CloudFormation

Configuration Management and Continuous Delivery in the Cloud

Sadek, Faras

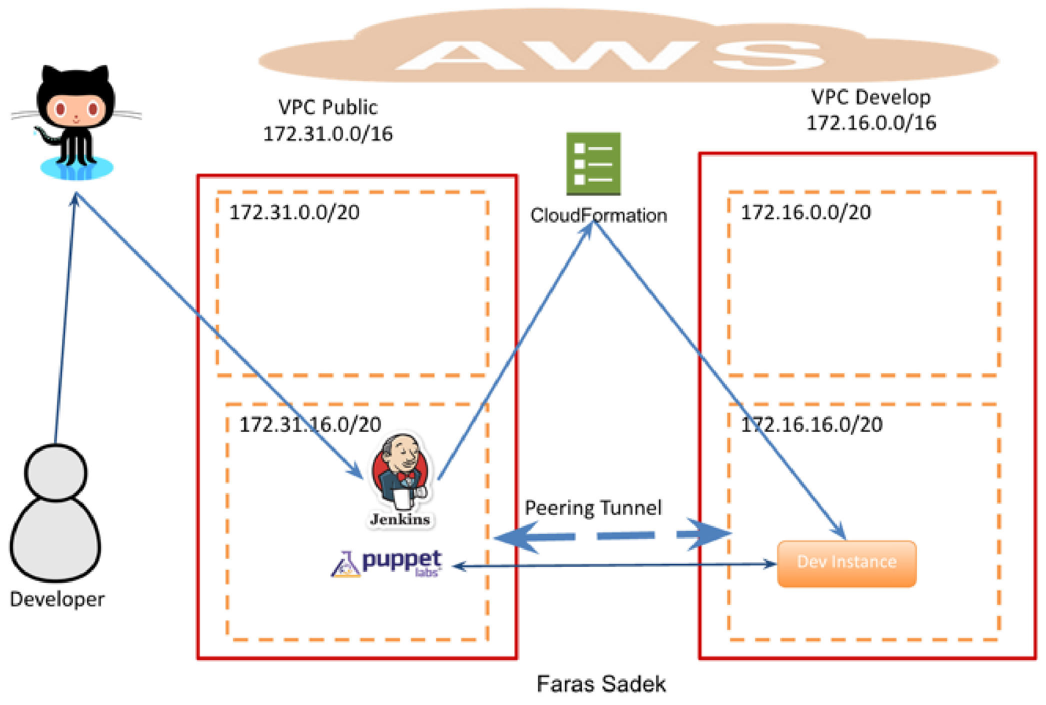
cscie90 Cloud Computing

Harvard Extension School

prof. Zoran B. Djordjevic

Summary

My project uses several tools to deliver the goal of continuous delivery and integration in the cloud. For a development users , the devops users, it make their life easy in term of automation and the isolation of the development environment test from the production environment.
The used pushes his change to the development repository branch and the system builds the necessary environment to build and test the new configuration.
puppet used for the autoconfiguration and management system. Jenkins used to trigger the build process and test the result.



1. DevOps user pushes new change to the development branch in puppet repository,
2. github will capture the change and send a post web-hooks to Jenkins,
3. Jenkins will create an EC2 instance in the development VPC,
4. the instance will be configured using puppet,
5. if the process succeed, Jenkins will show a success and only in this case the developer can merge the development branch to the master branch to be ready for the test.

The Links for the Videos and the Demo will be in a README File. I am having difficulties uploading my videos to youtube, its taking forever today.

Cloud Formation

Tools for the Project

- **CloudFormation**
- **Puppet**
- **GitHub**
- **Jenkins**

Lets starts some details on each of those tools and how they fit together.

Cloud Formation

CloudFormation

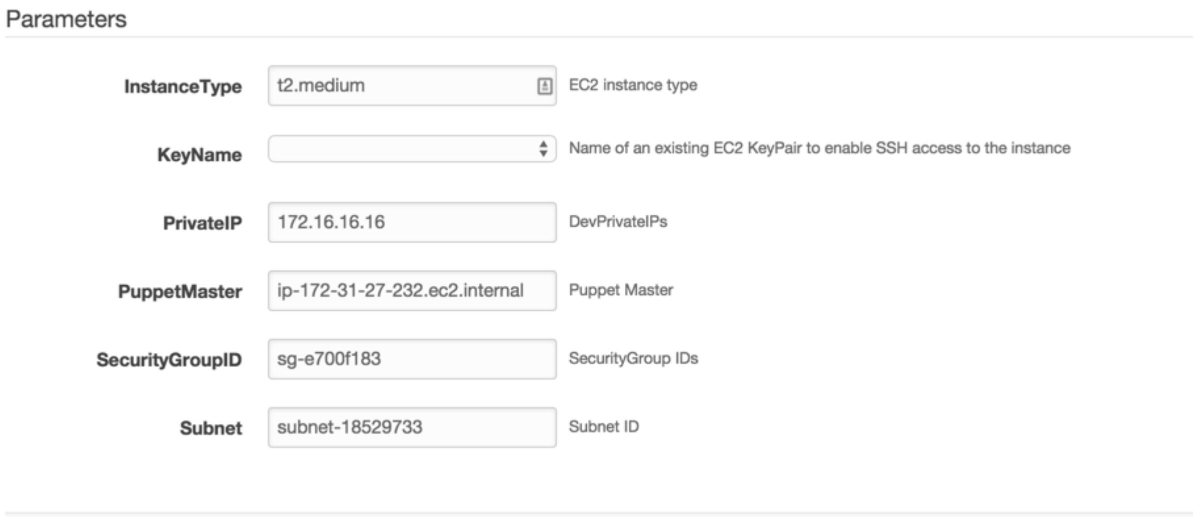
CloudFormation is a great bootstrapping scripting tool. By using other automation tools for configuration management like puppet, and continues delivery like Jenkins, system automation become more powerful and fun. Therefore, I will talk about the continues delivery and configuration management using cloudFormation, puppet and Jenkins.

