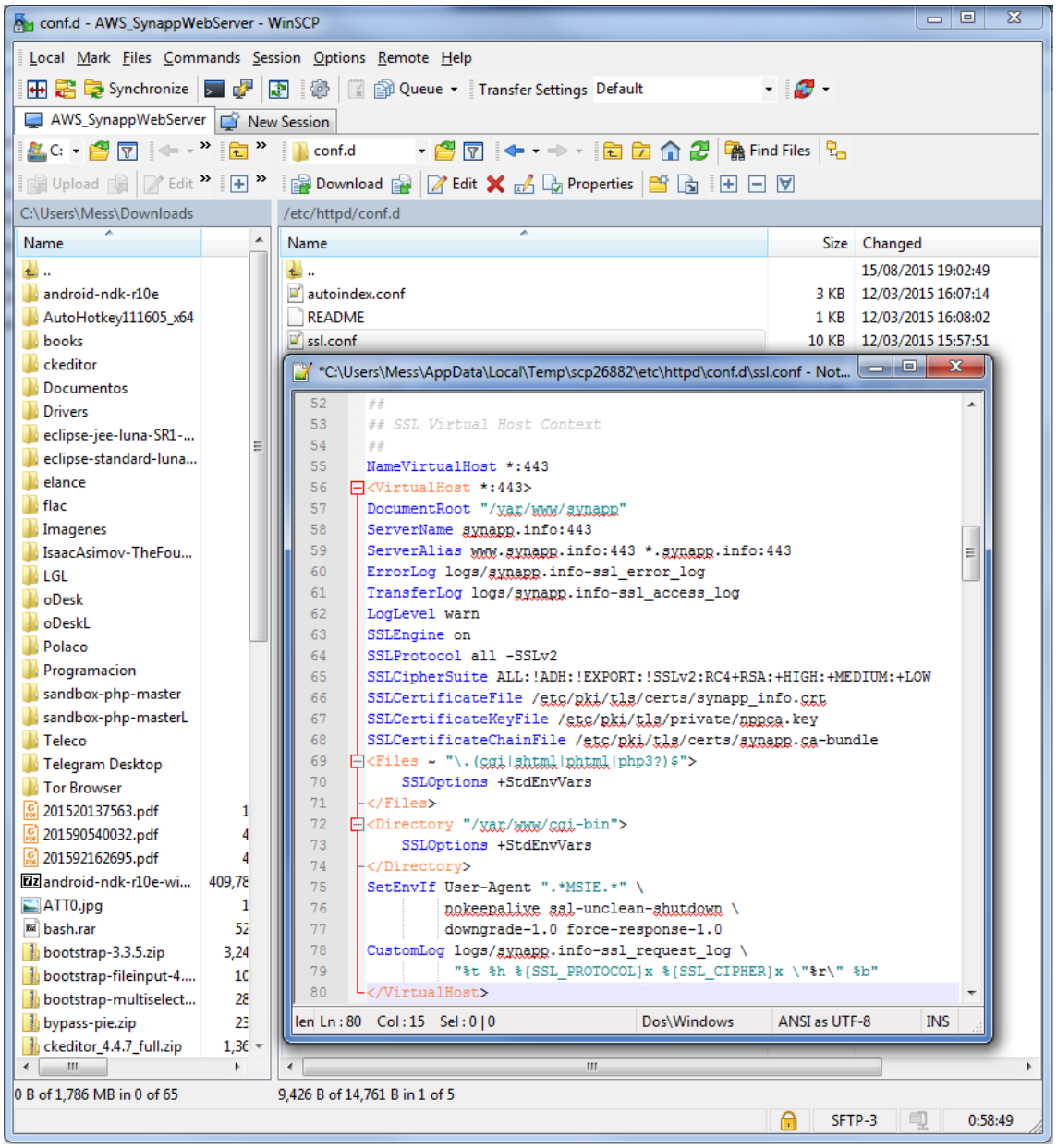

root@ip-10-0-0-13:~
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
bind-address=localhost
Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
Settings user and group are ignored when systemd is used.
If you need to run mysqld under a different user or group,
customize your systemd unit file for mariadb according to the
instructions in http://fedoraproject.org/wiki/Systemd
[mysqld_safe]
log-error=/var/log/mariadb/mariadb.log
pid-file=/var/run/mariadb/mariadb.pid
#
include all files from the config directory
#
!includedir /etc/my.cnf.d
~
~
~
"/etc/my.cnf" 20L, 593C written


root@ip-10-0-0-13:~
[root@ip-10-0-0-13 ~]# mysql_secure_installation
/usr/bin/mysql_secure_installation: line 379: find_mysql_client: command not fou
nd
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.
Enter current password for root (enter for none):
OK, successfully used password, moving on...
Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.
Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.
Remove anonymous users? [Y/n] Y
... Success!
Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.
Disallow root login remotely? [Y/n] Y
... Success!
By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.
Remove test database and access to it? [Y/n] Y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.
Reload privilege tables now? [Y/n] Y
... Success!
Cleaning up...
All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.
Thanks for using MariaDB!

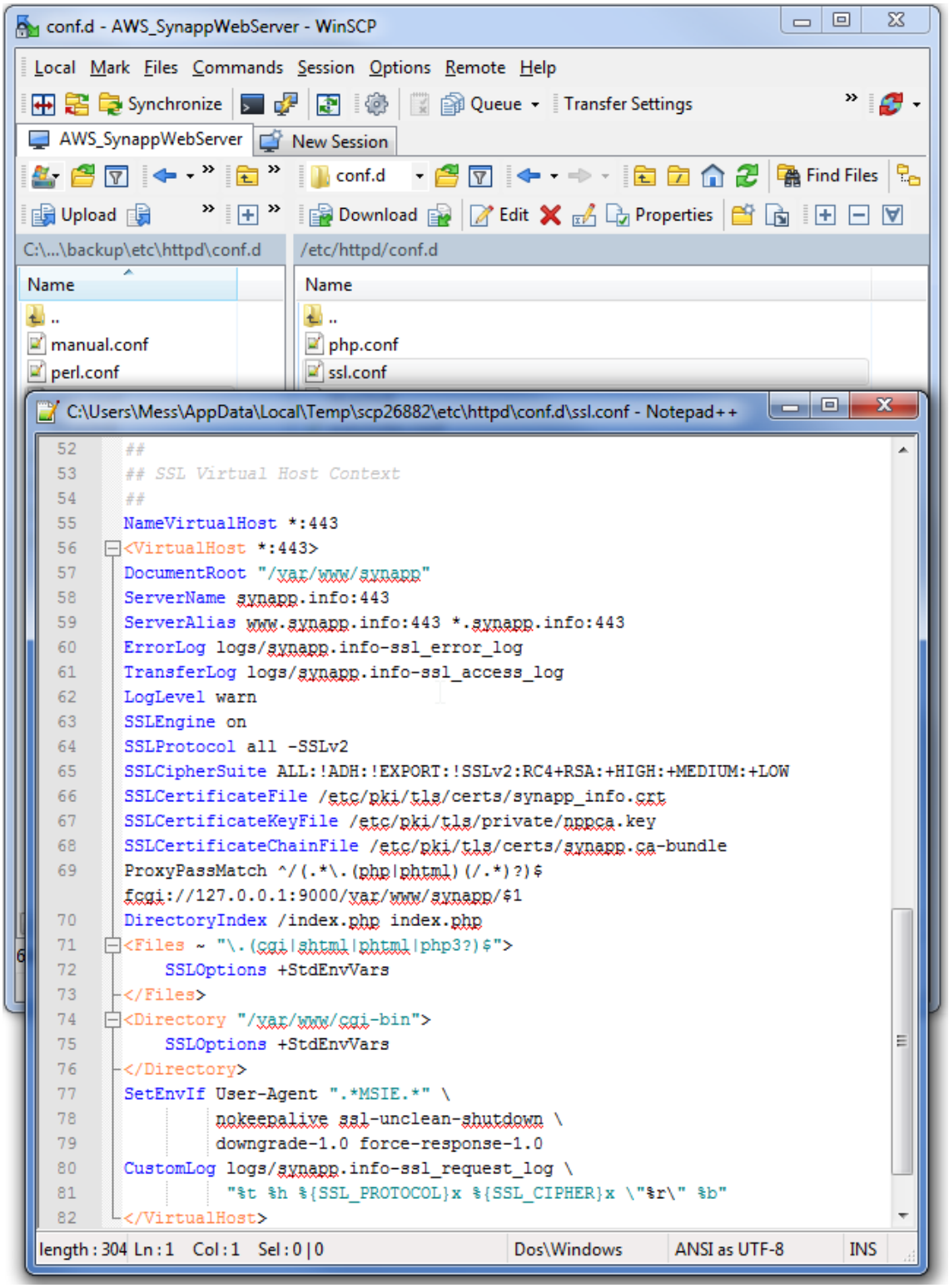
Now we will set up the SSL redirection on the httpd.conf file as shown by the highlighted text on this screenshot, so our clients are pointed to a secure connection when they try to reach us through a plain HTTP connection on port 80.



