Installing ABCD/iah under Linux/Apache on a virtual AWS server

Recommendations and notes on Linux commands

Prepared by Wenke Adam – 1st version 12.11.2017

1. WHAT is AWS?

AWS – Amazon Web Services is a group of cloud services maintained by Amazon. Check it out at http://aws.amazon.com

In cloud services everything is elastic and scalable, adapting automatically to the volume of resources needed to run any kind of applications at any time with any level of traffic. Resources can expand and contract without our intervention. For new clients a part of these services are free for one year, within certain limits of space and traffic. If the client's installation exceeds these limits, charges start to apply automatically for the excedent. A typical ABCD installation does not exceed those limits.

If you have an Amazon account (for example for Kindle) the same account can be used to open account in AWS. If not, you create a new, free account at http://aws.amazon.com

AWS presents lots of documentation and tutorials in English and other languages. The interface is in English and has contextual help at every step.

1.a. Advantages and disadvantages of AWS for ABCD:

Advantages:
You will be able to create a virtual dedicated server where everything -i.e. the operating system, Apache and other software and the ABCD application itself - is installed and configured by you as the administrator or "owner" of the virtual server.

Disadvantages:
You will have to create a virtual dedicated server where everything -i.e. the operating system, Apache and other software and the ABCD application itself - is installed and configured by you as the administrator or "owner" of the virtual server.

This means that AWS has a learning curve!

Cloud services (and there are many of them) tend to create their own terminology and lots of acronyms, which may also complicate things for you in the beginning. For example, in AWS a server is called an "instance", go figure it, not intuitive to a new client.

1.b. What is this manual and what is it not?

This manual is aimed at colleagues who have little experience with Linux but some experience with Apache and ABCD on windows (like me).

In this manual I present the basic Linux commands and some explanations and recommendations that I figured out while installing **iah** as a OPAC for a small project. I installed it in AWS / EC2 on a virtual server (instance) with the **Amazon version of Linux** (similar to CentOS), a **LAMP** package with Apache, php and mysql, and an "abridged" version of **ABCD** containing only **iah, cgi-bin and bases.** I did not include the **Site** module, as my database will be accessed directly from a Wordpress webpage through a hyperlink.

In the manual I mention only briefly the process of creating an instance in AWS / EC2, as this is well documented on the AWS site and the interface. There are numerous external web fora dedicated to AWS that you can use.

2. Preparations

Before you start, you should have the following free software installed on your local PC:

PUTTY to communicate remotely with Linux and give commands for the creation of directories and files, give permissions, stop and restart Apache, etc.

WinSCP to communicate in graphic mode with the file structure of your server (instance) in order to upload files from your PC. WinSCP is like a light C-panel, but on your local machine.

Notepad++ to edit configuration files and avoid windows codes that Linux cannot read. Notepad++ has a lot of nice features, like the possibility of maintaining open for comparison and/or editing two or more files with the same name that reside in different folders, or to open a series of files to do search and replace in them simultaneously, etc.

3. Creating your first virtual server (instance) in AWS / EC2

1. Open an account at aws.amazon.com if you don't have one already.
2. Sign in to the Management **Console** and go to the service **EC2** (acronym for Elastic Compute Cloud).
3. Open the tutorial **Getting started guide** on the right side column. Mantain the tutorial open and follow it step by step.
4. In the Console click **Launch Instance** (big blue button).
5. Step 1 opens. Chose the Linux version **Amazon Linux AMI 2017.09.0 (HVM), SSD Volume Type - ami-c5062ba0.** It is a CentOS compatible version.
6. Follow the tutorial instructions till the end in order to create your first instance.
7. As a free first year client, you are only allowed to have one instance in operation at any time. Take note of the URL and IP of your instance. Stop and restart the instance (Actions->**Stop** and Actions->**Restart**). Note that after restarting the URL and IP have changed. This would of course be disastrous in a production setting. Rebooting the instance (Action->**Reboot)** will not cause URL/IP change. You should therefore create an "**Elastic IP**" that maintains its value after stopping the instance. More about this further down in this manual.
8. **Terminate** your first instance as instructed in the tutorial (Actions->**Terminate**), wait a minute or two until you see it marked as terminated. Continue creating and terminating instances a few more times until you get the hang of it. Until now we are only training!
9. When you create your first instance, you will also create a Security Grup and a Key Pair. The next step before starting to install LAMP and ABCD is to create at least **one user with administrator privileges** and login again to the Console using this identity before creating your definitive instance. Go to the main manual **User Guide for Linux Instances** for more information.
10. Before creating your definitive instance, delete all the Security Groups and Key Pairs that you created before and start the whole process again from scratch. This is advisable because the Groups and Keys are specific to each instance and if you later get them mixed up you will not be able to access your instance again. (That happened to me in the beginning, out of sheer enthusiasm).