What is cloudformation?

CloudFormation is a script, specifically JSON script, to allocate virtual resources in AWS environment, such as EC2 instances , Elastic IP, S3 bucket , IAM users, etc.

The template is split up into four sections:

1. **Parameters**: parameters are values that the user define in the template. Then will be prompted to enter their values during the stack creation process. I this project I used the following parameters:



Parameters

InstanceType	t2.medium	EC2 instance type
KeyName		Name of an existing EC2 KeyPair to enable SSH access to the instance
PrivateIP	172.16.16.16	DevPrivateIPs
PuppetMaster	ip-172-31-27-232.ec2.internal	Puppet Master
SecurityGroupID	sg-e700f183	SecurityGroup IDs
Subnet	subnet-18529733	Subnet ID

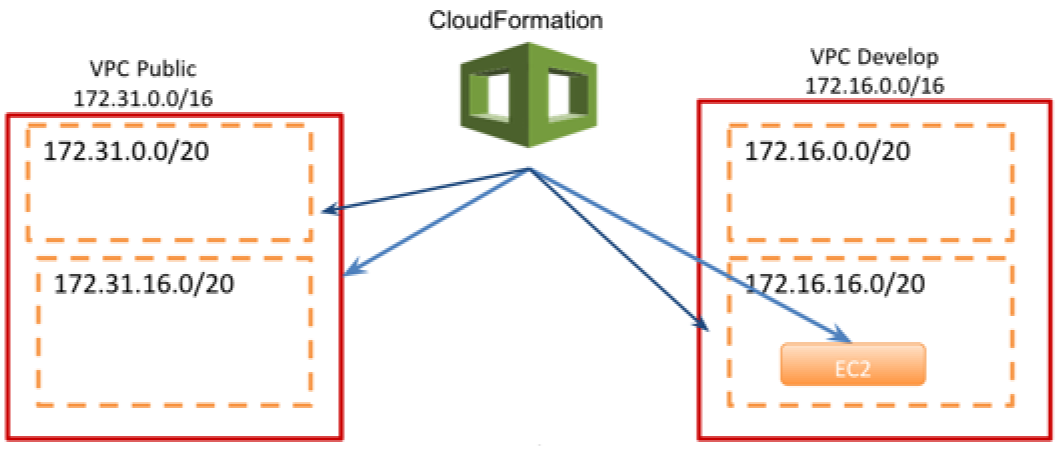
2. **Mappings**: Mapping are for specifying conditional parameter values in the template. For instance you might want to use a different AMI depending on the

region your instance is running on. I did not have to use the mapping in my project.

3. **Resources**: Resources are the most important part of the CloudFormation template. Inside the resource section, you define and configure your AWS components. Resources that created using my templates are:
    a. VPC
    b. Public subnet
    c. Private Subnet
    d. EC2 Instance
4. **Outputs**: Output will return values, if the stack resources are created successfully. It may return values like IP address or DNS of the created instance. This is optional and I did not pay great attention to it.

I used two templates to create resources in my project. Those templates used to :

1. Create a VPC with private and public subnets. This case is used to build my environment, by applying this template twice for the public and development VPCs.
2. Another CloudFormation template used to create an EC2 instance in development VPC, in the peering subnet of the development VPC, specifically in the subnet that peer the development with the public VPC.



Cloud Formation

The second one is of the greater importance to the devops user. Since this is the template that will run the development VPC.

CloudFormation repository can be found in this link:
https://github.com/FarasSadek/e90_cfm

Puppet

Puppet is a declarative language for IT Infrastructure automation. Puppet can manage the provisioning, configuration and patching packages over the operating system. Its a great tool for managing application component across data-centers and various cloud environment. Puppet is a configuration management that I used heavily for the automation process. I used puppet to install and configure the required packages in the EC2 instance.

When the instance is created in the development VPC using the cloudFormation template, puppet will take over to install and configure the packages and application on the machine over the peering tunnel.
Puppet master is stored in a git repository in gitHub. It has two branches , the master and the development branch. The development branch will be merged to the master if it work. Jenkins will decide if the development branch test succeed or not.

When the user push a change to the development branch, it will trigger gitHub to send a web-hooks post to Jenkins, then Jenkins will do the rest.

Puppet repositoty can be found in this link: https://github.com/FarasSadek/e90_puppet

Cloud Formation

Jenkins

We finally get to Jenkins. Jenkins is an open source continuous integration tools. It used to be known as Hudson. I personally had a great experience with Hudson as a testing and automation tools.