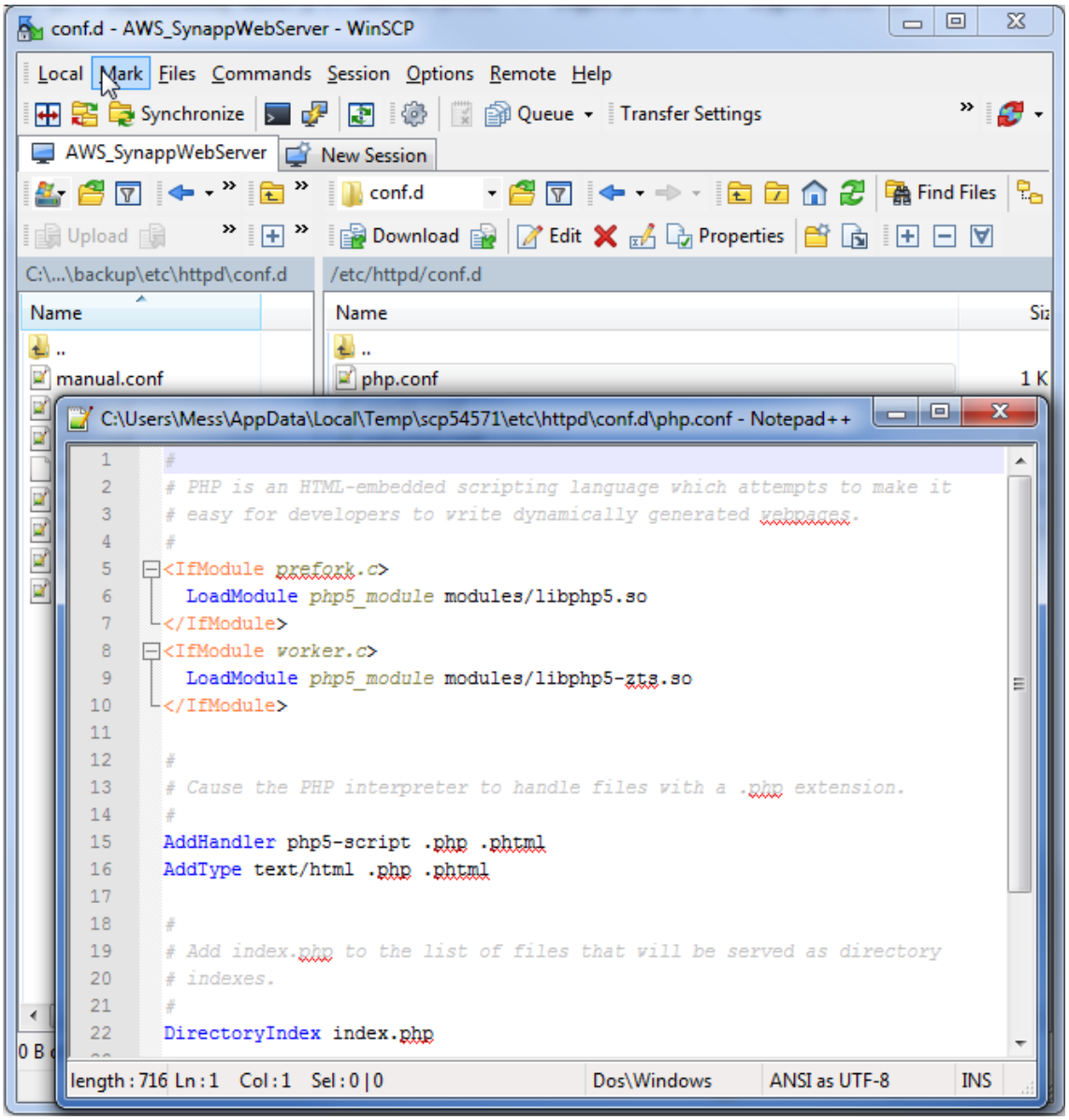
Continuing with the web server set up, this is how your SSL virtual host configuration should look like if you don't have fpm enabled.



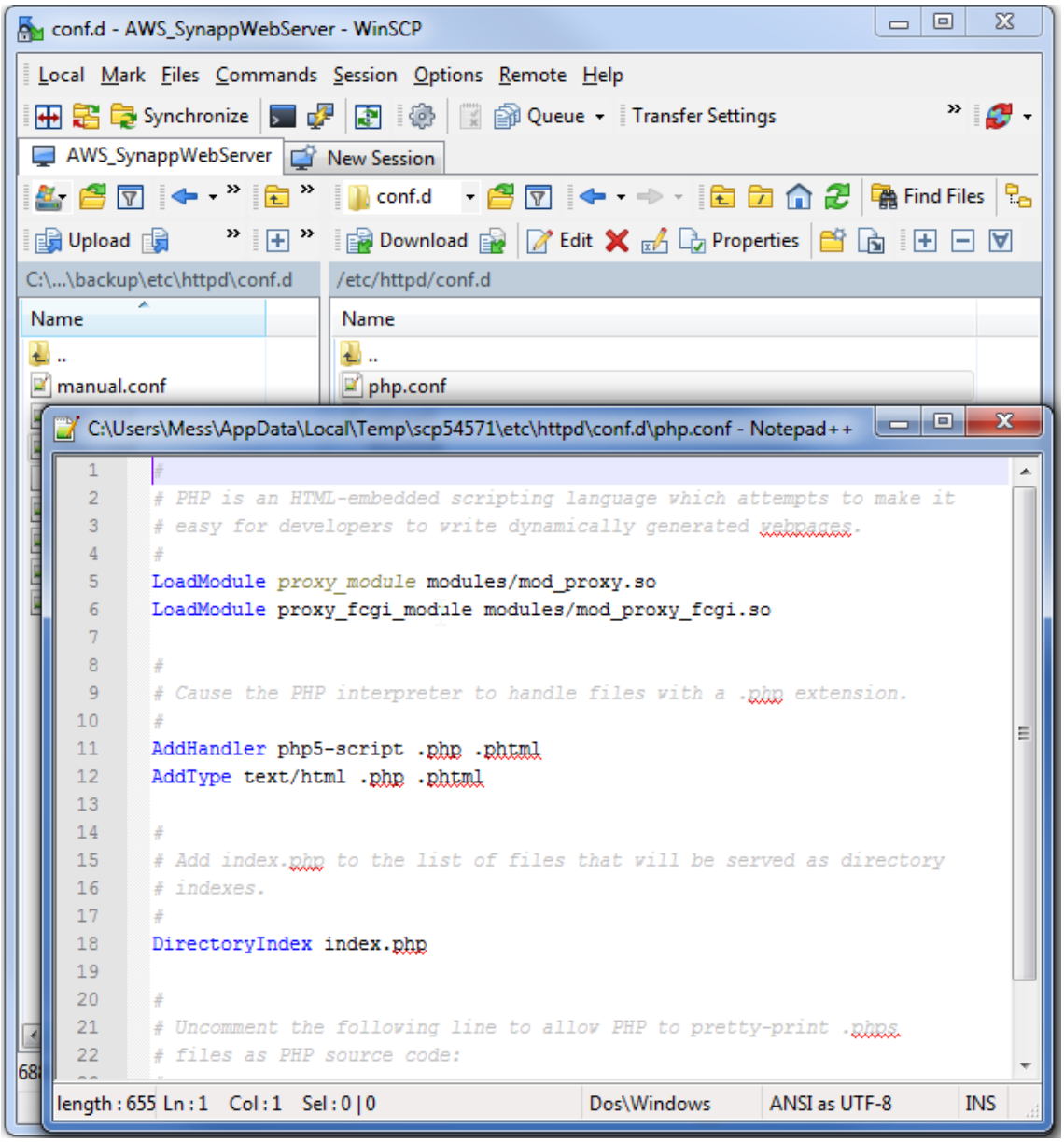
If you do (which you should if you were following this tutorial). It will look like this.



The same goes for the CGI module on php.conf Apache configuration file. This is without fpm.