3.a. Creating the definitive instance

3.a.1. Create the instance chosing the **AWS Amazon Linux** as indicated above.

3.a.2. Security Group

One Security Group will be created automatically by default. You can create another one with a name chosen by you.
You should create three **Inbound Rules** in the active Security Group**.**

```
ssh     tcp  22
http    tcp  80
https   tcp 443
```

3.a.3. Key Pair

Create a Key Pair for this instance and give it a name. Download the Key to your PC (and keep an extra copy in a safe place) and open **PUTTYgen** (a small program that comes with PUTTY) to create a PUTTY like key with the extension **.ppk**. Follow the instructions in the AWS and PUTTY manuals. Create the .ppk key with **Save**, do not generate it with **Generate**, it will not work!

3.a.4. Elastic IP

I strongly recommend creating a Elastic IP at this stage of the process. You have the right to one Elastic IP per instance. Amazon only charges you if you create it and don't use it. Follow the instructions in the AWS / EC2 manual and **Associate** the Elastic IP to your instance. From now on your Elastic IP becomes permanent and will also be reflected in the URL name. If your Elastic IP is **123.45.67.89**, your URL will be **ec2-123-45-67-89.us-east-2.compute.amazonaws.com**. (In my case, clients form Chile are automatically placed in the US Ohio, or east2, AWS zone. Yours could be different).

4. Installing LAMP (Apache, php, mysql) with PUTTY

Configure PUTTY to communicate with your instance. You will need your URL, the user name which is always **EC2-user**, the full path to the .ppk Key on your computer and a password. Follow the AWS and PUTTY instructions.

Here we start giving commands to Linux via PUTTY

First we will install LAMP. I chose to install it with php 5.6.

Note: Sudo is a Linux command that tells the system that we are communicating with the authority of a root administrator and have access to all the contents on our server. **Use it wisely!**

4.a. Update LAMP (checking for LAMP updates before installing):

```
sudo yum update -y
```

4.b. Install LAMP:

```
sudo yum install -y  httpd24 php56 mysql55-server php56-mysqlnd
```

Note: if you prefer php70, enter:

```
sudo yum install -y  httpd24 php70 mysql56-server php70-mysqlnd
```

And if you don´t need mysq, just enter `httpd24 php70.`

I recommend installing the full package, in case you later want to use the full ABCD with EmpWeb.

4.c. Verify LAMP installation:

```
sudo yum list installed httpd24 php56 mysql55-server php56-mysqlnd
```

Should show a list similar to:

```
Loaded plugins: priorities, update-motd, upgrade-helper
Installed Packages
httpd24.x86_64          2.4.25-1.68.amzn1            @amzn-updates
mysql56-server.x86_64   5.6.35-1.23.amzn1            @amzn-updates
php70.x86_64            7.0.14-1.20.amzn1            @amzn-updates
php70-mysqlnd.x86_64    7.0.14-1.20.amzn1            @amzn-updates
```

4.d. Start Apache:

```
sudo service httpd start
```

4.e. Make Apache refresh config on each reboot:

```
sudo chkconfig httpd on
```

4.f. Confirm that Apache is running:

```
chkconfig --list httpd
```

Shows this list:

```
httpd   0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

5. Testing Apache in the browser

5.a. Here I will use a token IP / URL as example.

With the Public DNS:

```
ec2-123-45-67-89.us-east-2.compute.amazonaws.com
```

With the IPv4 Public IP (This is the elastic IP):

```
123.45.67.89
```

You should see the Apache greetings page.

5.b. Where are Apache and home files? :

```
ls -l /var/www
```

should show something like:

```
total 16
drwxr-xr-x 2 root root 4096 Jul 12 01:00 cgi-bin
drwxr-xr-x 3 root root 4096 Aug  7 00:02 error
drwxr-xr-x 2 root root 4096 Jan  6  2012 html
drwxr-xr-x 3 root root 4096 Aug  7 00:02 icons
drwxr-xr-x 2 root root 4096 Aug  7 21:17 noindex
```

The default doc root is: **/var/www/html/,** but we will use /var/www/vhosts/ABCD/

6. Configuring Apache and ABCD

6.a. Create directory vhosts:

```
sudo mkdir /var/www/vhosts/
```

6.b. Create a subdir for each application you would like to install (for the moment, only ABCD):

```
sudo mkdir /var/www/vhosts/ABCD/
```

6.c. Create the subdir www of ABCD while we are at it

```
sudo mkdir /var/www/vhosts/ABCD/www/
```

6.d. Adding ownership and permissions

6.d.1. Add ec2-user to the Apache group: (you can add more users later)

```
sudo usermod -a -G Apache ec2-user
```

6.d.2. Log out and login again in PUTTY to refresh and verify your membership in the Apache group:

Log out: `exit`

PUTTY disconnects.

Login: Reconnect PUTTY to the instance (do a login from PUTTY).

6.d.3. Check status of the Apache group:

```
groups
```

You should see:

```
ec2-user wheel Apache
```

6.d.4. Change ownership of var/www and its contents to the Apache group:

```
sudo chown -R ec2-user:Apache /var/www
```

6.d.5. Add write permissions to var/www to prepare for future subdirs:

```
sudo chmod 2775 /var/www
find /var/www -type d -exec sudo chmod 2775 {} \;
```

6.d.6. Add recursive permissions to subdirs:

```
find /var/www -type f -exec sudo chmod 0664 {} \;
```

6.d.7. Create a test PHP doc in doc root:

```
echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

If echo doesn't work, do logout/login again to refresh Apache with the new permissions.

6.d.7. In the browser, test php:

```
      http://ec2-123-45-67-89.us-east-2.compute.amazonaws.com/phpinfo.php
or    http://123.45.67.89/phpinfo.php
```

You should see the PHP inf page.

6.d.8. Deleting the phpinfo.php file: (After finishing all configurations!)

```
rm /var/www/html/phpinfo.php
```

7. Configuring VirtualHost

We have already created the var/www/vhosts/ABCD/ directory to contain ABCD.

7.a. Create a file vhost.conf in the directory /etc/ where the Apache configurations are:

```
cd /etc/httpd/conf.d
sudo touch vhost.conf
```

7.b. Open the Linux nano editor to add contents to vhost.conf:

```
sudo nano /etc/httpd/conf.d/vhost.conf
```

7.c. In nano, add a VirtualHost for ABCD and a generic one that follows the default config.

```
<VirtualHost *:80>
      ServerName ec2-123-45-67-89.us-east-2.compute.amazonaws.com
      ServerAlias 123.45.67.89
      DocumentRoot /var/www/vhosts/ABCD/www/htdocs/
      ScriptAlias /cgi-bin/ "/var/www/vhosts/ABCD/www/cgi-bin/"
      ServerAdmin mimail@gmail.com
      ErrorLog /var/www/vhosts/logs/ABCD_error_log

      <Directory /var/www/vhosts/ABCD>
            AllowOverride All
      </Directory>

      <Directory "/var/www/vhosts/ABCD/www/cgi-bin/">
            Options +ExecCGI
            AddHandler cgi-script .cgi
      </Directory>

</VirtualHost>

<VirtualHost *:80>
      ServerName default:80
      DocumentRoot /var/www/html/
      ServerAdmin mimail@gmail.com
      ErrorLog /var/www/vhosts/logs/error_log
</VirtualHost>
```

7.d. Save vhost.conf and close nano:

`Ctrl-x` and then `Y` (yes) to confirm saving.

7.e. Note: The log files are optional. If we declare them in VirtualHosts, we also have to create the subdir /logs/ and the two log files before refreshing Apache again.

```
cd /var/www/vhosts
sudo mkdir /var/www/vhosts/logs
sudo touch error_log
sudo touch ABCD_error_log
```

7.f. Restarting Apache

```
sudo service httpd restart
```

8. Uploading ABCD to the instance:

First you have to configure WinSCP to communicate with the server. Follow the instructions in the AWS manual and WinSCP help.