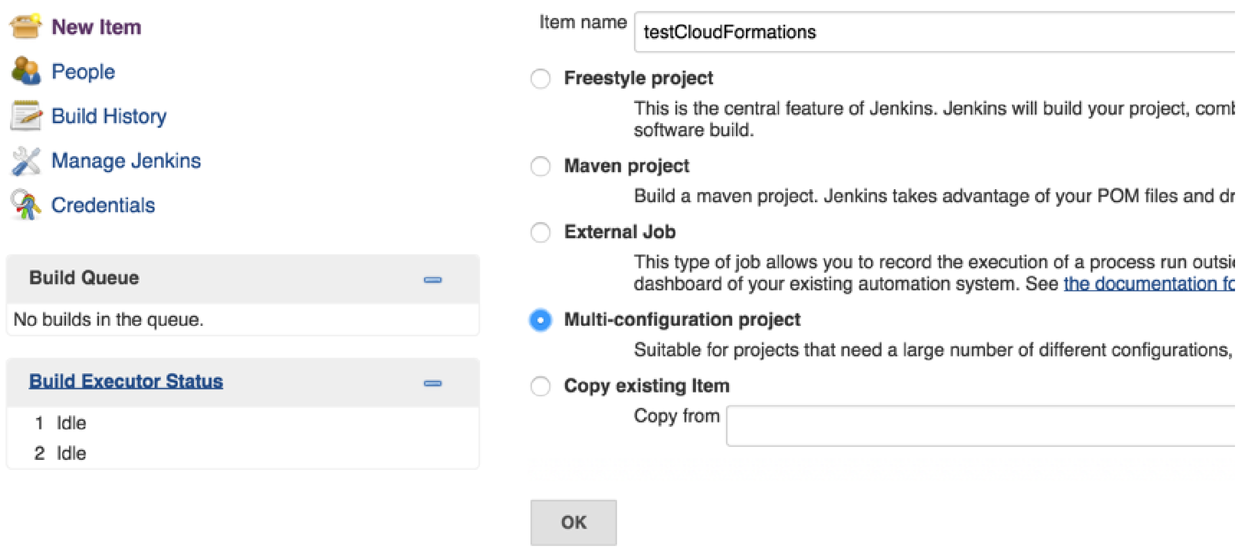
Jenkins will be trigger to start new build when it receive a new web-hooks from the puppet repository.

A new stack will be build, with the following parameters that needed to be available.

- InstanceType
- KeyName
- PrivateIP
- SecurityGroupID
- Subnet

Jenkins Configuration

1) Go to manage Jenkins and install the github plugins and the cloudFormation plugins.
2) Click on new item and create a new job



New Item
People
Build History
Manage Jenkins
Credentials

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle

Item name: testCloudFormations

( ) Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, comt
software build.

( ) Maven project
Build a maven project. Jenkins takes advantage of your POM files and dr

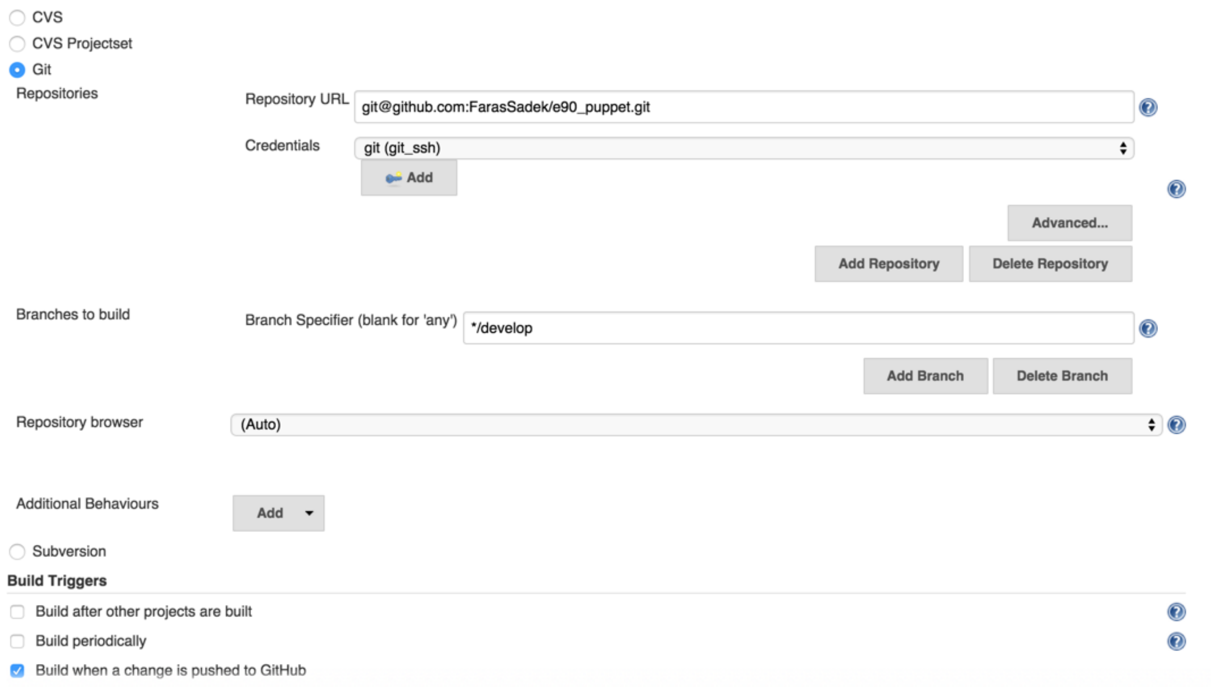
( ) External Job
This type of job allows you to record the execution of a process run outsi
dashboard of your existing automation system. See the documentation fc

(●) Multi-configuration project
Suitable for projects that need a large number of different configurations,

( ) Copy existing Item
Copy from

OK

3) Put the configuration for git as below.