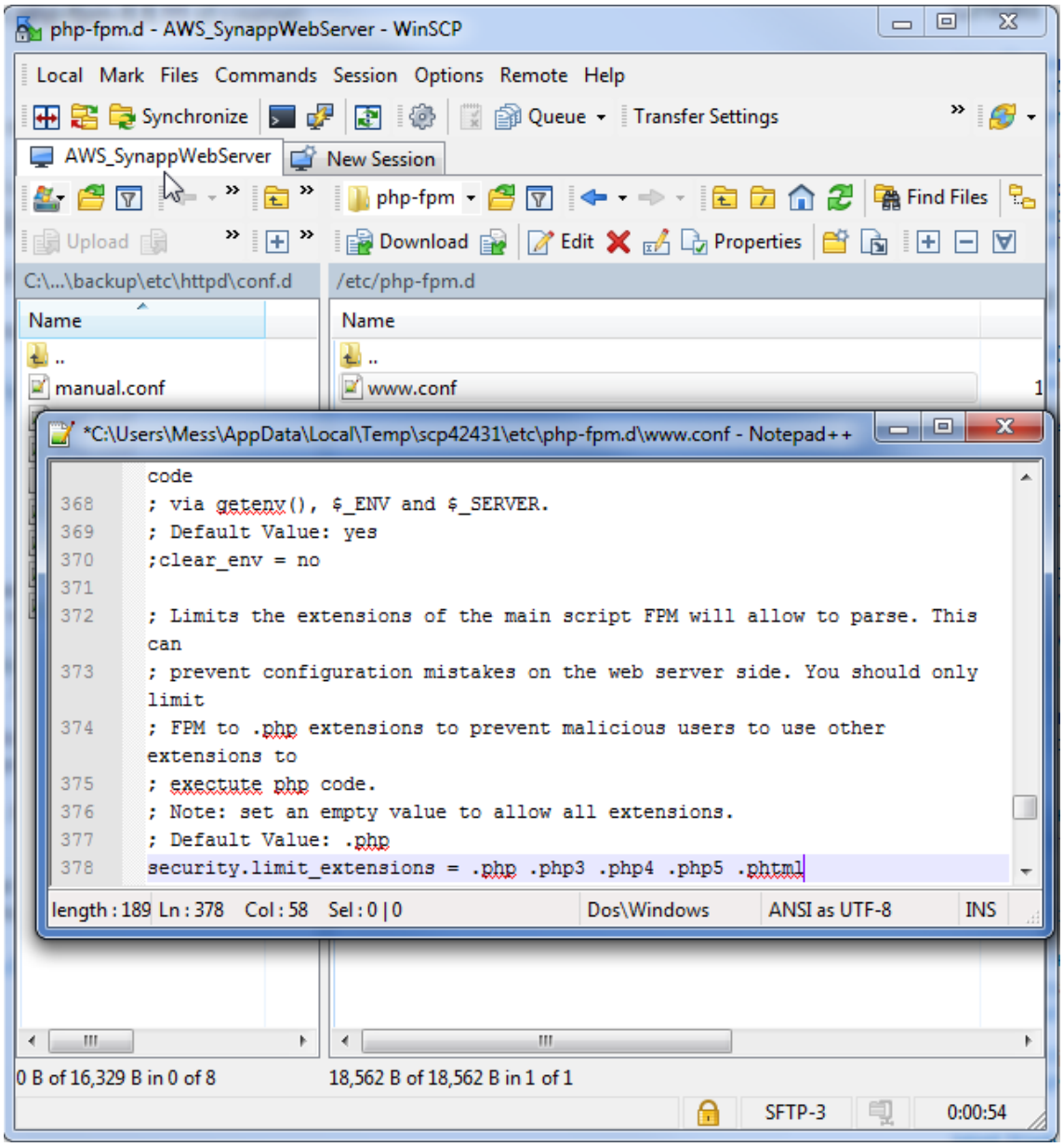


And this is with fpm.

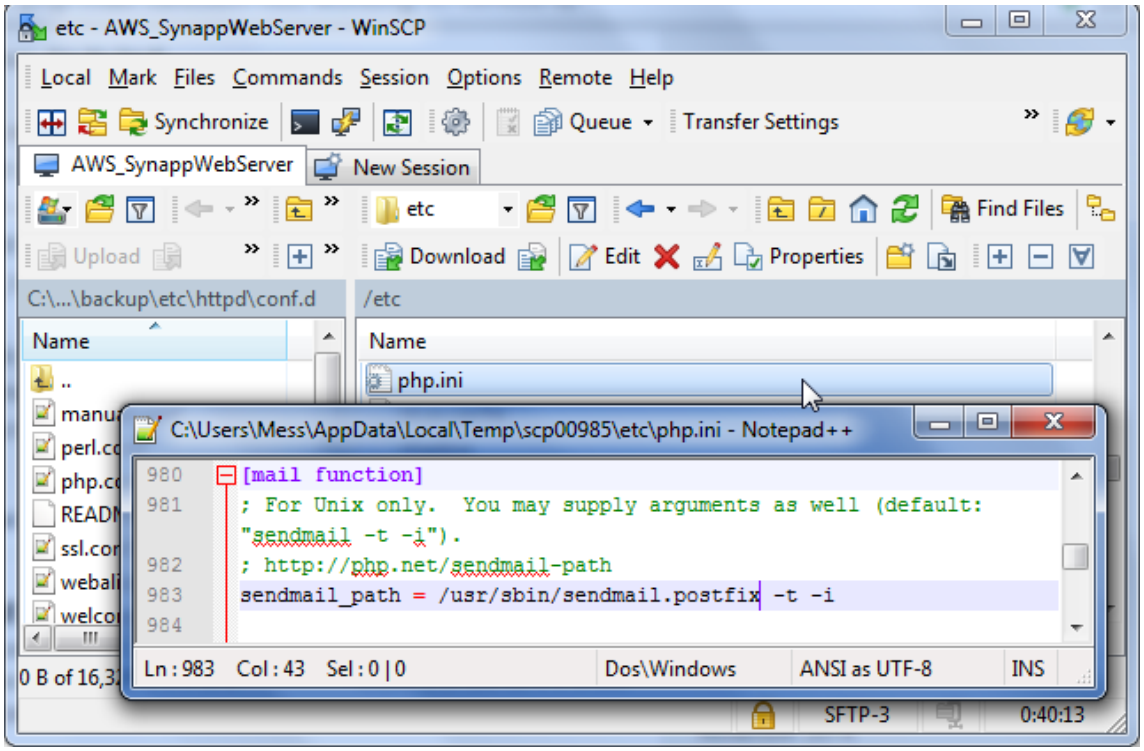


We move on now to the PHP configuration itself. It is only a couple of lines on a couple of files you must edit to get it to work:

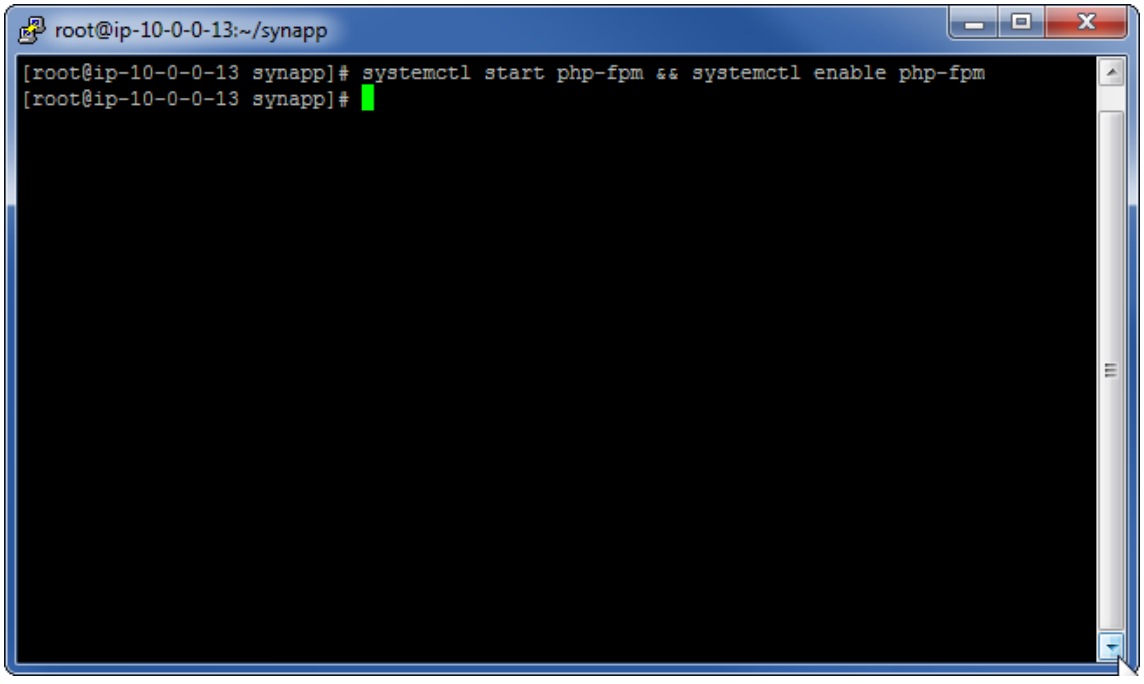
- First, configure the allowed extensions on the etc/php-fpm.d/www.conf file.



- Now, edit the sendmail path on the /etc/php.ini PHP configuration file to make it work with postfix, which is the default MTA we installed in the second part of this series.



What you can see right now above this paragraph is an example showing how to enable a PHP module. Most of them come enabled by default, gnupg, which I built and installed from PECL repo on the second part of this series, does not, but as you can see it is very easy to activate it if you require it. Just don't forget to restart the php-fpm daemon after doing it.



That's the end of it. Now you can safely deploy pretty much every PHP web application there is around on your system, including the most popular CMSs such as WordPress, Drupal or Joomla. I kindly invite you to follow me to the next and last part of this tutorial if you want to see a demo on how to deploy the latest release of my very own project using git and

composer, plus a little bonus on how your DNS zone file should look like if you want to link an Internet domain to your machine.

5/5: DEPLOYING A LAMP WEB APP AND SETTING THE DNS RECORDS

This is the last part of this five part tutorial on how to set up a LAMP stack on AWS EC2 and deploy an PHP web application with SSL support on it.