You can of course upload the full ABCD package to the `/var/www/vhosts/ABCD/www/` folder. In my case I prepared a minimal ABCD only for iah on my computer (windows) and made sure that everything was working fine in localhost. Then I uploaded **bases, cgi-bin and htdocs**, following the ABCD structure. If something doesn't work in the AWS installation, but worked fine in localhost, you will know that the problem most probably lies in errors of paths in the config files, lack of permissions, or having uploaded windows versions instead of Linux versions of cgi-bin and bases.

The iah installation should have the following ABCD folders:

`cgi-bin`	Copy the Linux version of the ABCD cgi-bin folder.
`bases`	Contains the databases biblo y mybase, the par folder and the config files. Biblo was included for testing purposes, just in case mybase had problems. Copy bases from the Linux version of ABCD. Remember that if you upload a database from a windows installation you must first apply crunchmf and crunchif to the database files before uploading.
`htdocs`	Copy the full iah folder into htdocs.

These three folders should go into the `/var/www/vhosts/ABCD/www/` directory with WinSCP.
The cgi-bin and bases folders must upload as binary files, and htdocs as text files.

With WinSCP we give the following general and recursive permissions (Or you can do it with PuTTY using sudo chmod and find):

```
cgi-bin  2775
bases    0777
htdocs   0664
```

8.a. Configuring ABCD/iah

Edit the ABCD configuration files if you haven't done it before uploading. Use nano or the WinSCP editor on the server, or edit locally with Notepad++.

```
/var/www/vhosts/ABCD/www/htdocs/iah/scripts/iah.php.ini
/var/www/vhosts/ABCD/www/bases/abcd.def
/var/www/vhosts/ABCD/www/bases/bases.dat
```

8.b. Configuring iah.def.php

```
<?php
[PATH]
PATH_DATA=/iah/
PATH_CGI-BIN=/var/www/vhosts/ABCD/www/htdocs/iah/scripts/
PATH_DATABASE=/var/www/vhosts/ABCD/www/bases/
PATH_DEF=/var/www/vhosts/ABCD/www/bases/par/
[APPEARANCE]
BODY BACKGROUND COLOR=#F7F5D0   (in my case, I chose a light cream colour)
/* Please adjust /css/stylesheet.css */
[HEADER]
LOGO IMAGE=logo-ad.jpg
LOGO URL=^1http://www.mydomain.cl^2http://www.mydomain.cl^3http://www...
      mydomain.cl^4http://www.mydomain.cl
HEADER IMAGE=online.gif
HEADER URL=^1http://www.mydomain.cl^2http://www.mydomain.cl^3http://www..
      mydomain.cl^4http://www.mydomain.cl
[IAH]
MANAGER E-MAIL=mymail@gmail.com
REVERSE MODE=OFF
MULTI-LANGUAGE=OFF
AVAILABLE LANGUAGES=es
?>
```

In this example, I substituted the links that come as default on the iah interface (the logo link goes to BVS

Bireme and the heading “Search in databases” goes to Site). Instead I placed token links to the site www.mydomain.cl.

8.c. Configuring abcd.def

```
VERSION=1.4     (or the one you are using
LEGEND1=Name of organization
LEGEND2=Name of organization
URL1=http://www.mydomain.cl
URL2=http://www.mydomain.cl
dubcore=unicode
unicode=unicode
diglib=ffi
```

In my case, it doesn't seem necessary to edit the abcd.def file because it is not used by iah, it is only activated in Site and Central.