( ) CVS
( ) CVS Projectset
(●) Git

Repositories
Repository URL: git@github.com:FarasSadek/e90_puppet.git
Credentials: git (git_ssh)
Add
Advanced...
Add Repository Delete Repository

Branches to build
Branch Specifier (blank for 'any'): */develop
Add Branch Delete Branch

Repository browser: (Auto)

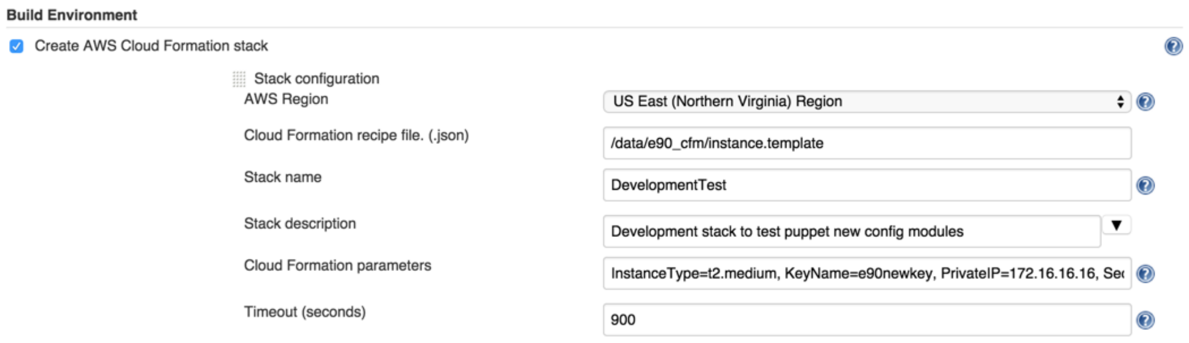
Additional Behaviours: Add

( ) Subversion

Build Triggers
☐ Build after other projects are built
☐ Build periodically
☑ Build when a change is pushed to GitHub

4) Configure CloudFormation plugin as shown below, pay attention to cloudFormation parameters.

Cloud Formation



Build Environment

☑ Create AWS Cloud Formation stack

Stack configuration

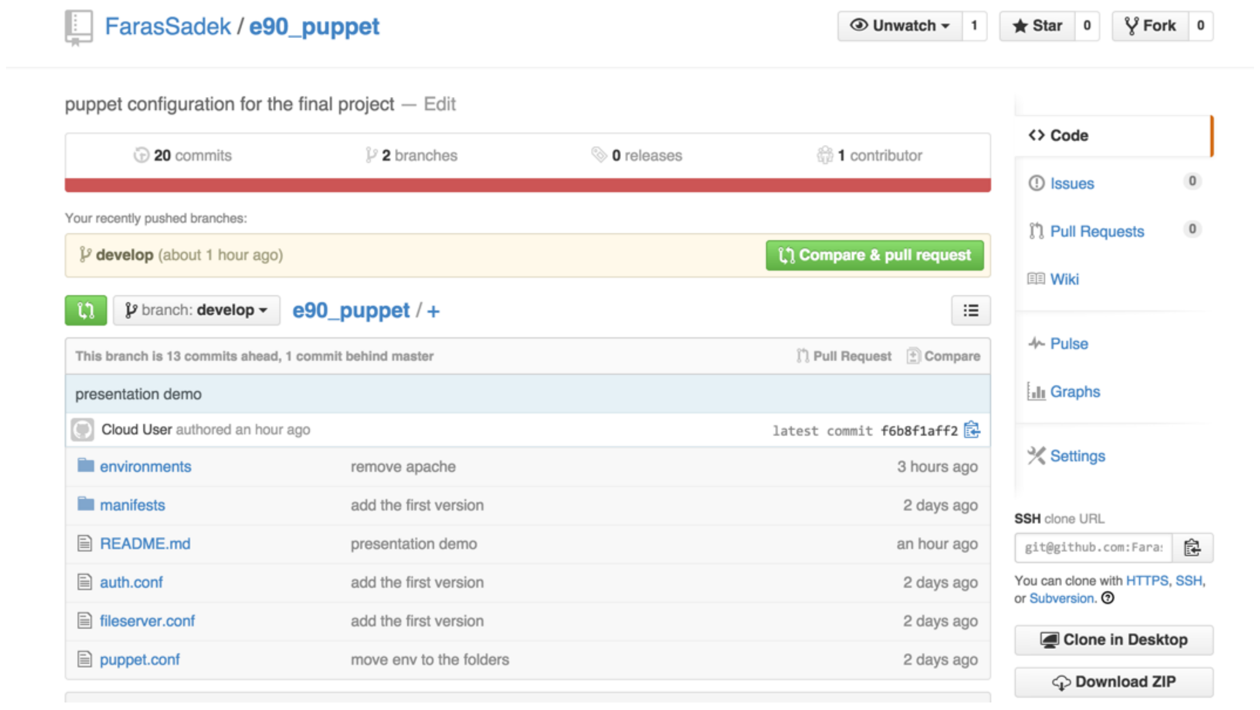
AWS Region	US East (Northern Virginia) Region
Cloud Formation recipe file. (.json)	/data/e90_cfm/instance.template
Stack name	DevelopmentTest
Stack description	Development stack to test puppet new config modules
Cloud Formation parameters	InstanceType=t2.medium, KeyName=e90newkey, PrivateIP=172.16.16.16, Se
Timeout (seconds)	900