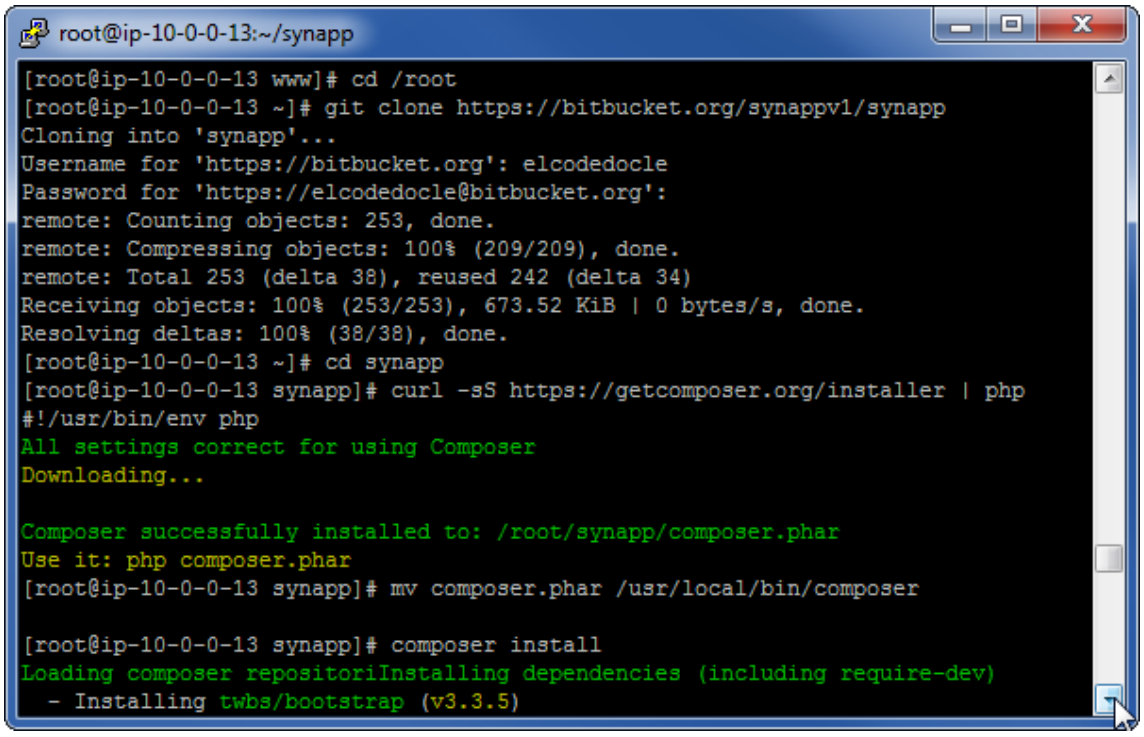
Up until now, I have been giving you vendor specific instructions on how to set up a free and open source system that can run pretty much any PHP web application on the market.

On this part I will show you how to set up a generic web application hosted on a git repository using composer dependencies; I will also show you a sample of a DNS zone file linking a domain to it so it can be easily accessed.

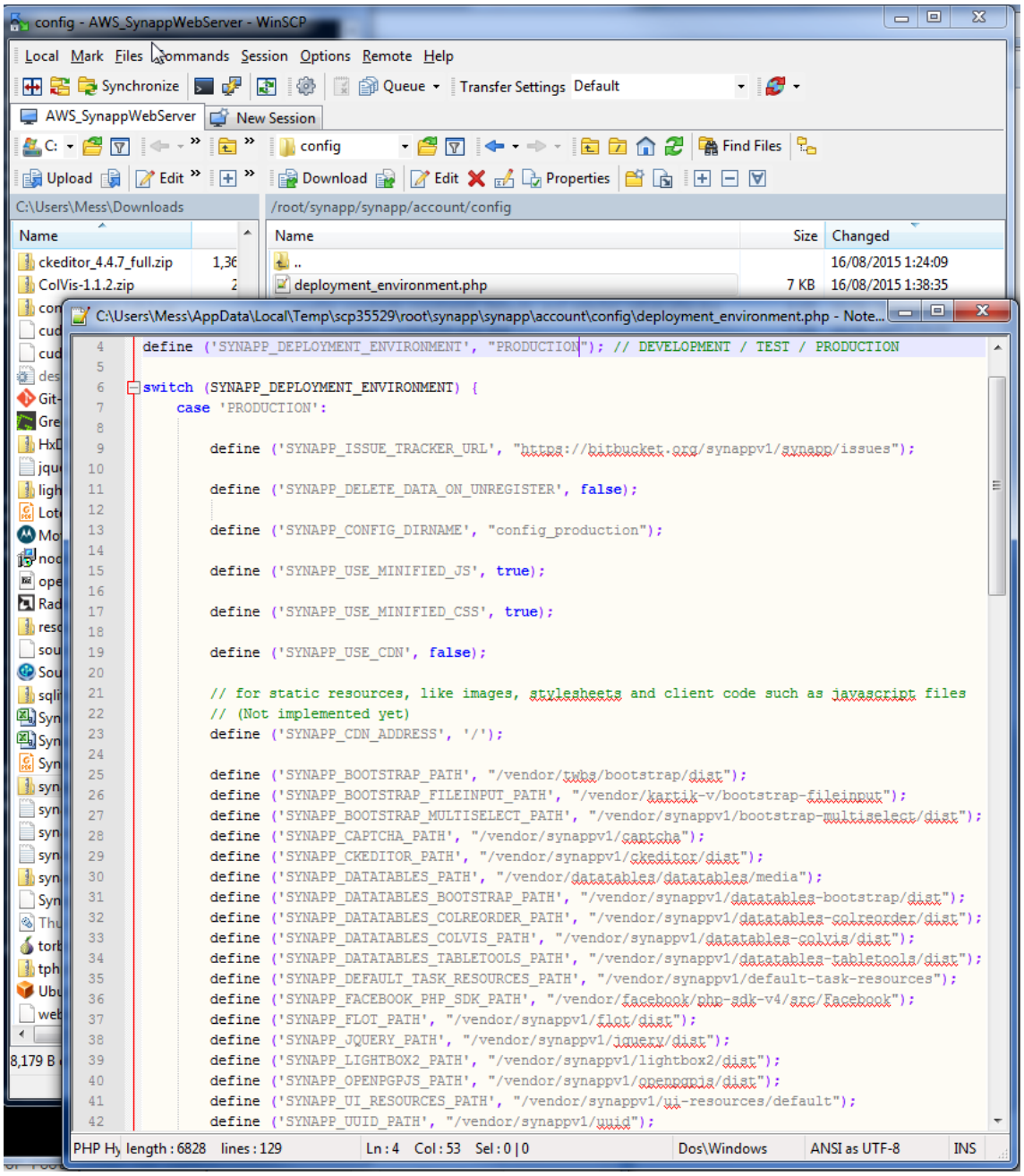
I will be deploying SynAPP, my own master thesis project which is currently online at synapp.info. Chances are you are not very interested on deploying this particular project, but I will be giving only but a few specific details about it which you can ignore to focus on the general aspects of deploying a web app if you wish. Those general aspects would be:

- Installing composer and building the app and its dependencies on a public web folder from its git repository;
- Importing the SQL DDL script containing the application database schema definition;
- Configuring the database connection settings and application paths and routes;
- And, when everything is ready, pointing the domain to the machine where the application is serving requests.

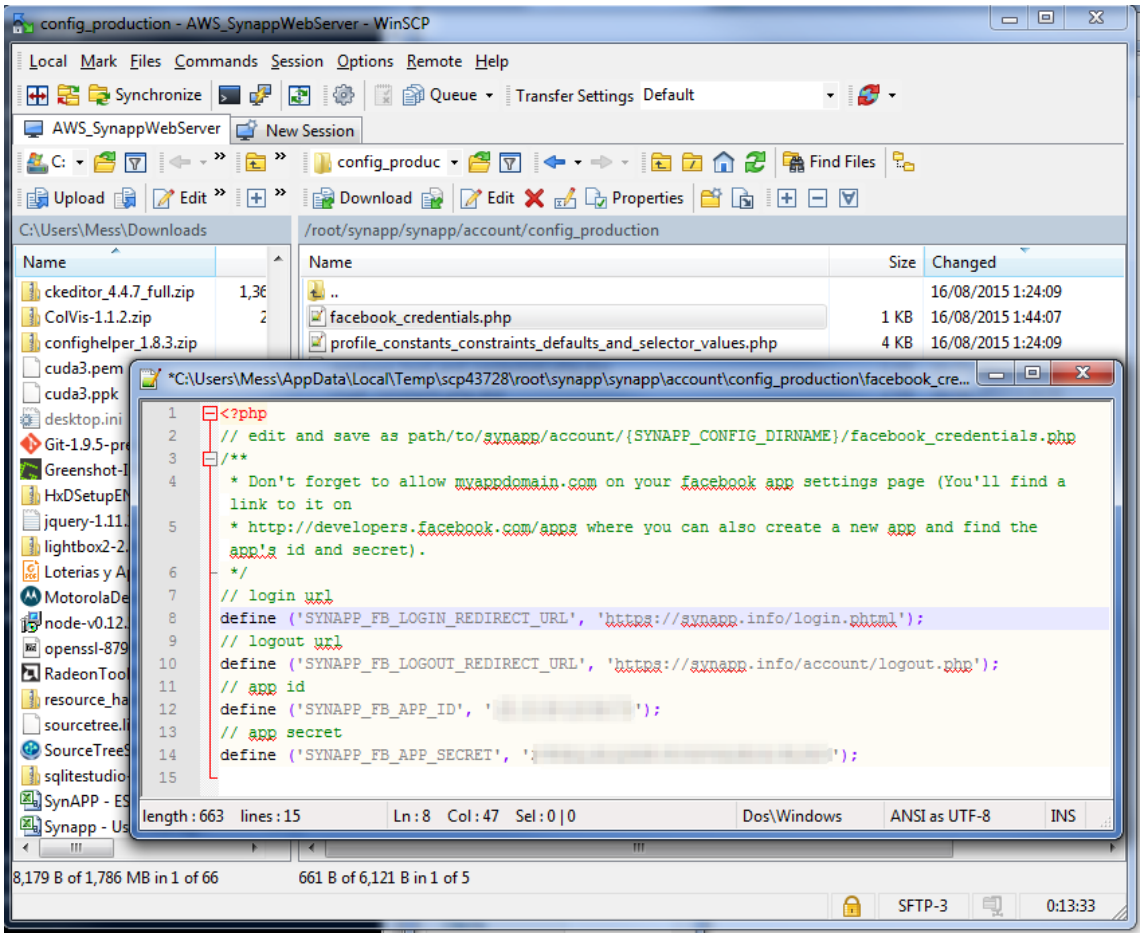
What you are seeing on the next screen capture are the commands required to complete the first step of the process: git cloning the repo; installing composer and running it to retrieve the app's dependencies. I am doing this as the root user in the root directory so when I finish setting up the application and move it to the public web folder all the file permissions and ownerships are set properly. That might not always be the case, so watch out for it or you could be leaving an exploitable attack vector on your server, like many Drupal users already know, unfortunately.



