



AWS Fundamentals – Assignment #1

OBJECTIVE

At the end of this assignment you will have created a web site using the following Amazon Web Services: EC2, EBS, ELB and S3 and CloudFormation.

Stage 1: Building the EC2 web server and Elastic Load Balancer.

Launch an EC2 instance, based on Linux or Windows, to meet the following objectives:

- ✚ the instance should be of type t1.micro.
- ✚ the instance should reside within region ap-southeast-1 within availability zone ap-southeast-1a.
- ✚ the instance should use a 1 GiB attached EBS volume and contain a valid partition table with one partition. The partition should contain a valid file system.
- ✚ the file system residing on the EBS volume should be mounted automatically upon reboot of the EC2 instance.
- ✚ the instance should serve web pages via an appropriate service such as Apache or IIS. This service should start automatically upon boot.
- ✚ the instance should serve a web page "index.html" containing well-formed HTML displaying the text "Hello AWS World" and display the screen shots created below in Stage 3 (they will be hosted separately). The HTML file should reside on the file system within the previously created EBS volume and be served as the default document from the web server root.
- ✚ the instance should use Security Groups effectively to allow administration and serve HTTP.

Stage 2: Configuring the Elastic Load Balancer:

Create an Elastic Load Balancer (ELB) with the following specification:

- ✚ the ELB should be created in the Singapore region.
 - ✚ the ELB should accept HTTP on port 80.
 - ✚ the Healthy Threshold for the ELB to be set to 2
 - ✚ deliver traffic to the EC2 instance created in Stage 1.
-



Stage 3: Configuring S3

Create a Simple Storage Service (S3) bucket with the following specification:

- ✚ the bucket should be created in the Singapore region.
- ✚ the bucket should be publicly readable.

Place in the S3 bucket screen shots, in png format, clearly showing the following:

1. The mounted EBS volume e.g. using Windows Explorer or run "df" from the console on a Linux host. This screen shot should be named as screen-shot1.
2. The index.html file resides within EBS e.g. using windows explorer or run "pwd; ls -l" from the terminal on a Linux host. This screen shot should be named as screen-shot2.png.
3. The web server has been configured to serve index.html from the EBS volume as the default document e.g. the relevant section of the Apache configuration file or IIS Manager. This screen shot should be named as screen-shot3.png.

Remember to use the S3 URLs in the index.html file hosted on EC2 (see Stage 1).

Optional: if you choose to extend the solution with CloudFront then use CloudFront URLs.

Stage 4: Create an Amazon Linux Web Server Using CloudFormation (CFN) Template:

In this stage, candidate is required to create a new EC2 instance for Web Server with a simple [CloudFormation](#) (CFN) template by meeting the requirements with detailed [description](#) as mentioned below:




1. [Parameters](#) - Instance type of "t2.micro" as default. Provide option to allow choosing instance types of others such as "t2.small" and "t2.medium" as well. [2 marks]
2. [Parameters](#) - Allow user to specify EC2 key pair name that you want to associate with this instance. Note: Name is required. [2 marks]
3. [Mappings](#) of Amazon Machine Image (AMI) ID for all the regions for AWS except China Region and US Government Cloud. [2 marks]
4. EC2 security group to enable for accessing Web and SSH traffic from outside. [2 marks]
5. [Resource](#) and [User Data](#) - Use a script in User Data to install and start Web

Server. [4 marks]

6. Define the [outputs](#) of the CFN with including of following items: [4 marks]
- Instance ID of the Web Server
 - Instance is running in Availability Zone
 - Public IP address of the instance
 - Instance Public DNS Name

DELIVERABLES

Please provide to janetli@amazon.com and yajimeng@amazon.com the following information:

-  the public DNS entry for the EC2 instance [2 marks]
-  the public URL to the web page via the ELB [2 marks]
-  Executable CFN in JSON format for Stage 4. Candidate needs to specify which items of Stage 4 that he/she could not generate/produce the results accordingly. Marks will be given for whatever codes that are valid instead. [16 marks]

Please Note: Ensure that you leave your solution up and running and available in order for it to be evaluated. This should occur within three to five business days of submitting your assignment. Upon receiving feedback from the Recruitment Team, you should terminate and delete all resources you used for the assignment so as to avoid any unnecessary charges.

Appendix A – How Do Scripts, Packages, and Templates Work Together?

CloudFormation is an orchestrated experience that combines scripts with a service named **cloud-init** that runs on the instance. Most Amazon Linux, Ubuntu, and Windows AMIs have this software already installed to run as a service. From the point of view of CloudFormation, both Linux and Windows behave exactly the same.

Bootstrap Applications from a CloudFormation Template

A three-way combination allows CloudFormation and an AMI to interact with each other.

1. You can specify helper scripts from the CloudFormation template.
2. There is a package known as *cloud-init* that runs on the AMI.
3. **cloud-init** runs at startup as an rc script: e.g. *K25cloud-init* in the *rc6.d* directory.

Helper Scripts

AWS CloudFormation provides the following helpers to allow you to deploy your application code or application and OS configuration at the time you launch your EC2 instances:

- **cfn-init**: Used to retrieve and interpret the resource metadata, install packages, create files, and start services.
- **cfn-signal**: A simple wrapper to signal a CloudFormation WaitCondition allowing you to synchronize other resources in the stack with the application once it's ready.
- **cfn-get-metadata**: A wrapper script making it easy to retrieve either all metadata defined for a resource, or supply a path to a specific key or sub-tree of the resource metadata.
- **cfn-hup**: A daemon to check for updates to metadata and execute custom hooks when changes are detected.

These scripts are installed by default on Amazon Linux AMIs in */opt/aws/bin*. They are also available in the Amazon Linux AMI yum repository, and via RPM for other Linux/Unix distributions.

cloud-init

cloud-init is an open-source package written by Canonical (the authors of Ubuntu) that runs under Linux to facilitate early startup scripts such as those in a CloudFormation template.

Amazon Linux, and most Ubuntu AMIs, and most recent Windows AMIs have the equivalent to *cloud-init* installed. You'll need to check other Linux distributions, and install if needed. We are not going to discuss how to install the package here.

cloud-init, cloudinit, and cloudinit.d are all synonyms for the same concept

Set up a Wait Condition

A WaitCondition tells the CloudFormation stack to wait for some future signal. Script might uses the **cfn-signal** command to send a wait condition called **WaitHandle** to signify that it has finished executing.

In order to instruct our CloudFormation stack to wait for this signal, we must define a **WaitHandle** resource, and then specify a **WaitCondition**:

```
"WaitHandle" : {
  "Type" : "AWS::CloudFormation::WaitConditionHandle"
},

"WaitCondition" : {
  "Type" : "AWS::CloudFormation::WaitCondition",
  "DependsOn" : "Ec2Instance",
  "Properties" : {
    "Handle" : {"Ref" : "WaitHandle"},
    "Timeout" : "300"
  }
}
```

The Timeout setting is the number of seconds before the **WaitCondition** assumes that something went wrong and reports an error. In this example, if no signal is received after five (5) minutes, CloudFormation will report that it failed to create the stack.

Log Files

CloudFormation and **cloud-init** create log files which can help diagnose issues when you begin to work with more advanced scenarios. The two log files of interest are **cfn-init.log** and **cloud-init.log**; on Linux systems, these are located in the **/var/log** directory.