8.d. Configuring bases.dat

Include the bases you have, for example BIBLO and MYBASE.

```
biblo|BIBLO|Y
mybase|MYBASE|Y
```

8.e. Test in the browser that wxis is working:

http://123.45.67.89/cgi-bin/wxis?hello

Wxis should reply:

```
CISIS Interface v5.7e/G/PC/W/M/32767/16/60/I/64bits - XML IsisScript WWWISIS
7.1f
CISIS Interface v5.7e/.iy0/Z/GIZ/DEC/ISI/UTL/INVX/B7/FAT/CIP/CGI/MX/W
Copyright (c)BIREME/PAHO 2010. [http://reddes.bvsalud.org/projects/cisis]
WXIS release date: Aug  5 2015

WXIS|missing error|parameter|IsisScript|
```

8.f. Test in the browser that wxis communicates with iah:

http://123.45.67.89/cgi-bin/wxis?IsisScript=getenv.xis

Wxis should reply something like:

```
CGI Environment Variables
AUTH_TYPE=
CONTENT_LENGTH=
CONTENT_TYPE=
GATEWAY_INTERFACE=CGI/1.1
HTTP_ACCEPT=text/html,application/xhtml+xml,application/xml;q=0.9,image/webp
,image/apng,*/*;q=0.8
HTTP_USER_AGENT=Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36
PATH_INFO=
PATH_TRANSLATED=
QUERY_STRING=IsisScript=getenv.xis
REMOTE_HOST=
REMOTE_ADDR=152.172.80.148
REMOTE_IDENT=
REQUEST_METHOD=GET
REMOTE_USER=
SERVER_NAME=123.45.67.89
SERVER_SOFTWARE=Apache/2.4.27 (Amazon) PHP/5.6.31
SERVER_PORT=80
SERVER_PROTOCOL=HTTP/1.1
SCRIPT_NAME=/cgi-bin/wxis
IsisScript Environment Variables
PATH_WXIS=/var/www/vhosts/ABCD/www/cgi-bin/
```

If wxis can not get to the script getenv.xis in the /iah/scripts/ directory, it will show instead error message:

```
WXIS|file error|file open|Isis_Script|
```

8.f. Test in the browser that iah and the base BIBLO are working:

```
http://123.45.67.89/cgi-bin/wxis/iah/scripts/?IsisScript=iah.xis&lang=es&base=BIBLO
```

You should see the main, simple search page of the BIBLO database.

If something doesn´t work it is almost invariably due to errors in configuration paths, either in Apache or ABCD.

8.g. To test your own database, just substitute BIBLO with MYBASE in the browser.

```
http://123.45.67.89/cgi-bin/wxis/iah/scripts/?IsisScript=iah.xis&lang=es&base=MYBASE
```

Now you can add this URL to the bookmarks bar, as MYBASE AWS, for example, so you can open it as often as you need until a permanent link has been established to the mother website.

9. **Next steps (work in progress)**

9.a. Create a repository outside EC2 (in Amazon S3) for images and othe static files linked to the database.
Still studying this. My database has more than 5.000 jpg medium resolution images and some 500 pdf files, plus the corresponding 5.500 thumbnail files that are shown in the records. I have uploaded them to subdirs in the iah directory, just as I had them in my localhost installation. My AWS / EC2 ABCD installation now weighs about 2,5 GB. That plus Linux and lamp is still far below the 5 GB limit of the free first year contract.

Storage in a “bucket”or container in the **S3** service of Amazon is free up to 5 GB. After the first year it will cost far less than EC2 storage, and there are other solutions for larger repositories too.

9.b. Associate a subdomain from www.mydomain.cl to this instance.
Still studying this. My domain and wordpress site are currently hosted at Hostgator. As far as I understand, the asociation is done through the AWS DNS manager called **Amazon Route 53.** It involves changing configurations in NIC Chile, Hostgator and Route 53.

9.c. Migrate the wordpress website from Hostgator to AWS.
Still studying this. Sounds terrifying. But doable. (I'm always optimistic). There are many forums about wordpress migration to AWS. Wish me luck!

10. Conclusions

While it looked complicated at first, once I got the hang of it I managed to complete the whole process (creating the instance, installing Linux and LAMP, uploading ABCD and configuring everything) in about 3 hours, plus the 6 hours I let the uploading of the repository running overnight unattended.

After the first free test year at AWS, I guess this particular installation will cost about USD 5,00 per month, which compares well to the prices of VPN dedicated virtual servers offered by traditional webhosting companies. It is important to know that Amazon charges for absolutely every resource used, by the hour or by the minute, but as the AWS services are designed with very large systems in mind, with heavy traffic (Netflix is among the giants that rely on AWS), the needs of a simple OPAC of a small entity will probably always fit under the lowest costs.

Try it!

wenkeadam@gmail.com