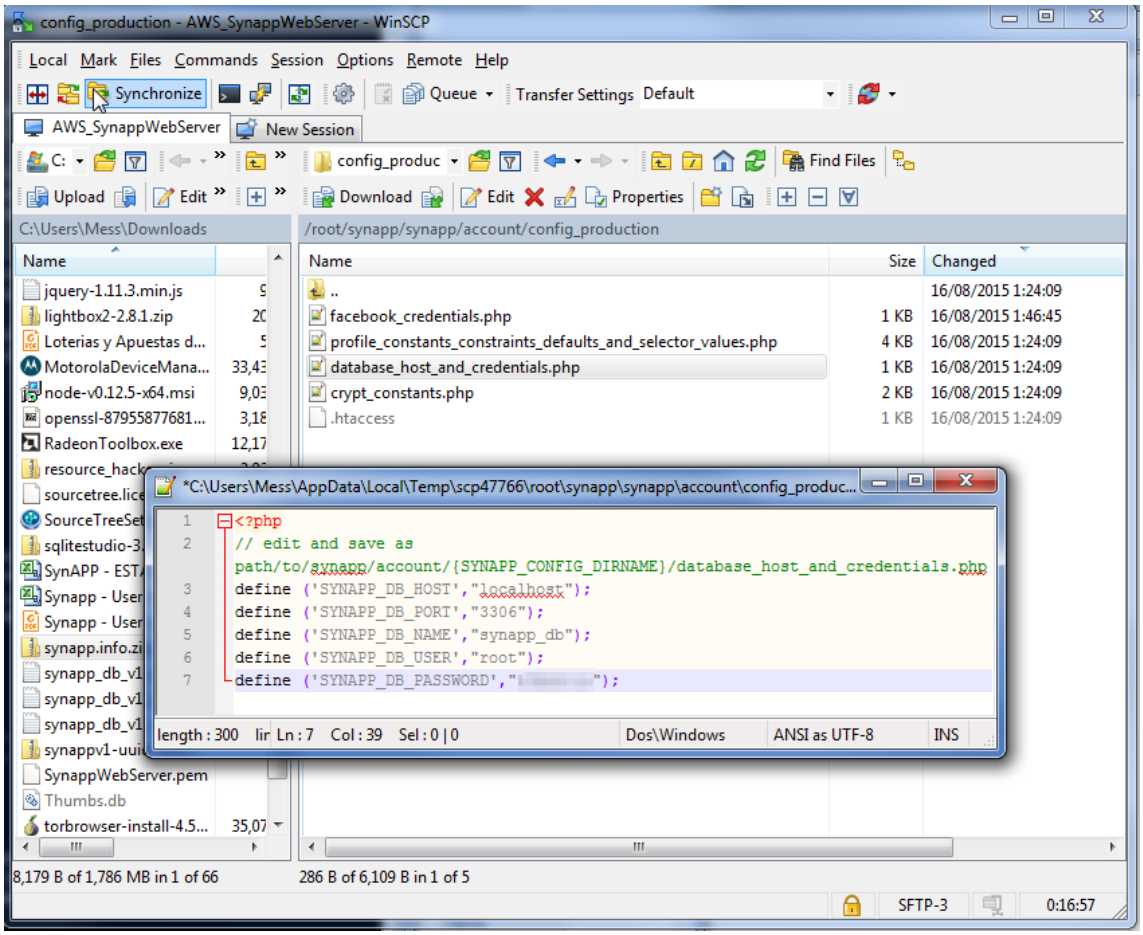
This is how the app's deployment script looks like when it's configured for production. The most important thing here is to set the proper 'SYNAPP_DEPLOYMENT_ENVIRONMENT' constant definition, which I have set to production, and inside the production environment definition the proper SYNAPP_CONFIG_DIRNAME constant definition, which points to the folder where the application expects to find the rest of the configuration files.



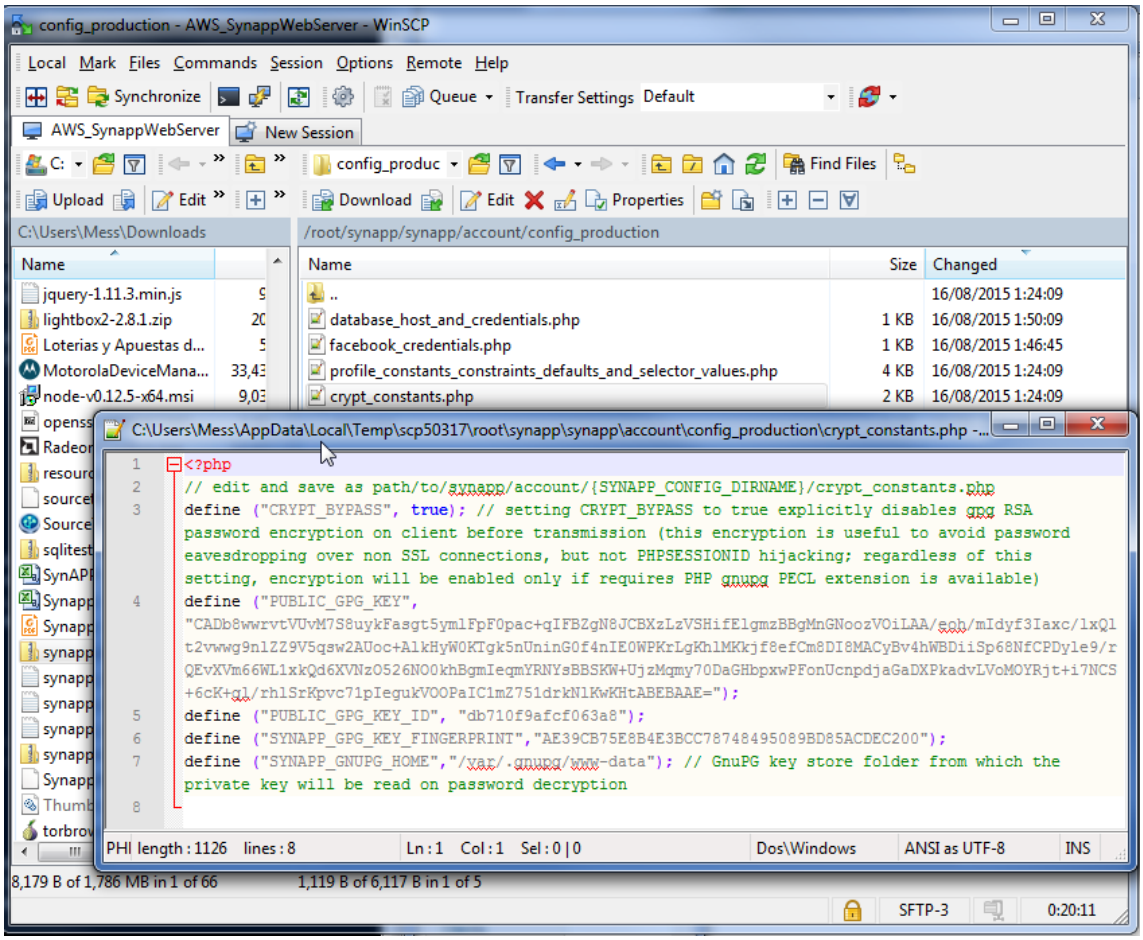
The first of those configuration files to be edited and placed on that folder is the facebook oauth credentials configuration file, which gives the application the parameters required to use facebook login to authenticate its users. There are four parameters to set: login and logout redirections which point to the app's respective processing scripts' URLs; and the application ID and secret which you can get when you register a new facebook app on facebook's developer console.



Then comes the database connection settings configuration. You can leave the default values and just edit the password by setting the same root password you chose when you executed the mysql_secure_installation script as indicated on the previous part of this tutorial. To leave a plaintext database root password around is not the best idea even if you are not going to deploy different web applications on the same database, so I suggest you to create a new database user with limited privileges and giving it access to the app database.