Cloud Formation

GitHub

Two repositories are stored in GitHub , the CloudFormation Templates and the Puppet configuration. A web-hooks service is created for the puppet repository, such that it will trigger Jenkins when a new push arrive to the repository, and here is how I create the web-hook service:

1) Go to the repository link.
2) Click on the setting.
3) Click on the webhooks and services
4) Create a new service.
5) Search for Jenkins-github
6) Put the link of your Jenkins URL appended with /github-webhooks



puppet configuration for the final project — Edit

20 commits | 2 branches | 0 releases | 1 contributor

Your recently pushed branches:

develop (about 1 hour ago)

branch: develop e90_puppet / +

This branch is 13 commits ahead, 1 commit behind master

presentation demo

Cloud User authored an hour ago latest commit f6b8f1aff2

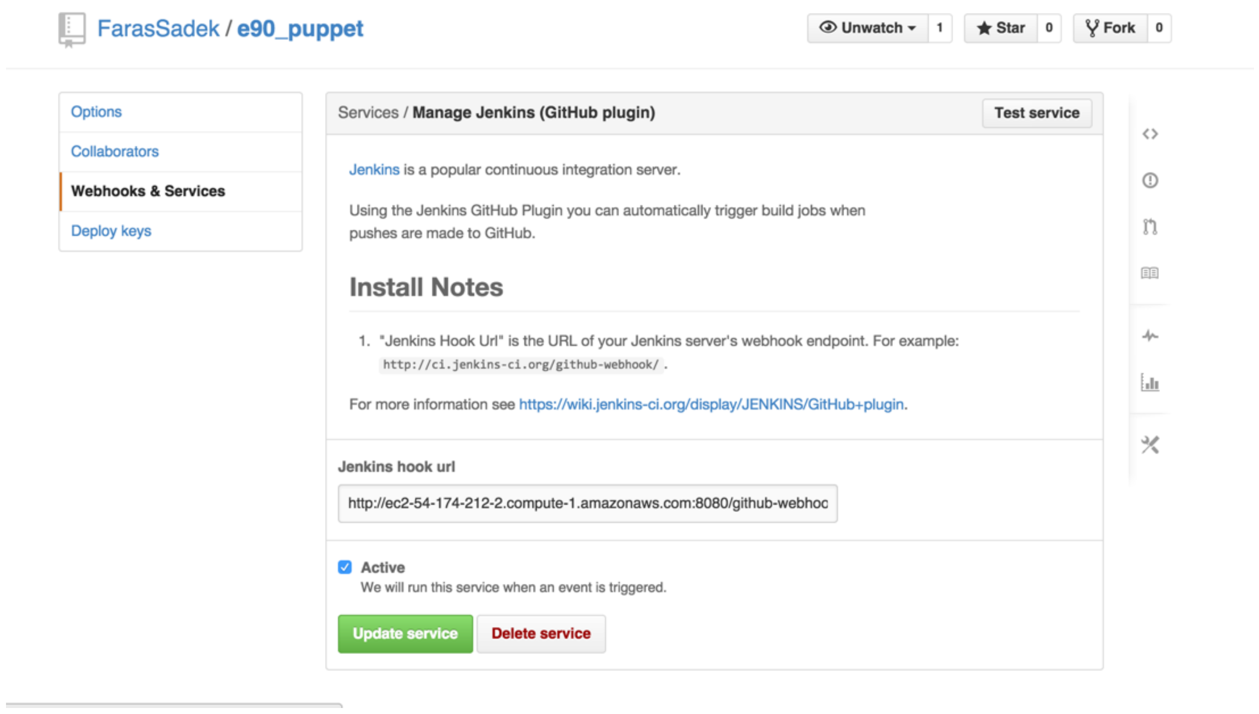
environments	remove apache	3 hours ago
manifests	add the first version	2 days ago
README.md	presentation demo	an hour ago
auth.conf	add the first version	2 days ago
fileserver.conf	add the first version	2 days ago
puppet.conf	move env to the folders	2 days ago

SSH clone URL

git@github.com:Fara:

You can clone with HTTPS, SSH, or Subversion.

Cloud Formation



Services / Manage Jenkins (GitHub plugin)

Jenkins is a popular continuous integration server.

Using the Jenkins GitHub Plugin you can automatically trigger build jobs when pushes are made to GitHub.

Install Notes

1. "Jenkins Hook Url" is the URL of your Jenkins server's webhook endpoint. For example: `http://ci.jenkins-ci.org/github-webhook/` .

For more information see https://wiki.jenkins-ci.org/display/JENKINS/GitHub+plugin.