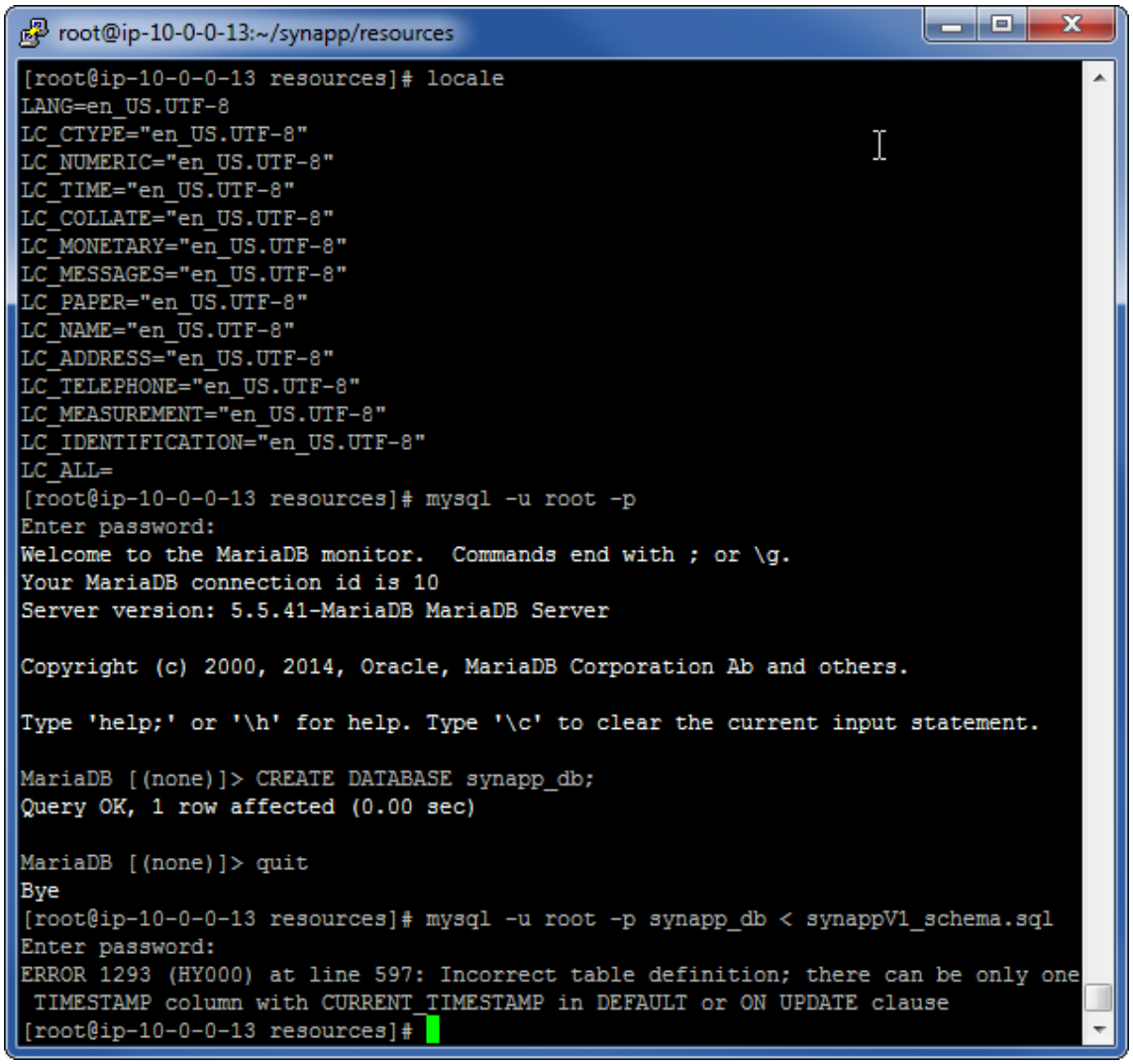
The last configuration file are gnupg parameters. This are legacy settings and you can just leave them as they are because they won't be used by the app unless you want to enable password eavesdropping protection over HTTP, which is pointless when you already using SSL.

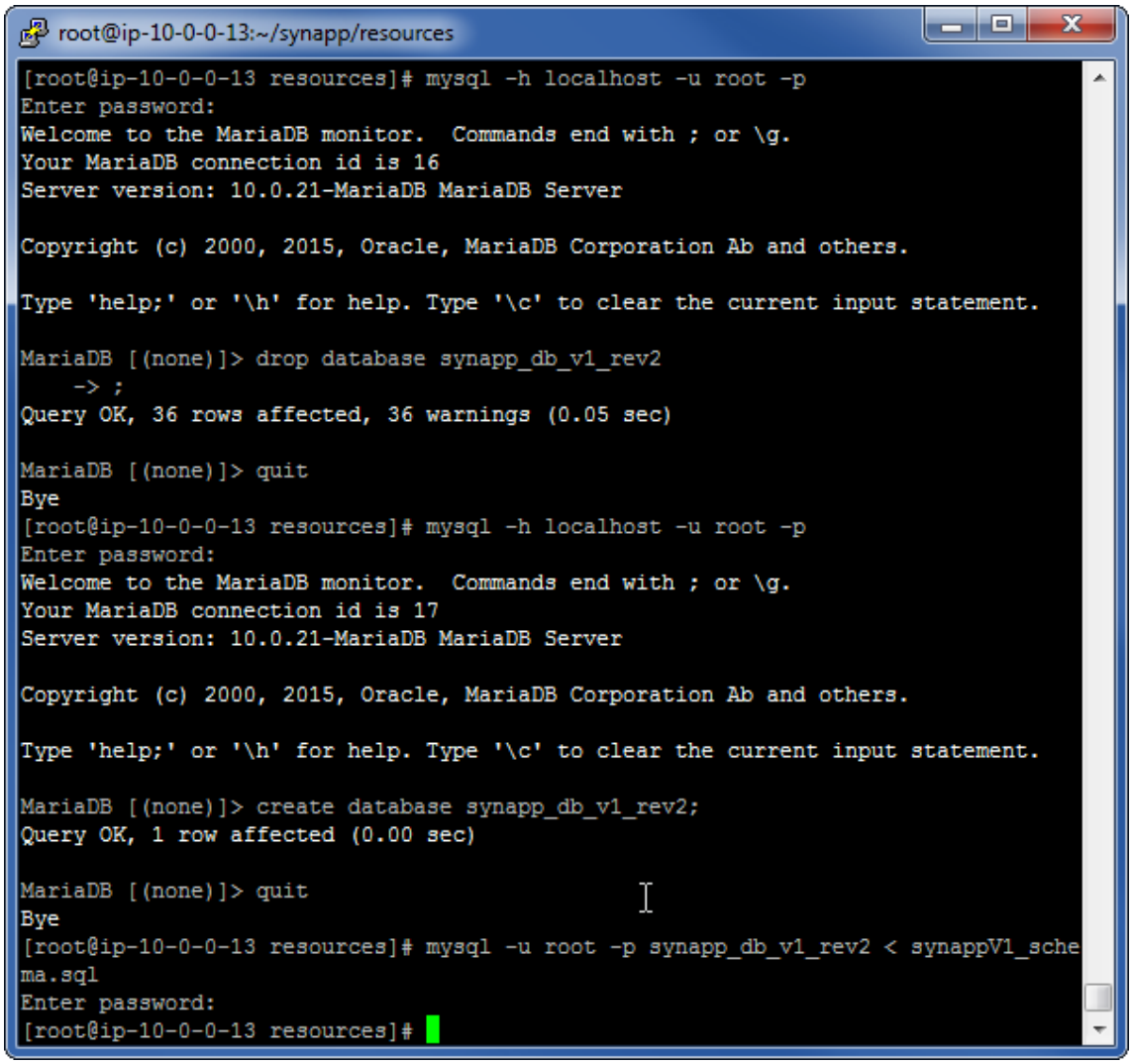


Now comes the part where we import the database. It is a very straightforward process with only two steps. The first one is creating the database and the second running the script that populates it. In this case the bundled script already creates a database, so the first step is optional.

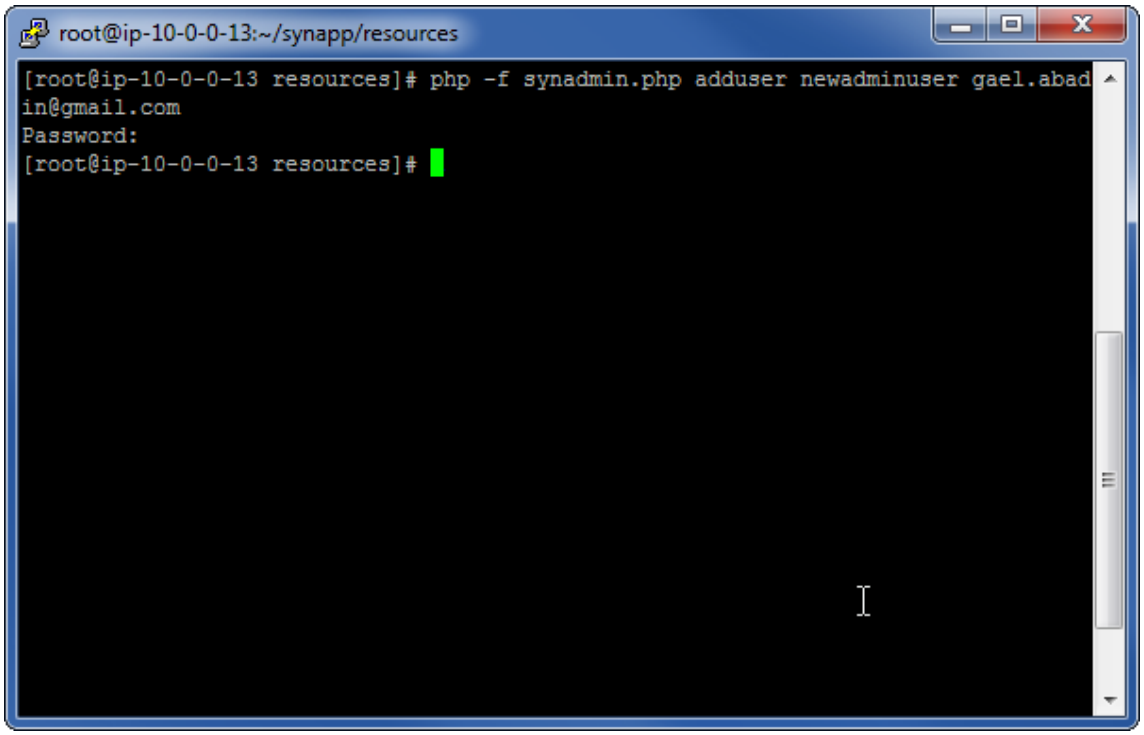
The only thing you should be careful with is to set up the proper charset encoding for your client. I will be importing directly from the console so I also need to make sure my locale is properly configured.

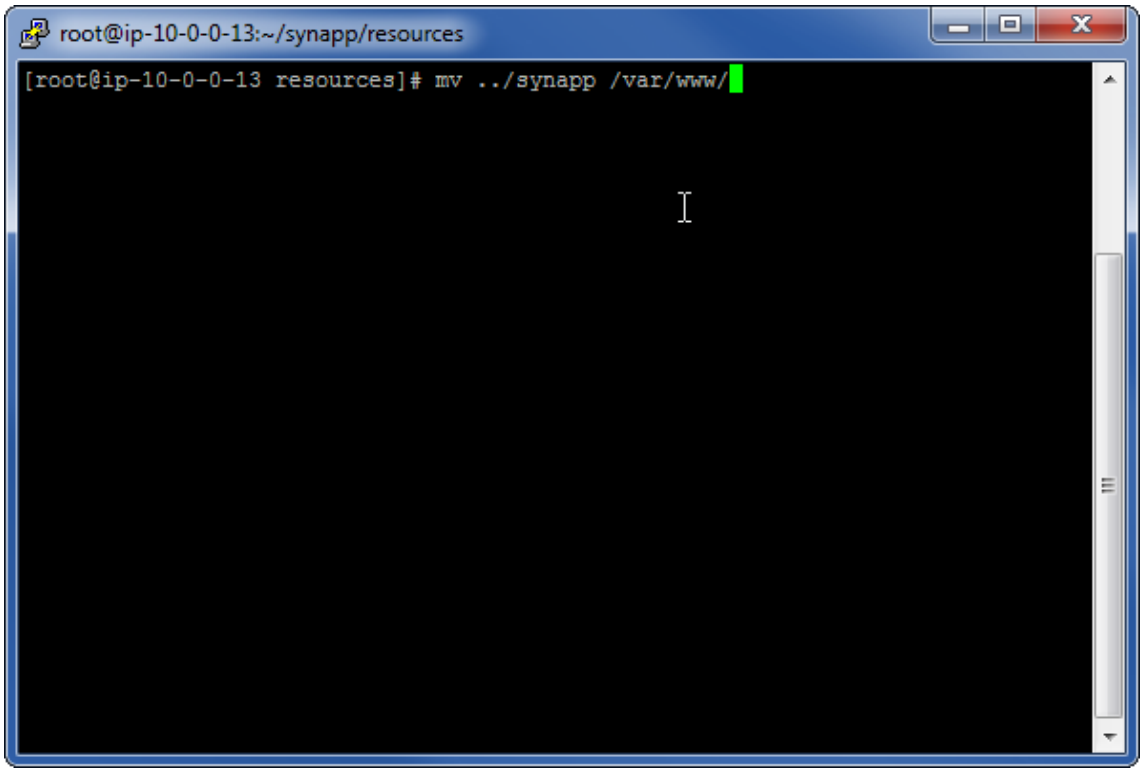
MariaDB client will connect using utf8 by default which is the encoding of the file I am about to import. The console charset is also set to UTF8 as the locale settings indicate. Once you make sure everything is alright, you can follow the process shown onscreen to load the application schema into the desired database. If you find an error like this, you need to upgrade your MySQL MariaDB version to one that supports CURRENT_TIMESTAMP as the default value on more than one column (that would be 5.6 or later, I believe)


root@ip-10-0-0-13:~/synapp/resources
[root@ip-10-0-0-13 resources]# locale
LANG=en_US.UTF-8
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
[root@ip-10-0-0-13 resources]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 10
Server version: 5.5.41-MariaDB MariaDB Server
Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MariaDB [(none)]> CREATE DATABASE synapp_db;
Query OK, 1 row affected (0.00 sec)
MariaDB [(none)]> quit
Bye
[root@ip-10-0-0-13 resources]# mysql -u root -p synapp_db < synappV1_schema.sql
Enter password:
ERROR 1293 (HY000) at line 597: Incorrect table definition; there can be only one
TIMESTAMP column with CURRENT_TIMESTAMP in DEFAULT or ON UPDATE clause
[root@ip-10-0-0-13 resources]#

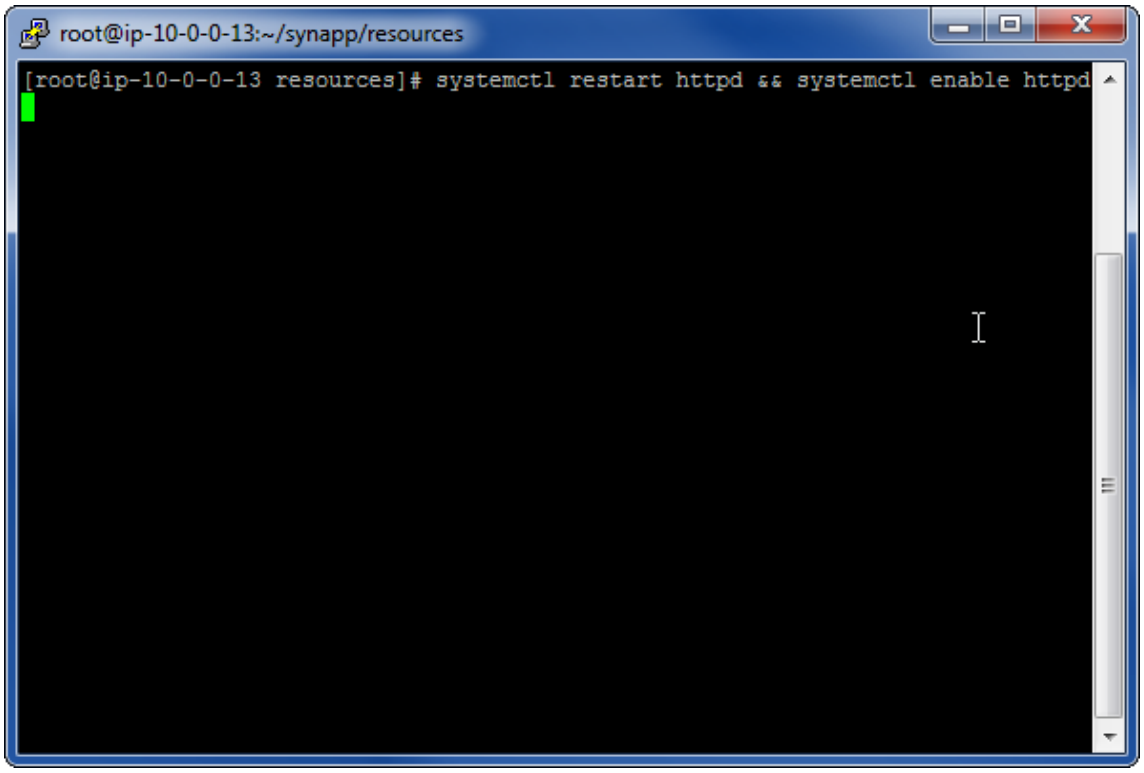


Now, this is only specific to synapp: If you want to access the administrative interface you must create an administrator using the provided CLI script, as shown on the screen. After that, you can move the application's synapp folder to a public web folder. In the second part we defined this path as /var/www, so we will move the project's synapp directory there.