Jenkins hook url

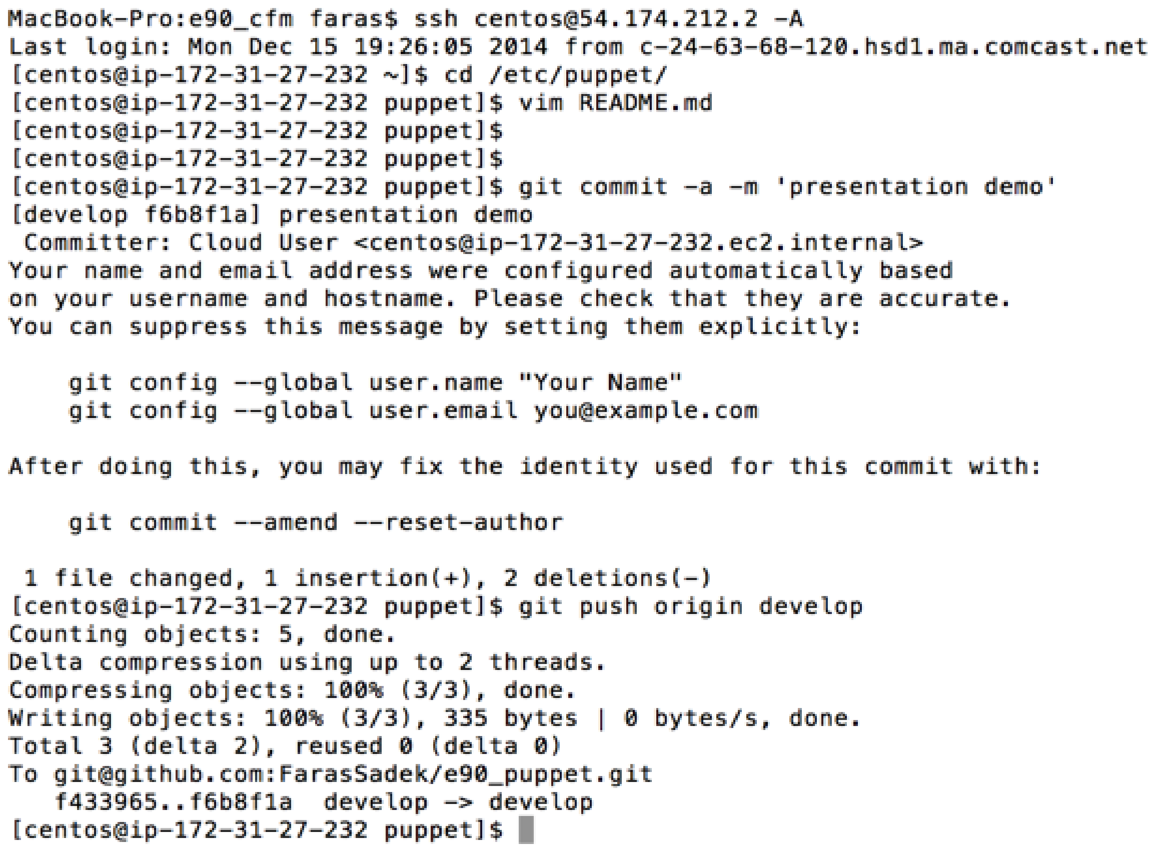
http://ec2-54-174-212-2.compute-1.amazonaws.com:8080/github-webhoo

[✓] Active
We will run this service when an event is triggered.

Cloud Formation

How it Work? or the build Process

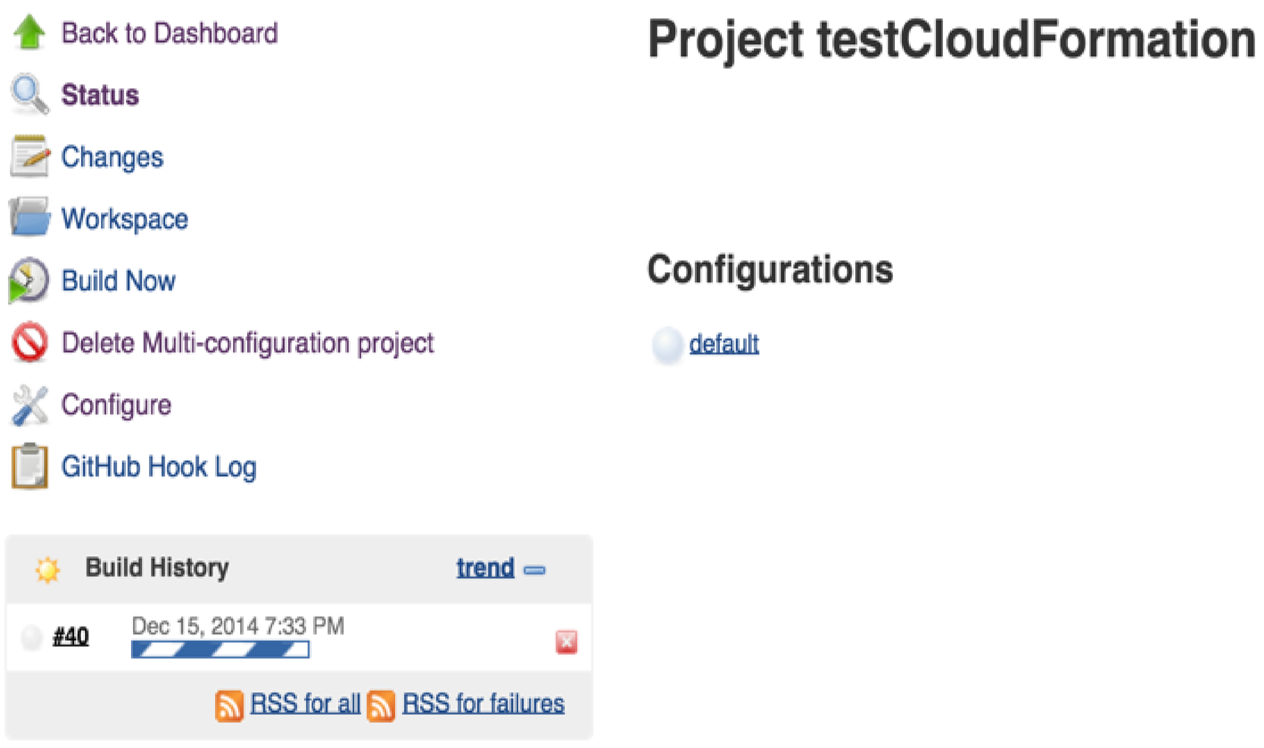
1) Change, commit and push change to the develop branch of the puppet master



```
MacBook-Pro:e90_cfm faras$ ssh centos@54.174.212.2 -A
Last login: Mon Dec 15 19:26:05 2014 from c-24-63-68-120.hsd1.ma.comcast.net
[centos@ip-172-31-27-232 ~]$ cd /etc/puppet/
[centos@ip-172-31-27-232 puppet]$ vim README.md
[centos@ip-172-31-27-232 puppet]$
[centos@ip-172-31-27-232 puppet]$
[centos@ip-172-31-27-232 puppet]$ git commit -a -m 'presentation demo'
[develop f6b8f1a] presentation demo
 Committer: Cloud User <centos@ip-172-31-27-232.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+), 2 deletions(-)
[centos@ip-172-31-27-232 puppet]$ git push origin develop
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 335 bytes | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
To git@github.com:FarasSadek/e90_puppet.git
   f433965..f6b8f1a  develop -> develop
[centos@ip-172-31-27-232 puppet]$
```