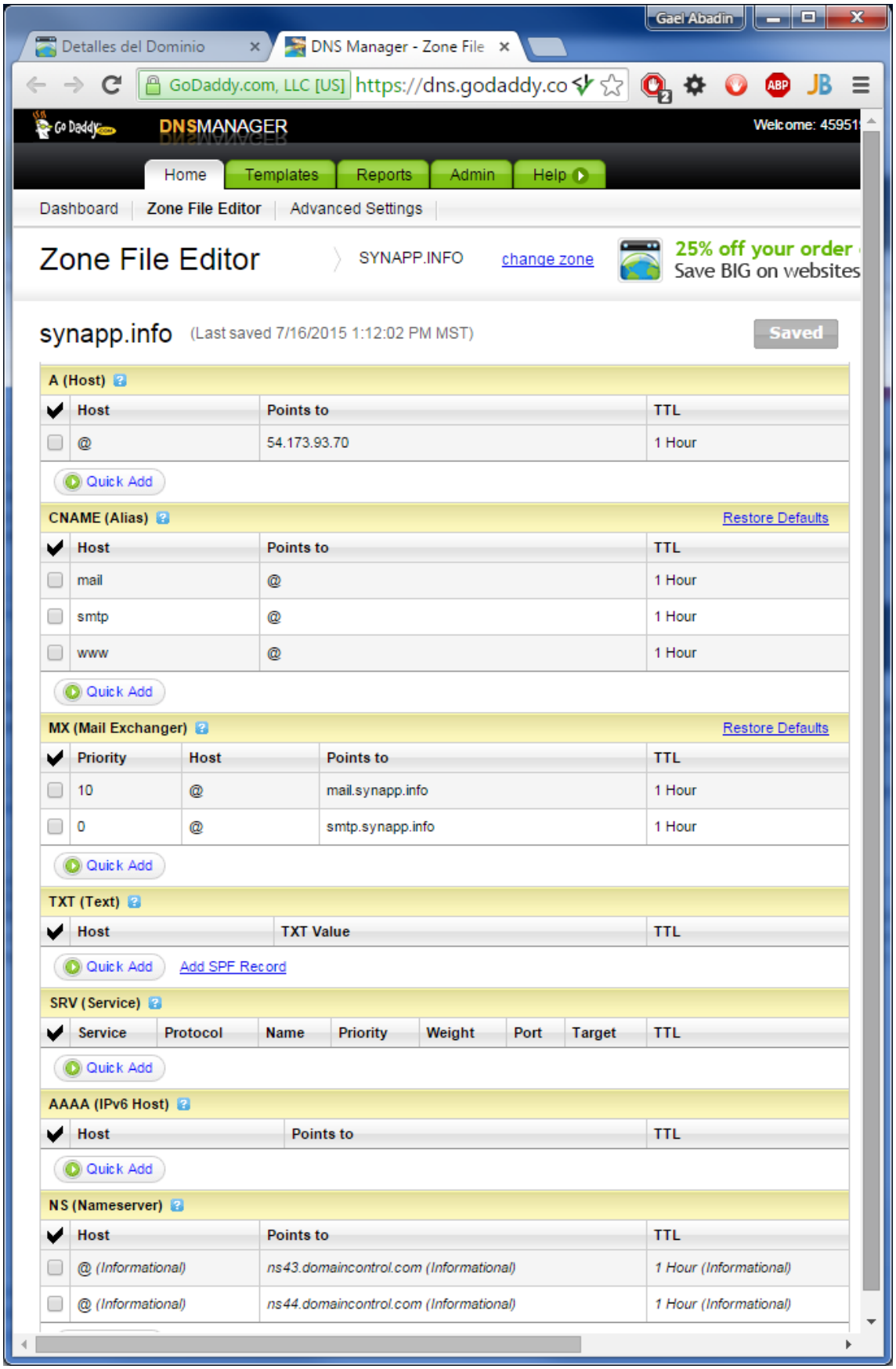




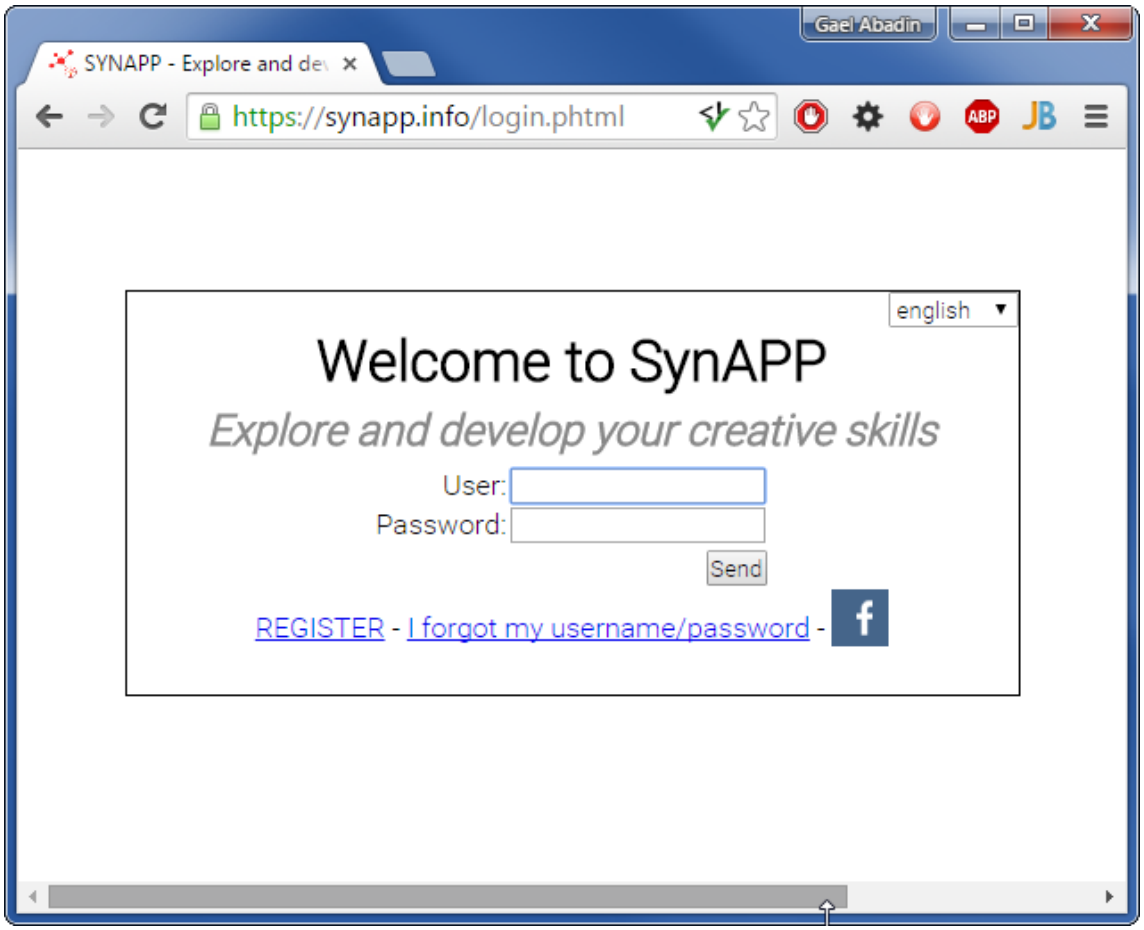
The only step left to do in the server would be to bring up the apache web server service.

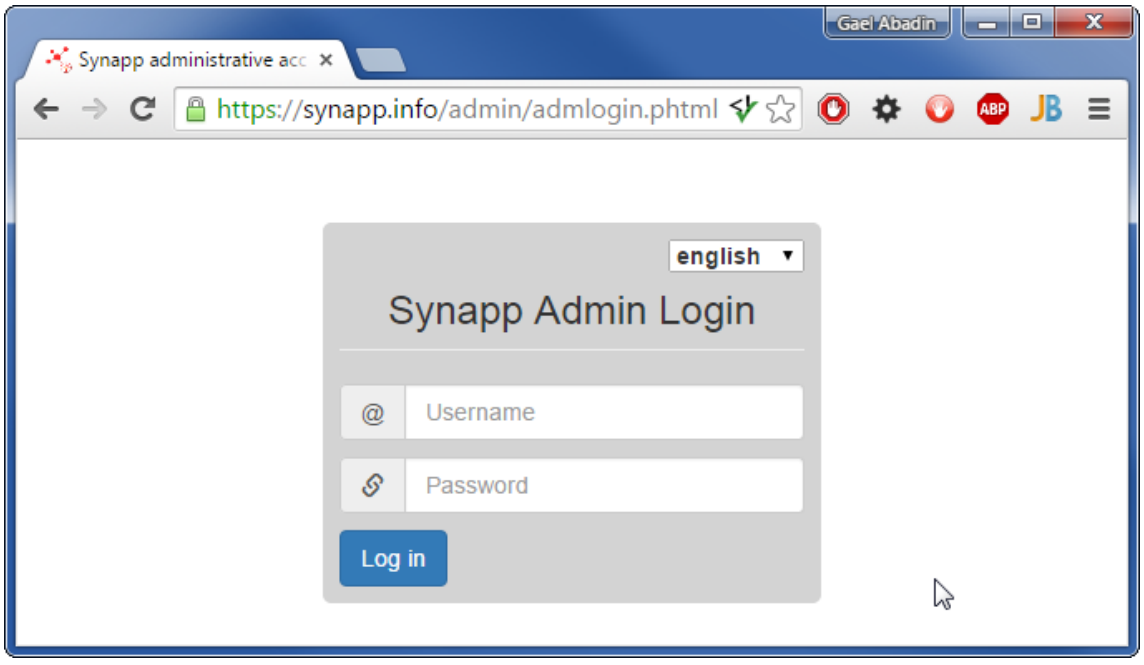


After that you should go to your registrar's web admin panel and edit the app domain's DNS zone file. What you see is an example defining 4 subdomains and a default domain plus two mx records for mail exchange services.



We have reached the end of this tutorial. If everything went ok you should be able to access your webapp, as you can see on the screen.





If you have any trouble, don't hesitate to contact me and I will answer happily to the best of my knowledge as soon as I have the time. Have fun, and happy hacking.