2) The push will send a web-hook to Jenkins, then a new build will start in Jenkins if this push is on the develop branch.

Cloud Formation


Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Multi-configuration project
Configure
GitHub Hook Log

Build History trend

#40 Dec 15, 2014 7:33 PM

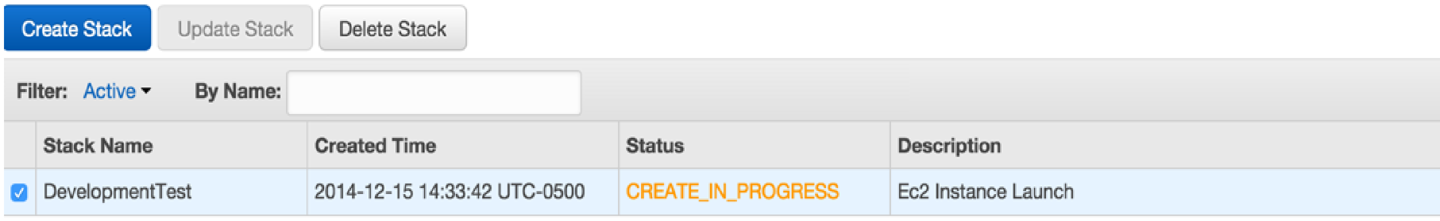
RSS for all RSS for failures

Project testCloudFormation

Configurations

default

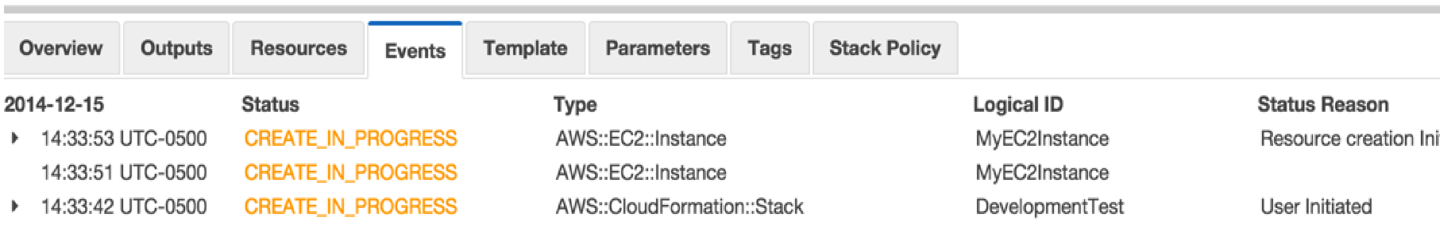
3) New stack in my AWS environment will be started. And a new machine in the Development VPC will be created. The Instance should contact the puppet master and ask for a configuration. The configuration is basic, simply add a file called testfile in the /tmp folder.



Create Stack | Update Stack | Delete Stack

Filter: Active By Name:

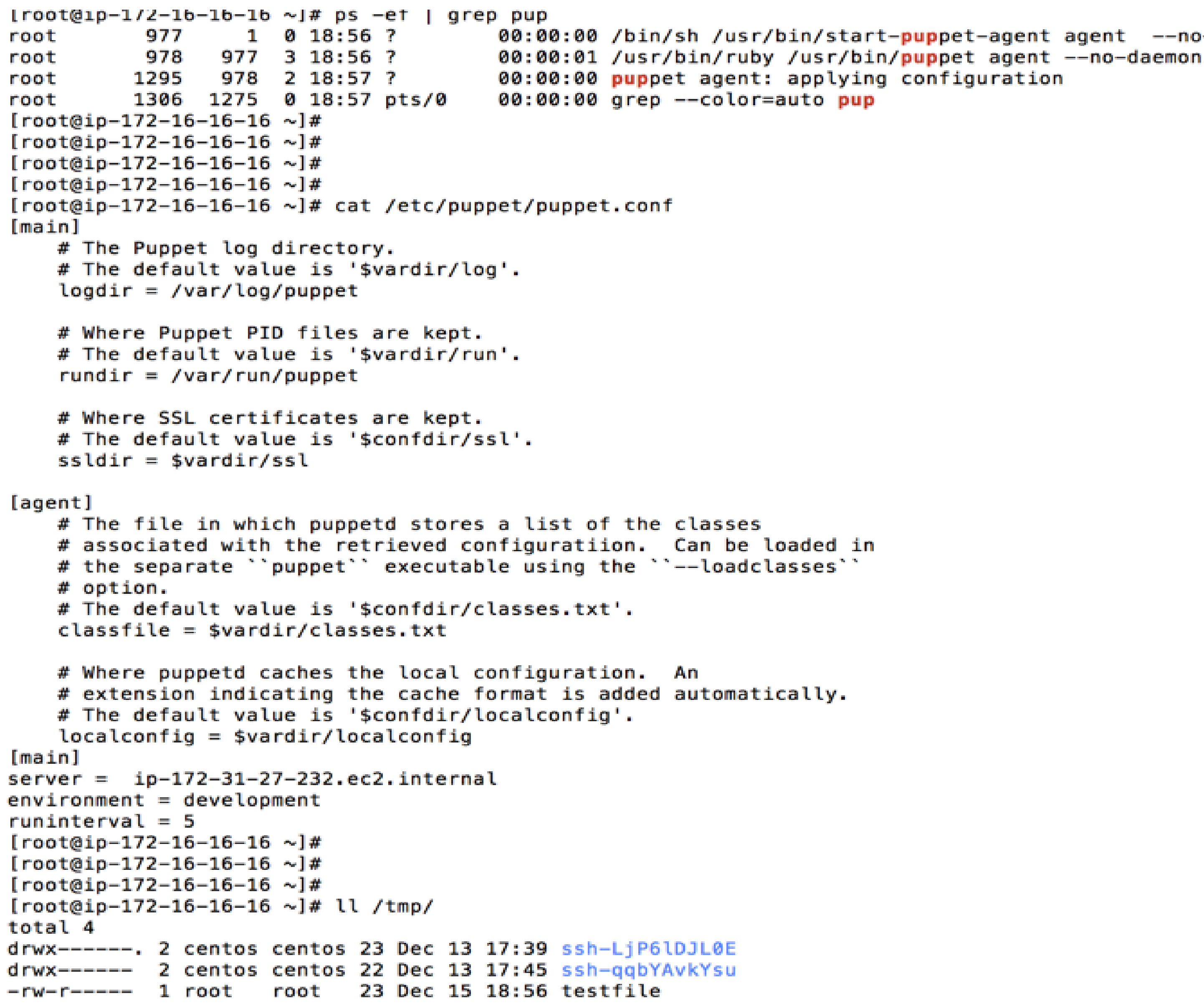
	Stack Name	Created Time	Status	Description
✓	DevelopmentTest	2014-12-15 14:33:42 UTC-0500	CREATE_IN_PROGRESS	Ec2 Instance Launch



Overview | Outputs | Resources | Events | Template | Parameters | Tags | Stack Policy

2014-12-15	Status	Type	Logical ID	Status Reason
14:33:53 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	MyEC2Instance	Resource creation Ini
14:33:51 UTC-0500	CREATE_IN_PROGRESS	AWS::EC2::Instance	MyEC2Instance	
14:33:42 UTC-0500	CREATE_IN_PROGRESS	AWS::CloudFormation::Stack	DevelopmentTest	User Initiated

4) Login to the new instance, and check if the file /tmp/testfile is created by puppet



```
[root@ip-172-16-16-16 ~]# ps -ef | grep pup
root       977     1  0 18:56 ?        00:00:00 /bin/sh /usr/bin/start-puppet-agent agent  --no
root       978   977  3 18:56 ?        00:00:01 /usr/bin/ruby /usr/bin/puppet agent --no-daemon
root      1295   978  2 18:57 ?        00:00:00 puppet agent: applying configuration
root      1306  1275  0 18:57 pts/0    00:00:00 grep --color=auto pup
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]# cat /etc/puppet/puppet.conf
[main]
    # The Puppet log directory.
    # The default value is '$vardir/log'.
    logdir = /var/log/puppet

    # Where Puppet PID files are kept.
    # The default value is '$vardir/run'.
    rundir = /var/run/puppet

    # Where SSL certificates are kept.
    # The default value is '$confdir/ssl'.
    ssldir = $vardir/ssl

[agent]
    # The file in which puppetd stores a list of the classes
    # associated with the retrieved configuratiion.  Can be loaded in
    # the separate ``puppet`` executable using the ``--loadclasses``
    # option.
    # The default value is '$confdir/classes.txt'.
    classfile = $vardir/classes.txt

    # Where puppetd caches the local configuration.  An
    # extension indicating the cache format is added automatically.
    # The default value is '$confdir/localconfig'.
    localconfig = $vardir/localconfig
[main]
server =  ip-172-31-27-232.ec2.internal
environment = development
runinterval = 5
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]#
[root@ip-172-16-16-16 ~]# ll /tmp/
total 4
drwx------. 2 centos centos 23 Dec 13 17:39 ssh-LjP6lDJL0E
drwx------  2 centos centos 22 Dec 13 17:45 ssh-qqbYAvkYsu
-rw-r-----  1 root   root   23 Dec 15 18:56 testfile
```

5) As shown above the instance puppet configuration and puppet agent is running which create the file /tmp/testfile from the puppet server. the puppet server DNS was a parameter in the CloudFormation template.

Cloud Formation